

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 1 – Introduksjon til programmering

Innhold

Kapittel 1

• Analog introduksjon til programmering.....	3
• Introduksjon til viktige begreper.....	5
• Introduksjon til Scratch.....	7
• Introduksjon til micro:bit.....	9
• Introduksjon til MakeCode.....	11
• Sensorer.....	13
• Feilsøking.....	14
• Hvordan virker en servo.....	15
• Miniutfordringer.....	17

Analog introduksjon til programmering

Oppgave 1

Dere skal jobbe i par.

Begge bygger en liten figur av de utdelte legoklossene. Ta vare på figuren din helt til slutten, men ikke vis den til noen andre.

Lag en oppskrift slik at den andre du jobber med klarer å bygge en helt lik figur. Du får ikke forklare ekstra underveis, alt må stå skrevet ned.

Så bytter dere oppskrift og bygger hverandres figurer.

- Ble de like?
- Hva måtte du endre i oppskriften for at figurene skulle bli helt like?
- Kunne du laget oppskriften kortere eller enklere?



Oppgave 2

Dere skal jobbe i par.

Begge lager en oppskrift på hvordan den andre må gå for å komme seg ut døra. Det er ikke lov å forklare underveis.

Så bytter dere oppskrift, og en om gangen prøver ut den andres oppskrift og ser hvor du havner.

Etterpå justerer dere begge oppskriftene, men da kan dere bare bruke kodeklossene på neste side.

Hvor få klosser er mulig å bruke for å få til oppgaven?



Feilsøking og forbedring

1. Skjedde det som skulle skje? / Fikk du til oppgaven?
2. Hva måtte du endre på i oppskriften for å få til oppgaven?
3. Kunne du laget oppskriften kortere eller enklere?

Kodeklosser til analog programmering

Gå ett steg

Gå to steg

Gå tre steg

Disse klossene sier hvor mange steg du skal gå bortover.

Sving 90° mot høyre

Sving 90° mot venstre

Snu 180°

Disse klossene sier hvor mange grader du skal snu.

Stopp

Denne klossen sier at du skal stoppe opp og stå i ro.

Gjenta 5 ganger

Gjenta 10 ganger

Gjenta 20 ganger

Dette er løkker. De sier hvor mange ganger du skal gjenta noe. Hvor mange ganger du skal gå rundt gjennom løkka.

Dermed må du alltid koble den sammen en annen kloss som sier akkurat hva du skal gjenta i løkka di.

Hvis du krasjer, så

Hvis du sovner, så

Hvis du snubler, så

Dette er hvis-klosser. De inneholder en betingelse, altså HVIS noe skjer, SÅ skal du gjøre noe.

Dermed må du alltid koble den sammen en annen kloss som sier akkurat hva du skal gjøre i tilfelle at du krasjer, sovner eller snubler.

Introduksjon til viktige begreper

- vilkår, hvis-klosser og løkker

Vilkår

Et vilkår er en påstand som kan være sann eller usann. Vi sammenligner ofte to tall og ser hvilket som er størst, eller om de er like store.

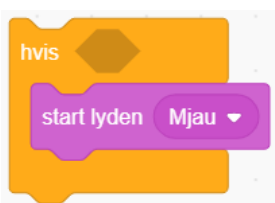
Eksempler på vilkår:

- Mamma er høyere enn pappa.
- Det er høyere temperatur om sommeren enn om vinteren.
- Læreren har lik skostørrelse som jeg har.

I Scratch er vilkårene grønne, sekskanta klosser.

Hvis-blokker

- Hva: En hvis-kloss inneholder en betingelse. Når betingelsen er sann, så vil det som står inni hvis-klossen skje.
- Hvorfor: En hvis-kloss gjør at et program kan gjøre flere forskjellige ting avhengig av om betingelsen er sann eller usann.



Diskuter

For hvilke av de tre grønne betingelsene vil vi høre mjauing med programmet til venstre?

Løkker

- Hva: Løkker gjør at en del av koden i et program blir gjentatt flere ganger. Det finnes flere typer løkker. Noen gjentar det du skal gjøre et bestemt antall ganger, mens andre løkker fortsetter evig. Den tredje typen gjentar frem til et vilkår ikke lenger er sant.



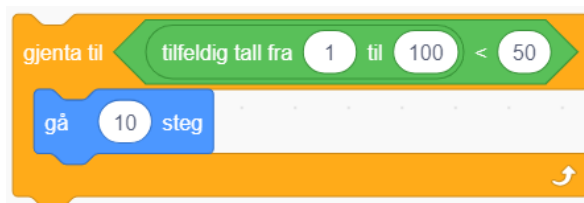
- Hvorfor: Programmet blir lettere å få oversikt over, og det blir færre klosser enn om man ikke bruker løkker.

Oppgaver

1. Lag to vilkår som er sanne og to som er usanne.
2. Er det alltid enkelt å vite om et vilkår er sant eller usant?
3. Hvilke type vilkår kan en datamaskin forstå?

Grubleoppgave

Hva tror du skjer i programmet under?
Vil det samme skje hver gang?



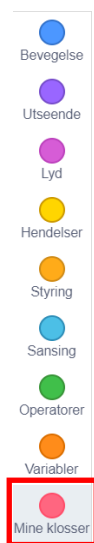
Introduksjon til funksjoner i programmering

- hvordan lage egne klosser

Hva? En funksjon er en kloss som du har laget selv. Altså et slags eget program. Det som skiller en funksjon fra et vanlig program, er at du først må definere en funksjon. Det vil si at du må lage den. Etterpå må du kjøre funksjonen i et program. Det som står inni en funksjon skjer ikke av seg selv før du bruker funksjonen i et program.

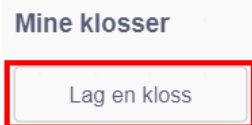
Hvorfor? Programmet du lager tar mindre plass, akkurat som når du bruker løkker. Funksjonen kan brukes flere steder i programmet uten å måtte lage alle blokkene på ny.

Hvordan?

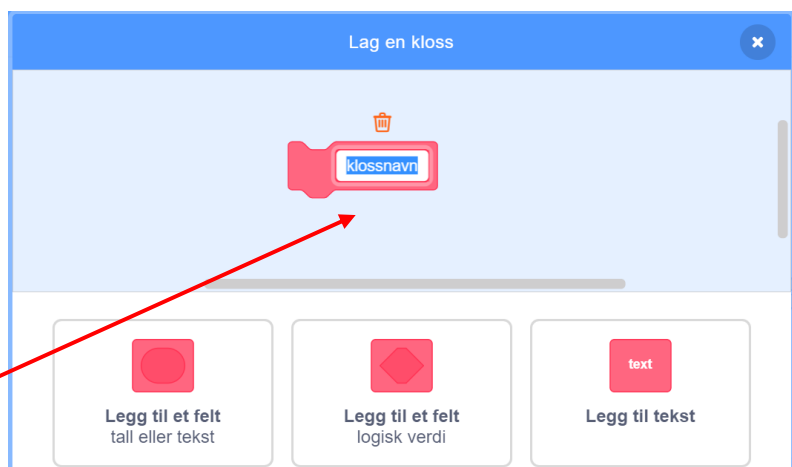


1. steg: Velg mine klosser i menyen på venstre side.

2. steg: Trykk på lag en kloss-knappen som da dukker opp.



3. steg: Nå må du skrive inn hva funksjonen din skal hete der det står klossnavn.



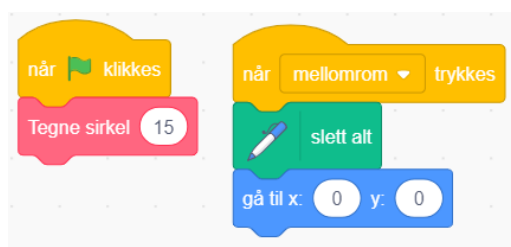
4. steg: Du kan velge å legge til om funksjonen din trenger å få noe informasjon for å virke. Dette kalles ofte for inndata eller input. Det trenger vi for eksempel om funksjonen skal regne ut forskjellige regnestykker.

Dersom du lager en funksjon som skal regne om en verdi fra meter til centimeter, må funksjonen din få vite hvor mange meter du vil gjøre om til centimeter. Da må du legge til et tall når du oppretter klossen til funksjonen din.



Diskuter

1. Hva gjør funksjonen til venstre?
2. Hvorfor er det en funksjon, og ikke et eget program?
3. Hva gjør programmet under?
4. Hvorfor er dette et program?



Oppgave

Lag en funksjon for at katten beveger seg i disse formene:

1. Kvadrat
2. Rektangel
3. Trekant

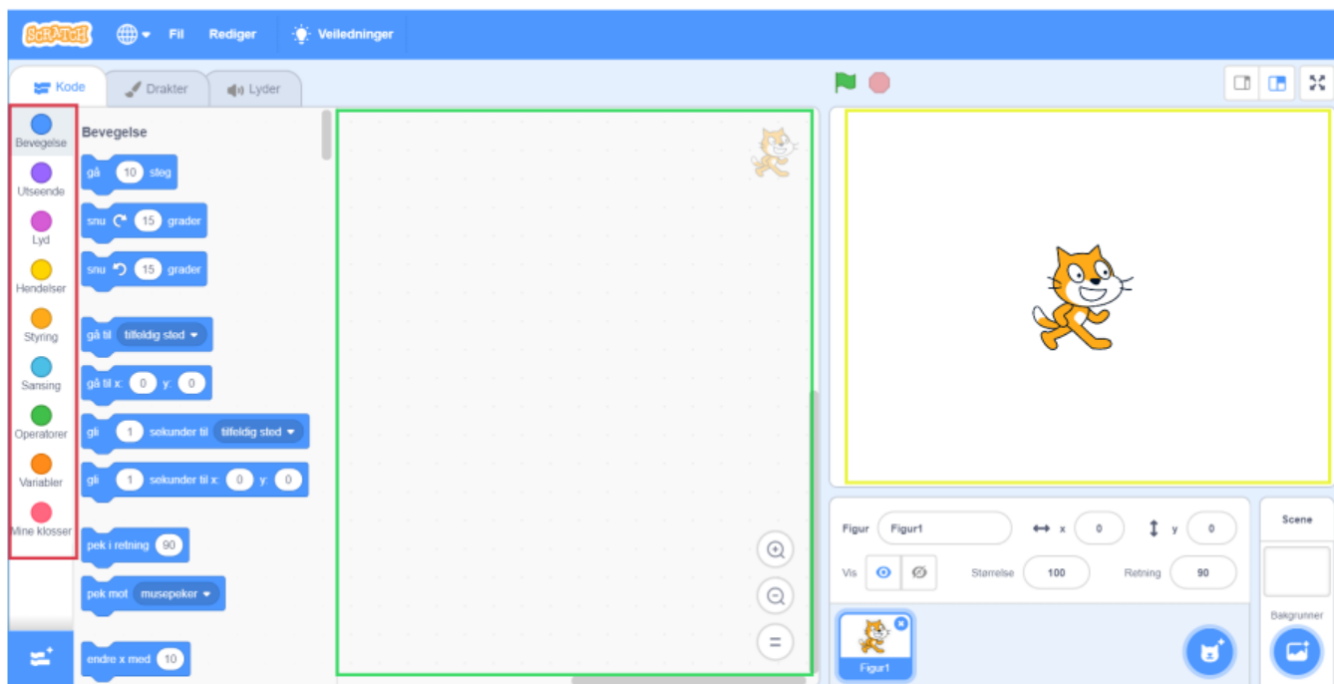
Kan du bruke funksjonene dine til å tegne en sammensatt figur?

Scratch

- en introduksjon



Gå inn på nettsiden: <https://scratch.mit.edu>



I den røde ramma er det ulike kategorier som gjør det enklere å finne de klossene man trenger.

Bevegelse: Her ligger klossene for hvor katten skal bevege seg.

Styring: Her ligger klossene som sier hvor mange ganger noe skal gjøres eller om det skal gjøres i det hele tatt.

Operatorer: Her ligger en del regneoperasjoner, som addisjon, subtraksjon og sammenligninger.

I den grønne ramma er feltet der man lager programmet.

Du trekker klossen du vil bruke inn hit, da er den en del av programmet ditt.

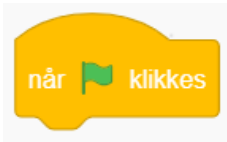
I den gule ramma ser du resultatet av programmet ditt.

Katten vil bevege seg slik programmet gir den beskjed om.

Dette området fungerer som et koordinatsystem, der katten starter i origo.

Intro til noen klosser i Scratch

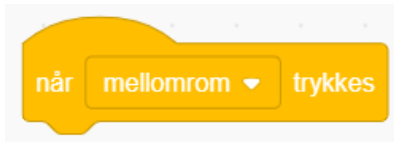
- Alle disse klossene finner du i kategorien som har lik farge som klossene



Når man klikker på det grønne flagget, skjer det som står i klossen koblet fast i denne.



Denne klossen kalles en hvis-kloss. Og hvis det som står i det grå feltet er sant, så vil det som står inni denne klossen skje.



Når man klikker på mellomrom-tasten, skjer det som står i klossen som er festet i denne.



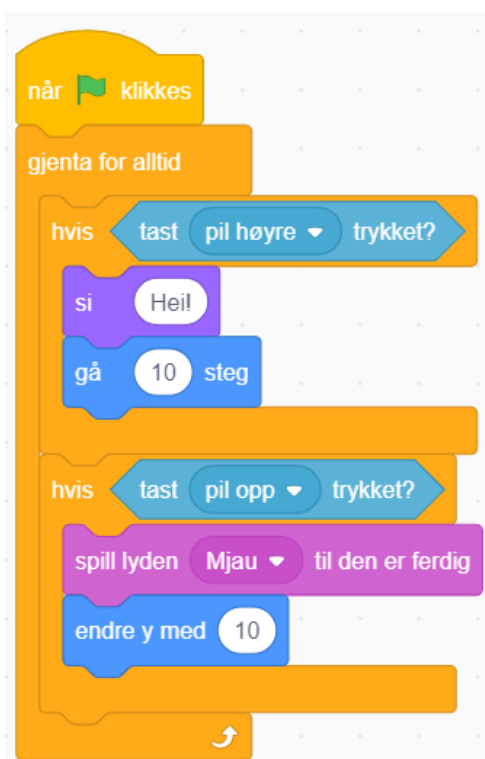
Eksempel på en betingelse som kan settes inn i det grå feltet til hvis-klossen. Dersom vi putter inn et tall som er større enn 50, vil det som står inne i hvis-klossen skje.



Denne klossen kalles en løkke. Det som står inne i denne klossen vil gjentas for alltid.



Denne klossen sier hvilke koordinater katten skal gå til.



Oppgave

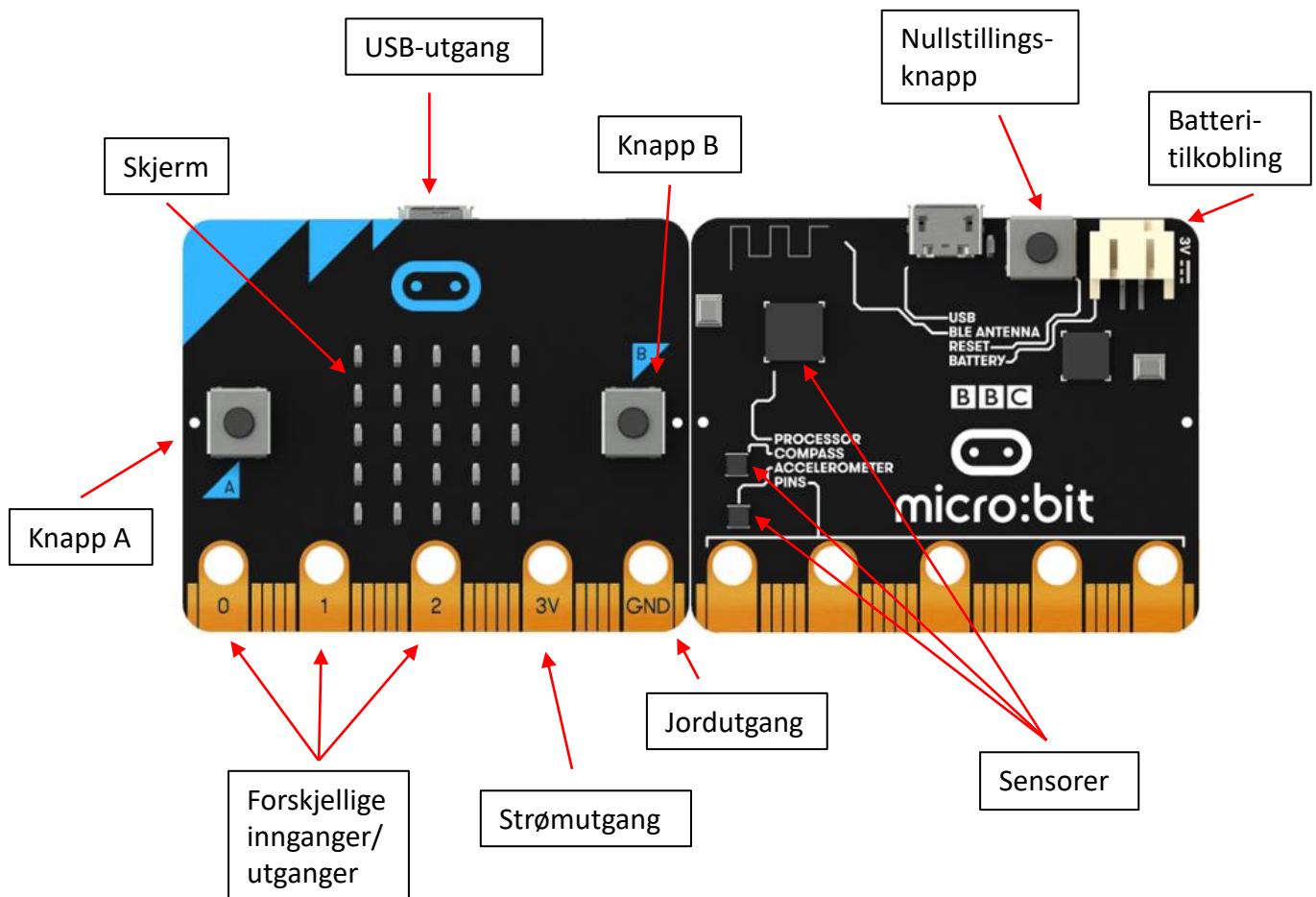
1. Gjett hva dette programmet gjør.
 - Fyll inn i det utdelte arket
2. Lag programmet i Scratch og se hva som skjer.
3. Hva må du gjøre for at katten skal kunne komme tilbake til origo?
 - HINT: Tenk på x- og y-koordinater.

Ekstraoppgave

1. Kan du få katten til å bevege seg i en rektangelform uten å bruke noen taster?
2. Hva med en trekantform?

Introduksjon til micro:bit

Micro:bit er en mikrokontroller, det vil si en enkel datamaskin som vi kan programmere. Vi kan programmere ved hjelp av en datamaskin, et nettbrett eller en mobiltelefon.



Skjerm: Micro:biten har en «skjerm» som består av 25 LED-lamper som vi kan programmere til å lyse, blinke og vise forskjellige bilder, tall eller bokstaver.

Knapper: Den har to knapper på framsiden, og en nullstillingsknapp på baksiden. Nullstillingsknappen gjør at programmet på micro:biten begynner på ny.

Mini-usb: Her kan du koble micro:bit til en USB-kabel dersom du bruker en datamaskin med USB-utgang.

Batteritilkobling: Her kan du koble til batterier for at micro:biten skal få strøm.

Sensorer: Micro:biten har noen innebygde sensorer som kan måle akselerasjon, temperatur og kompassretninger.

Tilkoblinger: Det går an å koble på andre sensorer, motorer, LED-pærer eller høyttalere på micro:biten ved hjelp av alligatorledninger.

Bluetooth (blåtann): Micro:biten har bluetooth-tilkobling (blåtann-tilkobling) slik at den kan få overført programmer fra for eksempel iPad som ikke har USB-utgang.

Introduksjon til bruk av micro:bit

- for datamaskiner

Gå inn på nettadressen: <https://makecode.microbit.org/#editor>

Da kommer du inn på denne nettsiden hvor du kan begynne å programmere micro:biten.

Simuleringsområde

I den gule ramma ser du resultatet av programmet ditt.

Klosseliste

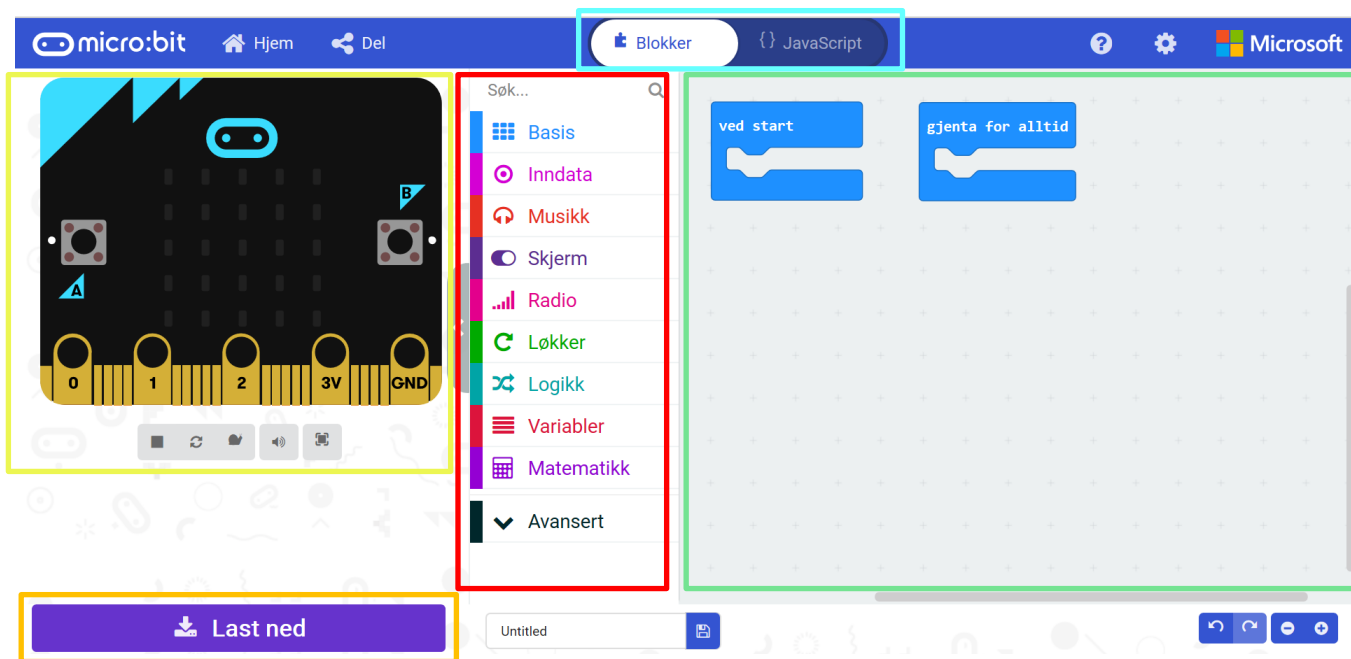
I den røde ramma er det ulike kategorier som gjør det enklere å finne de klossene man trenger.

Når du trykker på en av kategoriene, vil klossene som hører til komme fram rett til høyre.

Programmeringsområde

I den grønne ramma er feltet der man lager programmet.

Du trekker klossen du vil bruke inn hit, da er den en del av programmet ditt.



Nedlastingsområde

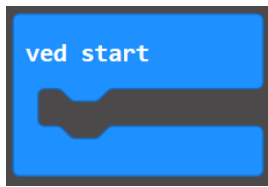
I den oransje ramma er kan vi laste programmet over til micro:biten.

Programmeringsspråk

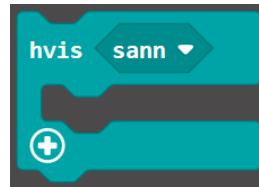
I den turkise ramma er feltet der du kan velge om du skal bruke klosser å programmere med, eller et tekstbasert programmeringsspråk kalt Python eller JavaScript.

Intro til MakeCode for micro:bit

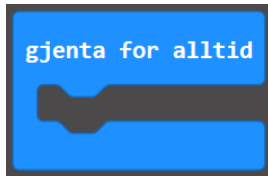
- Alle disse klossene finner du i kategorien som har lik farge som klossene



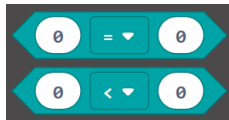
Straks programmet er overført til micro:biten skjer det som står inni denne klossen.



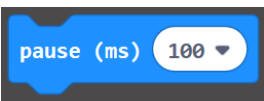
Denne klossen kalles en hvis-kloss. Og hvis det som står i det mørkegrønne feltet er sant, så vil det som står inni denne klossen skje.



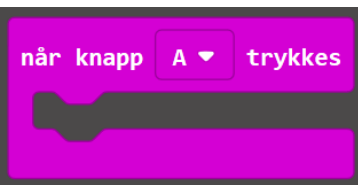
Denne klossen kalles en løkke. Det som står inne i denne klossen vil gjentas for alltid.



Eksempler på vilkår som kan settes inn i det mørkegrønne feltet i hvis-klossen. I det øverste vilkåret kan vi sette inn to tall som den sjekker om er like. I den nederste kan vi sette inn to tall der vilkåret sjekker om det første er mindre enn det andre.



Denne klossen gjør at programmet tar en pause før den går videre til neste kloss.



Denne klossen utfører det som står inni den når man trykker på knapp A på micro:biten.

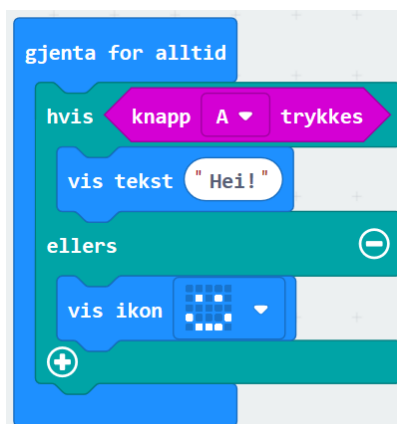
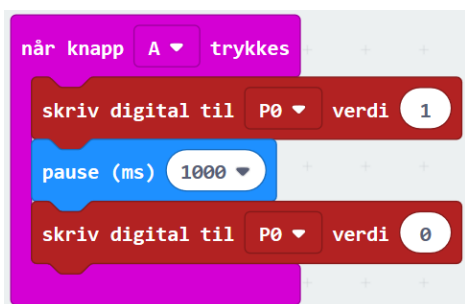


Når verdien er 1, vil denne klossen gjøre at det sendes strøm ut gjennom P0-utgangen. Når verdien er 0, sendes det ikke strøm ut gjennom utgangen.

Output til skjerm

Et viktig poeng med å bruke micro:bit er at vi kan vise informasjon på den. Det kalles gjerne utdata eller output, og kan vises på skjermen til micro:biten. Skjermen er ikke en vanlig skjerm, men består av 25 røde lysdioder som kan være av- eller påskrudd.

I micro:biten har vi ulike klosser som kan vise ting på skjermen. Vi kan vise tekster, bilder eller tall. Tekstene kan være litt vanskelige å lese, for de ruller over skjermen og består av en blanding av store og små bokstaver. Vi kommer til å vise flest tall i denne boka.



Oppgave

Gjett hva de to programmene gjør.

Lag programmene i makecode og overfør til micro:biten – ett om gangen.

Skjedde det du trodde?

Kan du gjøre noen morsomme endringer?

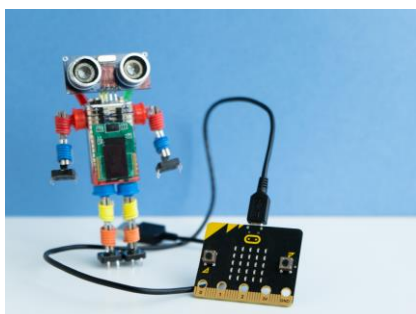
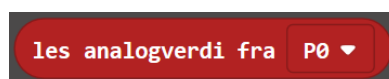
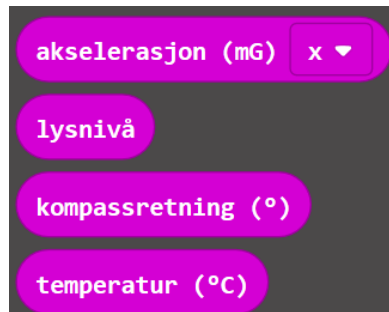
Inndata/input for micro:bit

Input fra sensorer

Mesteparten av poenget med å bruke micro:bit er at vi kan samle inn informasjon med den. Det kalles gjerne inndata eller input, og kommer fra forskjellige sensorer. De kan være innebygd i selve micro:biten, eller vi kan koble til andre sensorer.

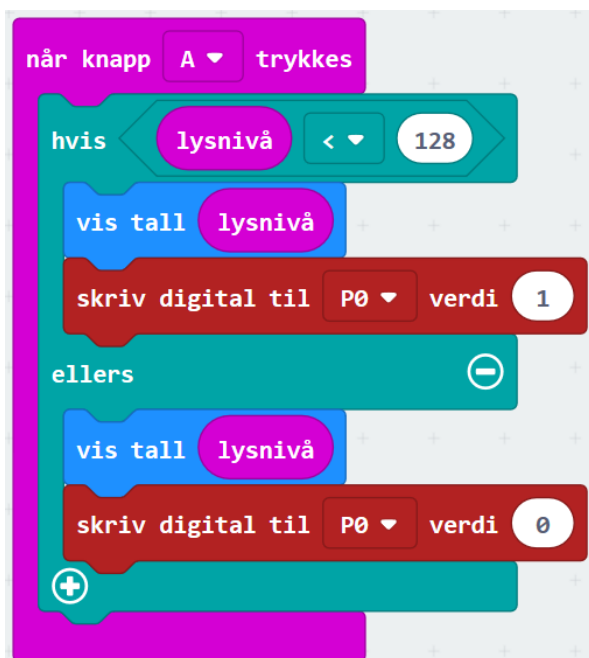
I micro:biten har vi innebygde sensorer som kan måle akselerasjonen, lysnivået, kompassretningen og temperaturen. Vi kommer oftest til å bruke temperaturen i denne boka.

Dersom vi ønsker å bruke en sensor som ikke er innebygget i micro:biten, må vi først koble den til en inngang på micro:biten. I tillegg trenger vi en kloss som leser av hvilken verdi sensoren sender til micro:biten. Da bruker vi denne klossen. Den leser bare av hva verdien er, men gjør ingenting med den. De fleste sensorer vi kobler på gir verdier mellom 0 og 1023, men det kan variere fra sensor til sensor. Det står mer om sensorer og hvordan de virker på neste side.



Snakk om

1. Hvilke sensorer vet du om?
2. Hvordan kan en micro:bit vise fram målingene fra sensorene?
3. Hva kan man bruke forskjellige sensorer til?
4. Kan du tenke ut en oppfinnelse som bruker en sensor for å løse en oppgave.



Oppgave

1. Gjett hva dette programmet gjør.
2. Lag programmet i MakeCode, overfør det til micro:biten og se hva som skjer.
3. Hva kan det tenkes at skal være koblet til micro:biten?
4. Hva ville du gjort med programmet for å forbedre det?

Sensorer

En sensor blir brukt til å måle forskjellige verdier, slik som temperatur, lysstyrke, fart, akselerasjon eller lydnivå. Sensoren kobles til en micro:bit og man kan måle hvor stor spenning som kreves for å trykke elektronene gjennom den. Da er det slik at i en temperatursensor vil spenningen variere i samme takt som temperaturen, og vi kan bruke spenningen vi måler til å regne ut nøyaktig temperatur.



Termistor
- måler temperatur



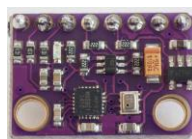
Lyssensor
- måler lysstyrke



Halleffekt-sensor
- måler magnetfelt



Ultralydsensor
- måler avstand



Akselerometer
- måler akselerasjon



Lydsensor
- Måler lydstyrke

Hva er motstand?

Når et materiale har stor motstand, er det vanskelig for elektronene å bevege seg gjennom materialet. Da trenger man høy spenning for å dytte elektronene gjennom. Motstand blir også kalt resistans, og enheten er ohm.

Elektriske ledere har liten motstand slik at elektronene kan bevege seg lett gjennom dem, mens isolatorer har veldig stor motstand slik at det er nærmest umulig for elektroner å bevege seg gjennom dem.

I sensorer varierer motstanden, og da vil også spenningen variere. Denne spenningen er enkel å måle med en micro:bit. I en lyssensor vil motstanden variere med lysstyrken, og vi får målt forskjellig spenning om det er mørkt eller lyst. For en ultralydsensor vil spenningen variere ut fra hvor stor avstand sensoren har til en gjenstand, og tilsvarende for andre sensorer.

Feilsøking

- debugging



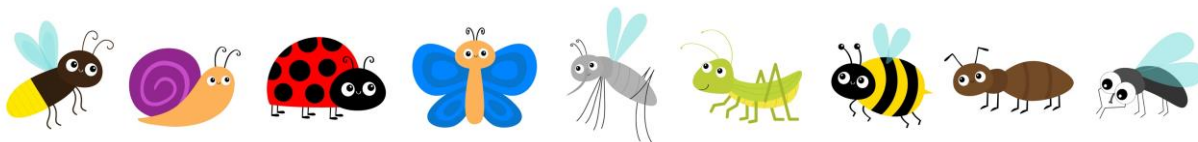
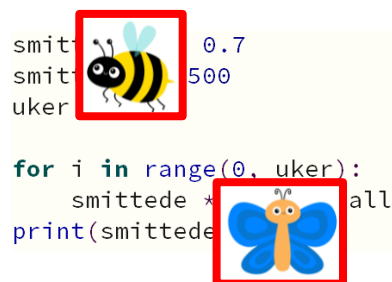
Finn meg!

Absolutt ingen som programmerer gjør alt riktig på første forsøk. Ikke engang de aller beste programmererne som har jobbet med programmering i mange år! Når programmet ikke oppfører seg som planlagt, trenger vi å finne ut hva som er årsaken til det. Vi må finne feilene i koden vår, også kalt bugs. Det kan være lurt å lage seg en egen liste med vanlige feil for deg, og bruk denne lista for å finne bugs i nye programmer.

Eksempel på en sjekkliste for feilsøking

- gjør gjerne det som er enklest først, ikke slett hele programmet

1. Kjør programmet og se hva som skjer
 - Hva er det som ikke stemmer med det du ønsker at programmet skal gjøre?
 - Skjer ting i den rekkefølgen du hadde planlagt?
 - Får du noen feilmeldinger (tekst-programmering)?
2. Gjør synlig det som skjer inni programmet og kjør det
 - Skrive variablene til skjerm
 - Legge inn pauser i koden for å rekke å se hvilke verdier forskjellige variabler har
3. Endre på programmet og kjør det
 - Har variablene riktig navn med stor/liten bokstav alle steder du bruker samme variabel?
 - Har løkkene riktig antall repetisjoner?
 - Endre bare på en ting om gangen, slik at det er enkelt å se hva en forandring fører til
 - Gir feilmeldinger noen hint om i hvilke linjer feilen ligger (tekst-programmering)?
 - Gir feilmeldinger noen hint om hva feilen er (tekst-programmering)?
 - Feil variabeltype
 - Glemt innrykk
 - Manglende kolon eller andre tegn
 - Søk på feilkoder på internett (tekst-programmering)



Søke etter svar

- Selv de flinkeste programmererne bruker mye av tiden på å søke opp feilkoder på internett. Det kan også være lurt å søke på navnet til forskjellige funksjoner for å se hvilke inndata de skal ha, og hvilke utdata de gir.
- Står det noe i læreboks di? Les gjerne nærmere gjennom oppgaven.
- Spør noen i klassen eller læreren.

Hvordan virker en servo?

En servo er en liten elektrisk motor som kan dreie frem og tilbake eller rundt og rundt. Den som bare kan dreie frem og tilbake kalles en 180°-servo. Den som kan dreie rundt og rundt kalles en 360°-servo (eller kontinuerlig servo).

Vinkler



Servoene får navn fra vinkler. En vinkel kan være mellom 0° og 360°. En vinkel på 90° kalles en rett vinkel og ser slik ut.



En vinkel på 180° er det dobbelte av en rett vinkel, og da får vi faktisk en rett linje!



Hvis du tar fire rette vinkler, ender du opp med en vinkel på 360°. Da har du kommet helt rundt! De servoene som kalles 360°-servoer går jo rundt og rundt.



Hva betyr kontinuerlig?

Noe som fortsetter uten opphold.

En 180°-servo kan bare dreie 180°, og ikke rundt og rundt.

Det går typisk ikke an å styre farten til denne typen servoer, bare vinkelen de dreier til.

Programmering av servoer

Da kommer du til å trenge noen av disse klossene.

180°-graders servo:

Denne setter hvilken vinkel servoen skal dreie til.

360°-graders servo:

Denne setter farten til servoen.

Denne stopper servoen.

Sett vinkel på servo P0 til 90°

servo P0 kjører kontinuerlig på 50 %

stans servo P0

Diskuter

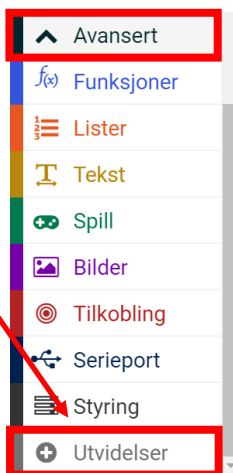
Hva må du gjøre for å få 360°-servoen til å dreie i motsatt retning?

Disse klossene finner du slik

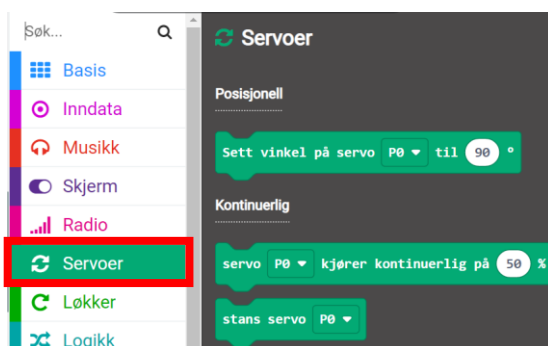
1. Velg avansert i menyen, den er svart.

2. Trykk der det står utvidelser, nederst.

3. Så velger du servo-utvidelsen.



4. Velg servo i menyen, så finner du klossene du trenger.



Servoene trenger litt mer strøm enn det som kommer fra micro:biten. For å være sikker på at den får nok strøm til å virke slik den skal, må man bruke et kretskort som gjør at vi kan koble til batterier i tillegg til micro:biten.

Dere kan la være å legge til servo-tillegget, da må dere bruke en annen kloss enn vist på forrige side. Det er bare en kloss uavhengig av hvilken type servo man bruker, men verdiene man skriver inn får ulikt resultat ut fra servotypen.

180°-servo: Verdien angir vinkelen servoen skal dreie til.

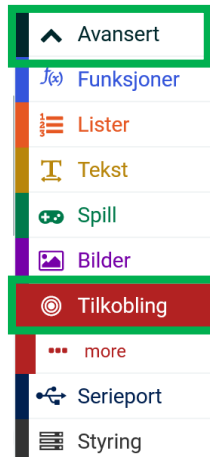
360°-servo: Verdien angir dreiehastighet som servoen dreier rundt og rundt med.

servo skriv til P0 ▼ verdi 180

Denne klossen finner du slik

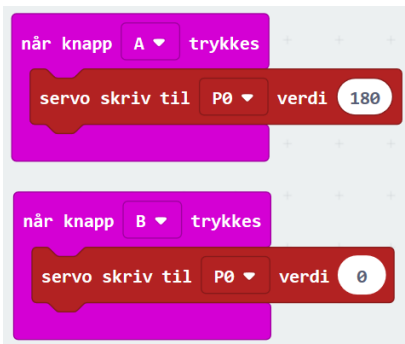
1. Velg avansert i menyen, den er svart.

2. Gå nedover i menyen til du finner tilkobling. Den er mørkerød.



3. Så finner du klossen du trenger nest nederst (og nederst).

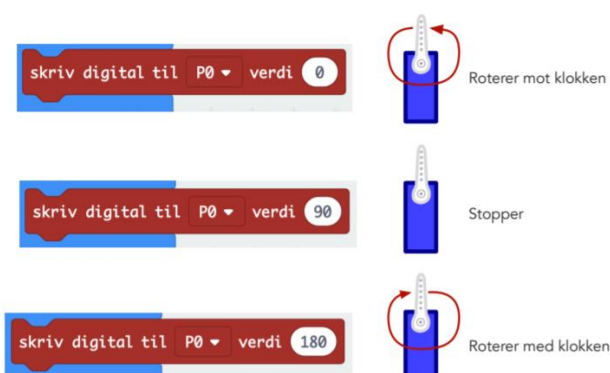
Programmering av 180°-servoer



En 180°-servo kan dreie 180° hver vei, men ikke dreie helt rundt. Man kan typisk ikke styre hastigheten til denne typen servoer, bare hvilken vinkel de dreier til.

Det øverste programmet gjør at servoen dreier 180° i den ene retningen. Det nederste programmet gjør at servoen dreier tilbake til utgangspunktet.

Programmering av 360°-servoer (kontinuerlige servoer)



Om vi skriver verdien 0 til servoen, vil den rotere mot klokken så fort den klarer.

Når vi øker verdien, vil den rotere saktere helt til vi kommer til 90 hvor motoren står stille.

Når vi øker verdien over 90 opp mot 180 grader roterer den raskere og raskere med klokken.

Her er verdien altså ikke vinkelen, men dreiehastighet.

Miniutfordringer

Det skjeve tårn i Pisa



Materialer: Papplata (30 cm x 30 cm), 18 papirark, tape og saks.

Tidsbruk: Planlegging 2 min, gjennomføring 10 min.

Krav:

Bruk de utdelte materialene.

Bygg tårn på papplata med gjennomsnittshøyde på minst 160 cm.

Klossene må være i kontakt med plata og plata skal være intakt.

Utskytningsmekanisme



Materialer: Alt som ligger klart.

Tidsbruk: Planlegging 2 min, gjennomføring 10 min.

Krav:

Lag en utskytningsmekanisme som får svampen til å fly minst 5 meter.

Utskytningsmekanismen må bestå av 5 ulike materialer.

Uidentifisert flygende objekt (UFO)



Materialer: Alt som ligger klart.

Tidsbruk: Planlegging 2 min,
gjennomføring 10 min.

Krav:

Lag et flygende objekt som skal være i luften lengst mulig.

Bruk de utdelte materialene.

Det flygende objektet skal dra en blyant.

Bakkeløp



Materialer: Alt som ligger klart.

Tidsbruk: Planlegging 2 min,
gjennomføring 10 min.

Krav:

Lage en gjenstand som bruker nøyaktig 5 sekunder på å trille ned papplatebakken.

Bruk de utdelte materialene.

Det er ikke lov å berøre gjenstanden etter at den er sluppet.

Brobygging



Materialer: Alt som ligger klart.

Tidsbruk: Planlegging 2 min,
gjennomføring 10 min.

Krav:

Bruk de utdelte materialene.

Bygg en bro med et spenn på minst 50 cm.

Broa skal tåle en full melkekartong midt på brospennet (1 kg).

Bordbasket



Materialer: Alt som ligger klart.

Tidsbruk: Planlegging 2 min,
gjennomføring 10 min.

Krav:

Ballen må følge en bane og gå gjennom en kurv.

Ballen kan ikke berøres etter at den er satt i gang.

Kurven som ballen skal gå gjennom må lages.

Banen må være minst 2 meter og kan ikke være en rett linje.

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 2

Innhold

Kapittel 2

• Opplegg 1 – Innovasjonsmetoden.....	3
• Demonstrer for ditt standpunkt.....	4
• Opplegg 2 – Celler.....	5
• Lag en modell av en celle.....	6
• Opplegg 3 – Drivhuseffekten.....	7
• Lag en modell av drivhuseffekten.....	8
• Opplegg 4 – Energiformer og –overganger.....	9
• Lag en kjedereaksjon.....	10

Opplegg 1 - Innovasjonsmetoden

- Hvordan løse en oppgave?

1. Undersøke og finne informasjon

- Let etter informasjon om temaet.
- Bruk ulike kilder.
- Finn ut av hva vanskelige begreper betyr.



2. Planlegge og idémyldre

- Bruk den informasjonen du fant for å prøve å finne løsninger.
- Finn flest mulig løsninger og velg ut den løsningen du tror er best.
- Planlegg gjennomføringen.



3. Utføre

- Tegn, bygg eller lag modellen.
- Lag første versjon av dataprogrammet (dersom det er en del av oppgaven).



4. Teste

- Prøv ut det du lagde i punkt 3.
- Gjør eventuelle målinger.



5. Evaluere

- Virker det du lagde slik som du hadde tenkt?
- Hva kan du gjøre for at det skal virke bedre?



6. Forbedre

- Gjør det du fant ut under punkt 6 at vil få det du lagde til å virke bedre.
- Gå gjerne tilbake til punkt 1 – 6 flere ganger.



7. Dokumentere

- Gjør en ny test av det du har laget, virker det nå?
- Ta et bilde eller skriv en liten tekst om det du har laget og hvordan det virker.

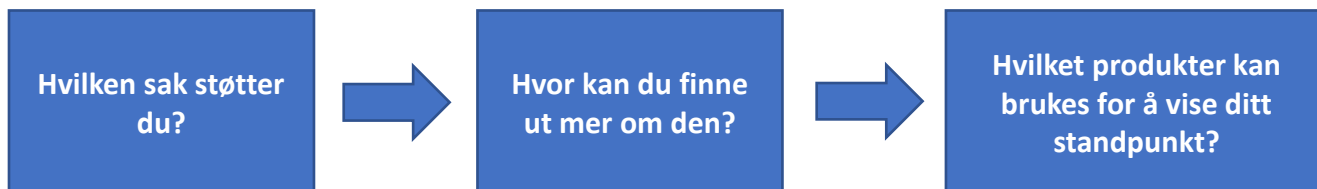


Demonstrer for ditt standpunkt



Bruk innovasjonsmetoden for å lage et produkt som kan benyttes for å fremme ditt syn på en viktig sak – det kan være en tegning, plakat eller noe helt annet.

Fase 1: Undersøke og finne informasjon



Fase 2: Idémyldre og planlegge

Hva er det som gjør at du støtter saken du valgte?

Hvordan kan du lage et produkt som overbeviser andre?

Hvilke virkemidler kan du bruke for å få fram standpunktet ditt?



Fase 3: Lag en skisse av produktet.

Fase 4: Spør andre om de ser hvilken sak du demonstrerer for.

Fase 5: Kan du forbedre produktet på noen måte?

Fase 6: Lag produktet.

Fase 7: Skriv en forklaring på hvorfor du valgte å demonstrere for akkurat den saken ved hjelp av det produktet du gjorde.

Opplegg 2 - Celler

Alle levende organismer er bygget opp av en eller flere celler. De fleste cellene er så små at man trenger et mikroskop for å se dem, men menneskets eggceller går an å se med det blotte øye. Det finnes mange forskjellige typer celler, og i menneskekroppen er det omtrent 220 ulike typer. Det kan være alt fra muskelceller til nerveceller og kjønnsceller. I menneskekroppen har vi omtrent 37 billioner celler (37 000 000 000 000).

Organeller (cellens organer)

Cellemembran – er en hinne rundt cellen. Den slipper inn næringsstoffer og slipper ut avfallsstoffer og andre produkter.

Cytosol – en tyktflytende væske som fyller innsiden av cellen. Den består i hovedsak av vann, mineraler og næringsstoffer.

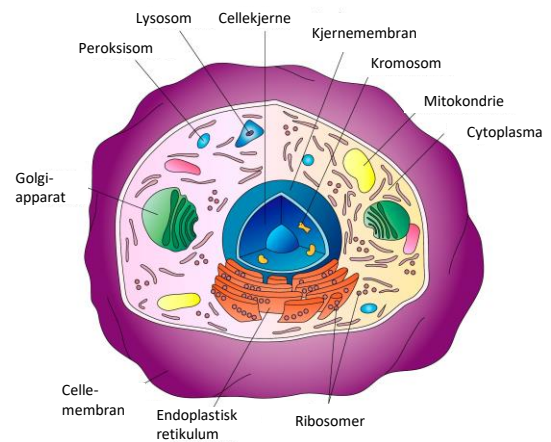
Cellekjerne – cellens kommandosentral, den bestemmer hva som skjer i cellen.

DNA – er arvestoffet vårt og befinner seg inni cellekjernen. Det bestemmer hvilke proteiner kroppen vår produserer.

Ribosom – er viktig i prosessen der cellen lager proteiner.

Mitokondrier – cellens kraftverk, de omformer den kjemiske energien fra maten vi spiser til energi kroppen kan bruke (celleånding).

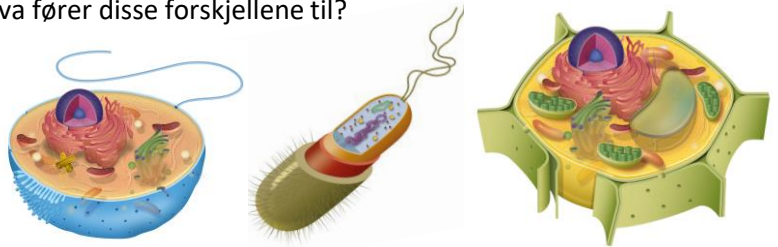
Lysosomer – cellens avfallsstasjon, de bryter ned avfallsstoffene i cellen.



Snakk om

Kan dere finne ut noen forskjeller på dyreceller, planteceller og encellede organismer er? HINT: Cellevegg, kloroplaster, lysosomer og cellekjerne.

Hva fører disse forskjellene til?



Du kommer til å trenge denne klossen. Når verdien er 1, vil denne klossen gjøre at det sendes strøm gjennom LED-pæra slik at den lyser.

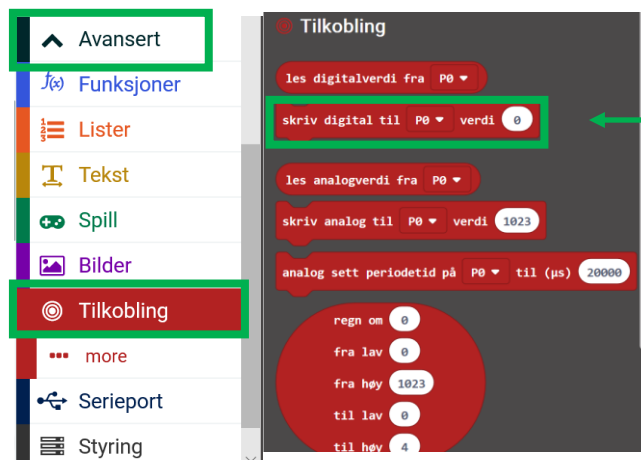
Verdi 1: Strøm på

Verdi 0: Strøm av

skriv digital til P0 ▼ verdi 0

Denne klossen finner du slik

- Velg avansert i menyen, den er svart.
- Gå nedover i menyen til du finner tilkobling. Den er mørkerød.



- Så finner du klossen du trenger nest øverst.

Lag en modell av en celle

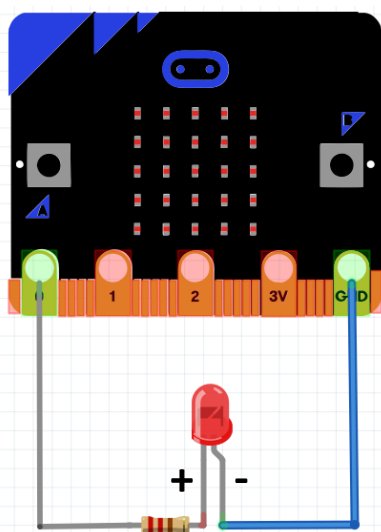
Oppgave

Lag en modell av en celle som har minst én lysende eller blinkende LED-pære.

Fase 1: Finn informasjon og inspirasjon til en cellemodell. Hvilke deler består den av? Hvordan ser disse delene ut?

Fase 2: Ha en idémyldring for deg selv. Hvordan vil du at din cellemodell skal se ut, og hvilke materialer skal du bruke? Tegn gjerne en skisse før du begynner å lage den. Cellemodellen skal også ha minst ett lys. Hvilke farger skal disse ha og hvor passer de best?

Fase 3: Tid for å lage cellemodellen og programmere micro:biten. Se under for oppkobling og programmering.



Koble opp micro:bit som vist på figuren.

Det lange beinet til LED-pæra skal kobles til en mostand og så til P0-utgangen. Dette blir den positive polen, og vi kan lage et program som sender strøm gjennom pæra.

Det korte beinet til pæra skal kobles til GND-utgangen. Det står for ground som betyr jord, og det blir den negative polen.

Hvilke endringer må vi gjøre for å få flere LED-pærer til å lyse?

gjenta for alltid

skriv digital til P0 ▼ verdi 1

Eksempel på program

Dette programmet får LED-pæra til å lyse konstant. Hva må du gjøre for å få den til å blinke?

Fase 4: Test programmet ditt.

Fase 5: Virker det slik det skal?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få en best mulig cellemodell. Gjør gjerne endringer i micro:bit-programmet ditt.

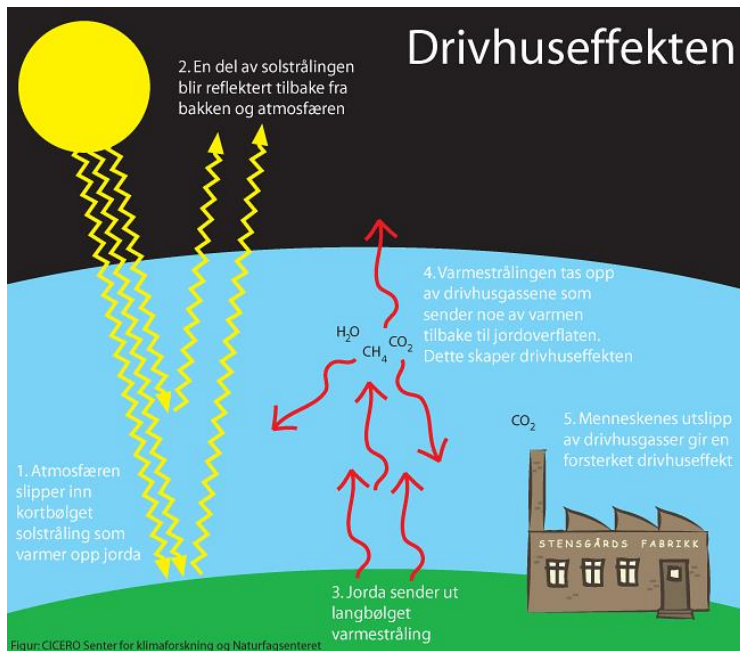
Fase 7: Ta gjerne en filmsnutt av den ferdige cellemodellen for å vise hvordan den virker.

Oppgave

Bare planteceller inneholder kloroplaster som gjør at de kan omforme energi fra sollyset til kjemisk energi i karbohydrater. Mens alle celler kan omforme energien fra næringsstoffer til den energien de trenger for å leve. Denne prosessen kalles celleånding.

1. Finn ut hvilke stoffer som inngår i reaksjonene for fotosyntese og celleånding.
2. Hva har disse stoffene med karbonkretsløpet å gjøre?
3. Kan dette påvirke drivhuseffekten på noen måte?

Opplegg 3 - Drivhuseffekten



Drivhuseffekten gjør at gjennomsnittstemperaturen på jordkloden er omtrent 15°C i stedet for -18°C , som det ville vært uten noen drivhuseffekt.

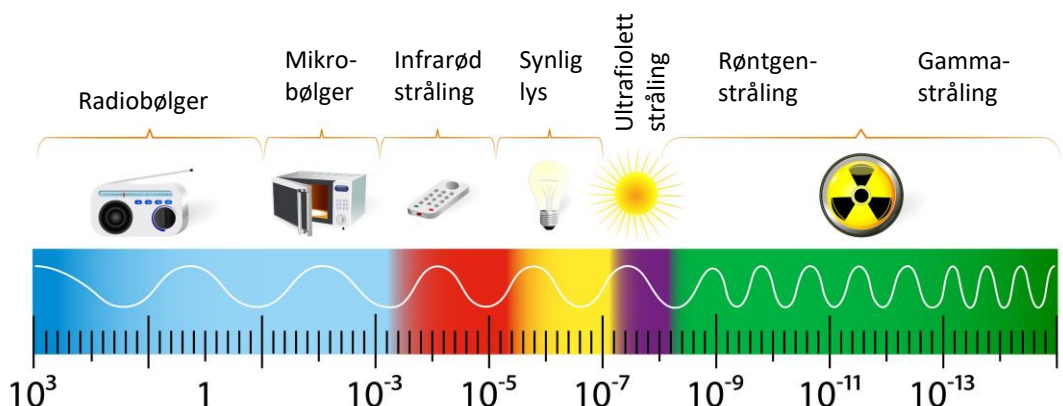
Det blir sendt ut stråling med mye energi fra sola, og denne treffer jordas atmosfære. En mindre del av denne strålingen forsvinner bare ut i verdensrommet, for atmosfæren og skyene virker som et speil og reflekterer en del av strålingen bort. Men mesteparten av denne strålingen blir absorbert (tatt opp) av atmosfæren og bakken. Da har jordkloden mottatt energi.

Jordkloden sender også ut sin egen stråling, varmestråling. Det er siden

jordkloden har en høyere temperatur enn verdensrommet. Verdensrommet har en temperatur på omtrent -270°C .

Varmestrålingen som jorda sender ut igjen, tas i stor grad opp av drivhusgassene i atmosfæren. Mengden av varmeenergi som blir tatt opp i atmosfæren, blir større når det blir mer av drivhusgassene (klimagassene). Da vil gjennomsnittstemperaturen på jordkloden øke, slik at den blir høyere enn 15°C . Dersom gjennomsnittstemperaturen stiger med over 2°C , vil det få store konsekvenser for livet på jorda. Store landområder vil komme under vann, det vil bli mer ekstremvær som kraftige stormer og uvær, og langvarig tørke som kan føre til stor fare for skogbranner. Dette vil påvirke både mennesker og dyre- og plantelivet på jordkloden.

De vanligste drivhusgassene er vanndamp (H_2O), karbondioksid (CO_2), metan (CH_4), lystgass (N_2O) og ozon (O_3). Av disse gassene er det i hovedsak CO_2 mennesker slipper ut. Den øker drivhuseffekten fra hvor kraftig den var før den industrielle revolusjon, da vi begynte å bruke kull til dampkraft.



Mer om stråling og bølgelengde

Strålingen fra sola er kortbølget, det vil si at bølgelengden er liten, og fargen vi kan se er gul/hvit. Strålingen med liten bølgelengde er den som inneholder mest energi. Denne strålingen blir absorbert av jorda, og sendt ut igjen som mer langbølget varmestråling. Varmestråling blir også kalt infrarød stråling og er usynlig for mennesker, siden den har større bølgelengde enn 800 nm. Mennesker kan bare se lys med bølgelengde mellom 400 nm og 800 nm der lys med bølgelengde på 400 nm tilsvarer en lillafarge og lys med bølgelengde 800 nm tilsvarer rød farge.

Lag en modell av drivhuseffekten

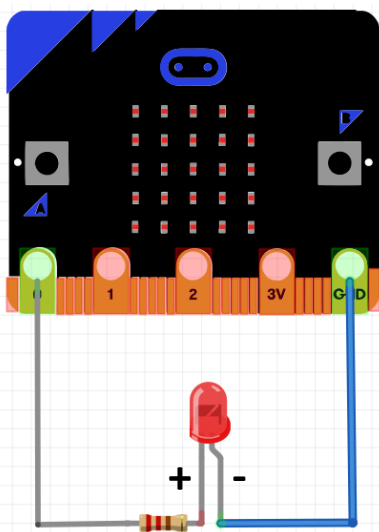
Oppgave

Lag en modell av drivhuseffekten som har minst én lysende eller blinkende LED-pære.

Fase 1: Finn informasjon og inspirasjon til en modell av drivhuseffekten. Hvilke deler består den av? Hvordan ser disse delene ut?

Fase 2: Ha en idémyldring for deg selv. Hvordan vil du at din modell skal se ut? Tegn gjerne en skisse før du begynner å lage den. Modellen skal også ha lys, hvilke farger skal disse ha og hvor passer de forskjellige fargene best?

Fase 3: Tid for å lage modellen, og programmere micro:biten. Se under for oppkobling og programmering.



Koble opp micro:bit som vist på figuren.

Det lange beinet til LED-pæra skal kobles til en mostand og så til P0-utgangen. Dette blir den positive polen, og vi kan lage et program som sender strøm gjennom pæra.

Det korte beinet til pæra skal kobles til GND-utgangen. Det står for ground som betyr jord, og det blir den negative polen.

Hvilke endringer må vi gjøre for å få flere LED-pærer til å lyse?

gjenta for alltid

skriv digital til P0 ▼ verdi 1

Eksempel på program

Dette programmet får LED-pæra til å lyse konstant. Hva må du gjøre for å få den til å blinke?

Fase 4: Test programmet ditt.

Fase 5: Virker det slik det skal?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få en best mulig modell. Gjør gjerne endringer i micro:bit-programmet ditt.

Fase 7: Lag en tilhørende tekst eller lydfil som forklarer hvordan drivhuseffekten virker og hvilke konsekvenser økt drivhuseffekt kan føre til.

Oppgave

De globale klimaendringene skyldes i hovedsak menneskers utslipp av CO₂.

1. Hvor mye har mengden CO₂ økt siden den industrielle revolusjon?
1. CO₂ er den klimagassen som er den minst effektive klimagassen, men likevel den det blir fokusert på. Hvorfor er det slik?
2. Konsekvenser av økt drivhuseffekt, er økt gjennomsnittstemperatur, som blant annet fører til mer ismelting og mer ekstremvær. Hvilke konsekvenser kan dette igjen ha for mennesker, dyr og planter?

Opplegg 4 - Energiformer og -overganger

Hva er energi?

Energi er det som får ting til å skje. Uten energi kunne ingenting bevege seg. Energi kan aldri oppstå eller forsvinne, den kan bare gå over til andre energiformer. Vi sier at energien er bevart.

Stillingsenergi (potensiell energi)

Når vi løfter en gjenstand høyere opp fra bakken, så tilfører vi den energi. Desto høyere opp gjenstanden er, desto høyere energi har den. Denne energien kan enkelt omformes til andre typer energi med en gang du slipper gjenstanden og den begynner å falle.

Bevegelsesenergi (kinetisk energi)

En gjenstand som beveger seg har mer energi jo større fart den har. Det er derfor det blir så mye større skader hvis en bil kolliderer når den har stor fart, siden energien er mye større enn om bilen hadde kjørt i lavere fart.

Lyd

Lyden energi finnes i lydbølgene som beveger seg til øret ditt. Lyd er egentlig trykkbølger i luft (eller andre materialer) som får luftmolekylene til å svinge. Siden luftmolekylene beveger seg, får de en type bevegelsesenergi som vi kaller lyden energi.

Varme

Varme er en form for energi, men en gjenstand kan ikke inneholde varme, den kan bare overføre varme. Varme er energi som går fra en gjenstand med høy temperatur til en gjenstand med lavere temperatur.

Elektrisk energi

Elektrisitet er ladninger som beveger seg, som regel elektroner. Da har disse en type bevegelsesenergi som vi kaller elektrisk energi.

Strålingsenergi (for elektromagnetisk stråling)

Stråling består av bølger av elektrisitet og magnetisme, man kan si at den beveger seg. Da har de en type bevegelsesenergi. Den kan forplante seg i lufttomt rom, og all energi vi får fra sola er av denne typen.

Kjemisk energi

Alle kjemiske forbindelser har lagret energi i bindingene sine. Denne energien kan frigis ved at vi bryter disse bindingene. For noen stoffer er det vanskelig å bryte bindingene for å frigi denne energien, men for noen stoffer er det veldig enkelt. I fossile brensler er dette enkelt ved å varme stoffet opp slik at vi får en forbrenningsreaksjon (se opplegg 31). Det er faktisk forbrenningsreaksjoner som gjør at vi får frigitt den kjemiske energien fra maten vi spiser.

Eksempel på energikjeder

Slippe en ball fra en høyde:

Stillingsenergi → bevegelsesenergi → lyden energi og varmeenergi

Bilkollisjon:

Bevegelsesenergi → lyden energi og varmeenergi



Snakk om

Du slipper en stein fra 1 meter over bakken, og etterpå slipper du den fra 2 meter over bakken.

1. Hva skjer med farten til steinen akkurat i det den treffer bakken i de to tilfellene?
2. Hvorfor er det slik, tror du?

Hvorfor kan vi ikke høre eksplosjoner som skjer i verdensrommet?

Energiproduksjon er egentlig bare energiomforming

Oftest snakker vi om å produsere energi, men det går ikke an å produsere eller lage energi, den kan jo bare omformes! Når vi snakker om energiproduksjon mener vi som regel å omforme en energiform til elektrisk energi. Eksempler på energiomforming:

- Vannkraft som omformer stillingsenergien til store vannmengder til elektrisk energi
- Vindkraft som omformer bevegelsesenergien til vinden til elektrisk energi
- Solceller som omformer strålingsenergien fra sola til elektrisk energi
- Fossil energi som omformer den kjemiske energien i olje, kull og gass til elektrisk energi

Hva er energikvalitet?

Den sier noe om hvor enkelt det er å bruke energien til noe nyttig. I en energikjede, vil energikvaliteten alltid synke når energien omformes til andre typer. Den blir mindre nyttig. Til slutt ender all energi opp som svak varmeenergi. Og den er veldig vanskelig å bruke til noe nyttig. Elektrisk energi er veldig enkelt å bruke til noe nyttig. Hva med de andre energitypene?

Lag en kjedereaksjon

- programmer en servo

Oppgave

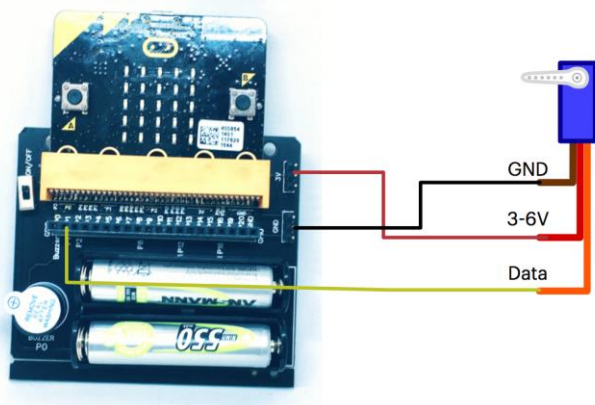
Lag en kjedereaksjon som forflytter en kule minst to meter uten at noen tar på den underveis. Den må bestå av minst tre deler og minst en servo.

Fase 1:



Fase 2: Hvor skal kjedereaksjonen få energien fra? Hvilke materialer og deler skal den bestå av? Hvordan kan du bruke minst en servo som en del av kjedereaksjonen din? Tegn en skisse.

Fase 3: Bygg kjedereaksjonen og lag programmet til servoen.



Koble opp micro:bit og kretskort med servoen som vist på figuren.

Servoen har tre farger på ledningene der det er viktig at den riktige ledningen havner på den riktige plassen.

Eksempel på program

Det øverste programmet gjør at servoen dreier 180° i den ene retningen.

Det nederste programmet gjør at servoen dreier tilbake til utgangspunktet.

Kan dere bruke noe av dette til deres kjedereaksjon?



Fase 4: Prøv ut kjedereaksjonen.

Fase 5: Kan dere gjøre den bedre på noen måte? Virker den som planlagt? Må noe justeres?

Fase 6: Lag den ferdige kjedereaksjonen.

Fase 7: Lag en filmsnutt av kjedereaksjonen, og skriv eller lag et lydklipp med forklaring på hvorfor dere valgte å lage kjedereaksjonen akkurat slik du gjorde.

Oppgave

Følg kula hele veien gjennom løypa og beskriv hvilke typer energi den har, og hvilke typer energi den mottar og mister underveis i løypa.

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 3 – Modellering med Geogebra

Lineær modellering

I dette kapittelet skal vi lære litt om lineær regresjon. For å forstå hva det er, skal vi se på et eksempel.

Når Line kommer inn i klasserommet en morgen, er det veldig kaldt. Varmeanlegget har vært avslått hele natten, men heldigvis har vaktmesteren skrudd det på igjen. Line har et termometer liggende på pulten sin og ønsker å finne ut hvor fort klasserommet varmes opp igjen. Hun måler temperaturen hver halvtime, åpner Geogebra og skriver resultatene inn i regnearket. Resultatene ser du nedenfor.

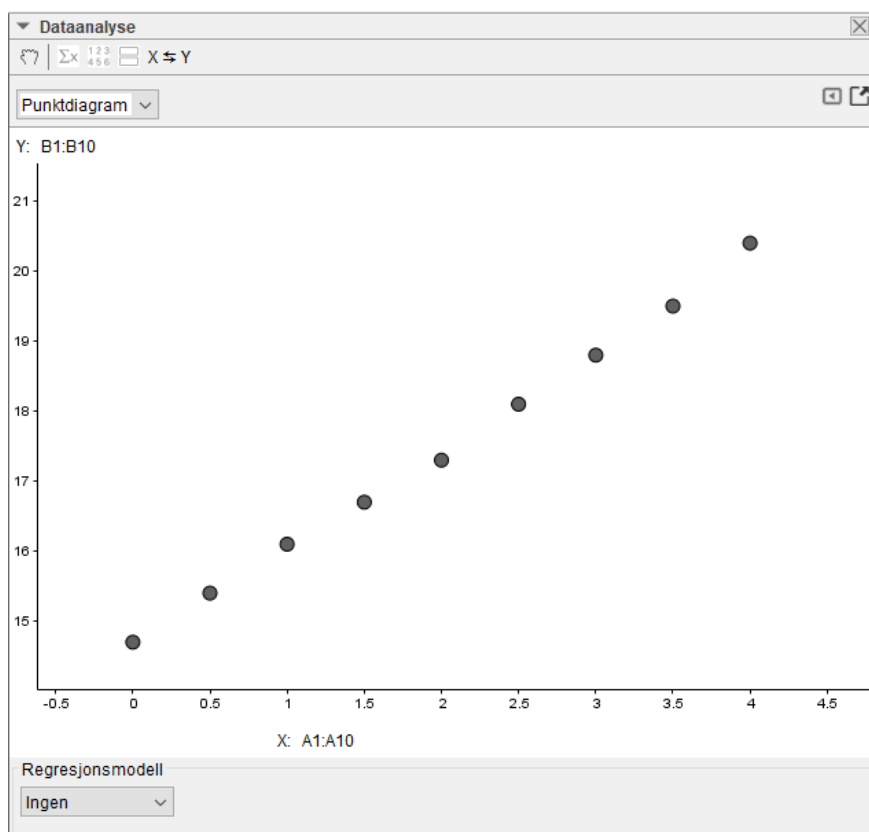
Timer	Grader
0	14.7
0.5	15.4
1	16.1
1.5	16.7
2	17.3
2.5	18.1
3	18.8
3.5	19.5
4	20.4

For å få bedre oversikt, ønsker Line å lage en graf av disse tallene. Det er det som kalles regresjon og vi skal nå se hvordan Line gjør det.

Først markerer hun tallene, og så trykker hun på dette symbolet:

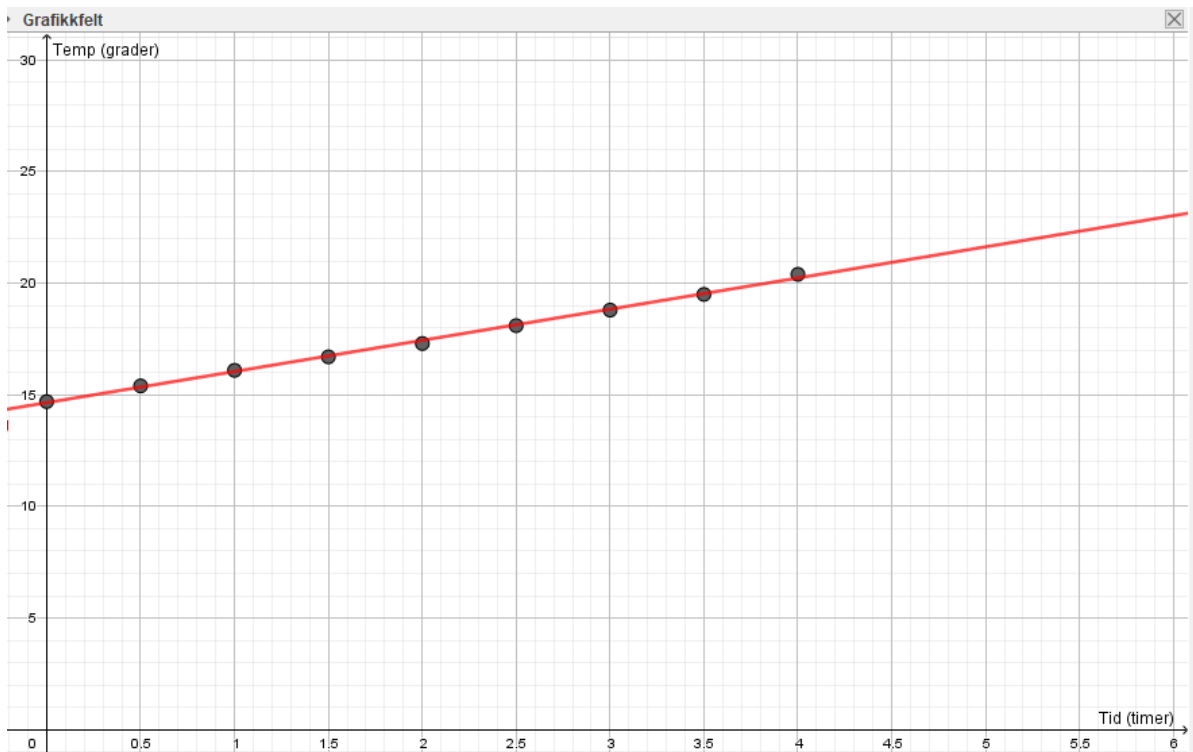


Deretter velger hun «regresjonsanalyse» og «analyser». Hun får da opp tallene i tabellen som punkter. Vi husker at det var 14,7 grader da hun startet å måle. Det tilsvarer punktet nederst til venstre. Etter en halvtime var det 15,4 grader. Det tilsvarer det neste punktet, og slik fortsetter det.



Line ser at punktene ligger omtrent langs en rett linje, og hun ønsker å få PC-en til å velge den linja som passer best med punktene. Det gjør hun ved å velge «regresjonsmodell» nederst i bildet. Hun velger «lineær» som betyr at grafen skal bli en rett linje. Hvis punktene ikke hadde ligget langs en linje, kunne hun valgt noe annet, og det skal vi se på senere.

Til slutt høyreklikker hun i bildet og velger «kopier til grafikkfeltet» og dermed er grafen ferdig.



Funksjonsuttrykket til denne grafen er $y = 1,4x + 14,65$. Hva betyr disse tallene?

Det tallet som står foran x-en, kalles et stigningstall. Jo større dette tallet er, jo brattere er grafen. Hvis tallet er negativt, synker grafen. Her er det 1,4. Det betyr at det blir 1,4 grader varmere for hver time som går.

Det andre tallet kalles konstantledd. Her er det 14,65. Det betyr at grafen starter på 14,65 grader, altså at det ifølge grafen var så mange grader da forsøket startet.

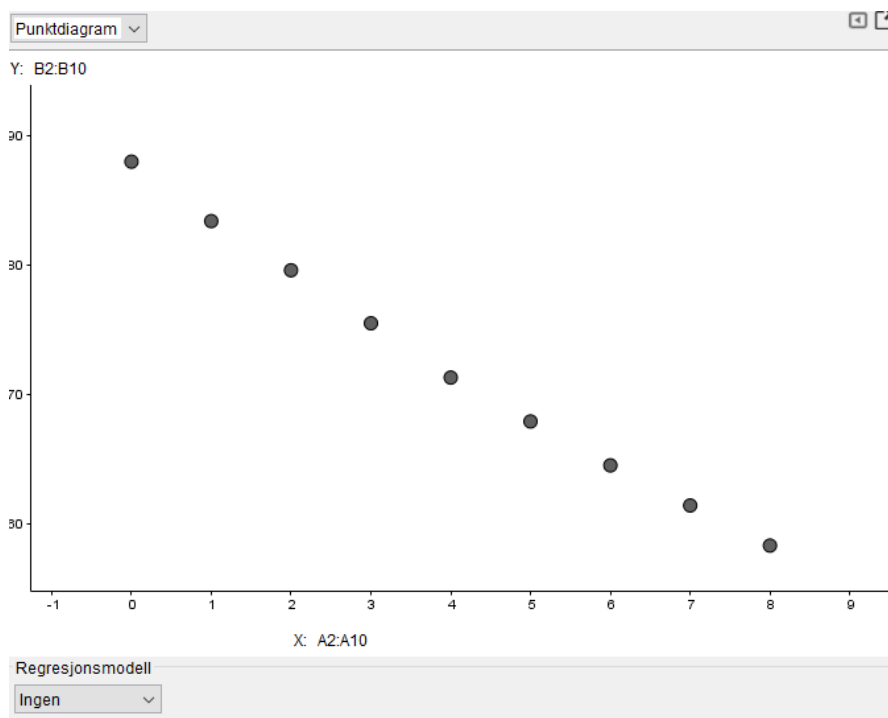
Og dermed har Line funnet svaret. Klasserommet varmes opp med 1,4 grader hver time.

Ikke-lineær modellering

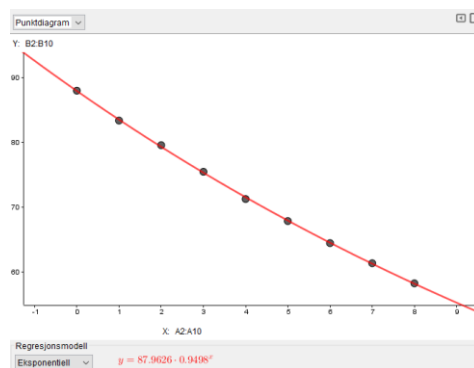
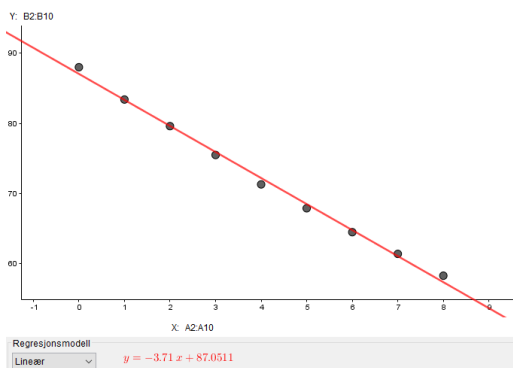
Vi skal nå se hvordan vi kan bruke Geogebra til å lage grafer når punktene ikke ligger på en rett linje.

Markus liker å ha med kakao på tur og han har fått en ny termos som han håper skal holde godt på varmen. Han har lyst til å teste den ut hjemme, og fyller den med varmt vann en morgen. Så måler han temperaturen en gang i timen. Han legger inn punktene på samme måte som Line (se eksempelet ovenfor).

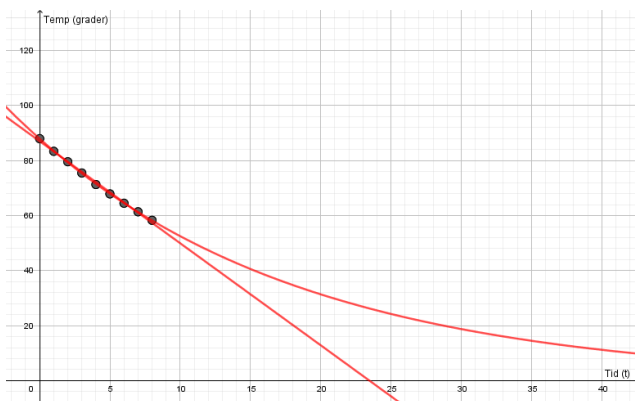
Timer	Temp
0	88
1	83.4
2	79.6
3	75.5
4	71.3
5	67.9
6	64.5
7	61.4
8	58.3



Vi ser at punktene til Markus kanskje ikke ligger helt langs en rett linje slik punktene til Line gjorde. Markus prøver ut forskjellige regresjonsmodeller. Da får han følgende bilder.



Disse grafene ser kanskje litt like ut, men vi ser klart forskjell hvis vi zoomer litt ut.



Hvordan kan vi finne ut av hvilken av disse som passer best med tallene til Markus? For å finne ut av det, må vi se på et tall som kalles korrelasjonskoeffisient. Det finner vi ved å trykke på «vis statistikk»-knappen.



Da får vi tabeller som disse.
Det er det nest nederste tallet (R^2) vi er interessert i.

For den rette linja får vi 0.996.
For den krumme linja får vi 0.9998.

Jo nærmere dette tallet er 1, jo bedre passer grafen. Dette betyr at det er den krumme grafen (den eksponentielle) som passer best.

Nå har Markus fått en fin graf som viser temperaturendringen i termosene over tid.

Lineær graf

Statistikk	
GjennomsnittX	4
GjennomsnittY	72.2111
Sx	2.7386
Sy	10.1807
r	-0.998
p	-1
Sxx	60
Syy	829.1689
Sxy	-222.6
R^2	0.996
SSE	3.3229

Krum graf (eksponentiell)

Statistikk	
GjennomsnittX	4
GjennomsnittY	72.2111
Sx	2.7386
Sy	10.1807
r	-0.998
p	-1
Sxx	60
Syy	829.1689
Sxy	-222.6
R^2	0.9998
SSE	0.2014

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 4

Innhold

Kapittel 4 – Lineær modellering i naturfag og matematikk (8. trinn)

- Variabler i Scratch og MakeCode.....3
- Opplegg 5 – Massetetthet og periodesystemet.....4
 - Lag en volummåler.....5
- Opplegg 6 – Friksjon og omforming av energi.....6
 - Lag et skråplan der vinkelen kan justeres.....7
- Opplegg 7 – Solenergi.....8
 - Lag en effektiv solfanger.....9
- Opplegg 8 – Formler, likninger og fart.....10
 - Lag det raskeste kjøretøyet.....11

Variabler i Scratch og MakeCode

En variabel er som en liten skuff som kan inneholde en verdi. For å kunne utføre regnestykker, trenger vi skuffer til å ha tallene våre i. Hver gang du putter et nytt tall i skuffen, forsvinner det gamle, for skuffene er små og har bare plass til det nyeste tallet.

Det finnes fire typer variabler:

Heltall – Tall uten desimaler

Desimaltall – Tall med desimaler

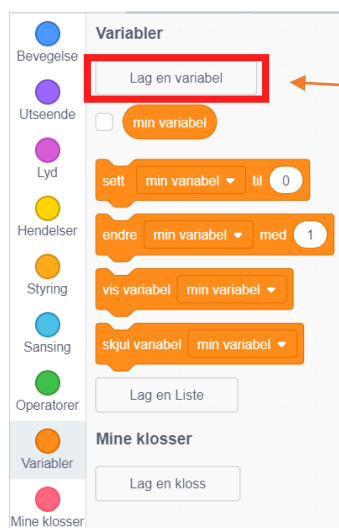
Tekst/strenger – Ord eller bokstaver

Boolske verdier – Sant eller usant



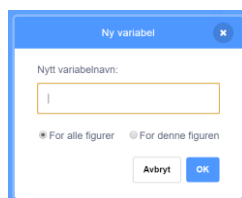
Variabler er nyttige når man skal kunne lage ett program som kan regne ut regnestykker med forskjellige tall, uten å endre hele programmet. I tillegg kan variabler brukes for å holde styr på antall runder i en løkke med en tellevariabel. Se Pythonheftet for mer om variabler.

Variabler i Scratch



For å kunne lage variabler i Scratch, må du finne menyen for variabler – den er mørk oransje. Deretter må du trykke på den øverste knappen der det står «lag en variabel». Det er markert med rødt på figuren her.

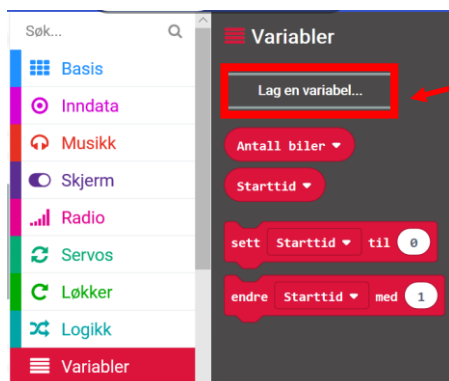
Da dukker dette vinduet opp og du lager et navn på variabelen din.



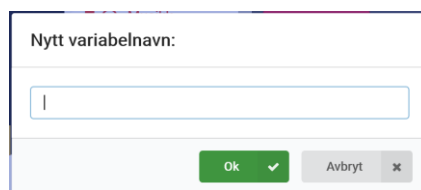
Dette gjentar du til du har mange nok variabler.

Variabler i MakeCode

For å kunne lage variabler i MakeCode, må du finne menyen for variabler – den er rød. Deretter må du trykke på den øverste knappen der det står lag en variabel. Det er markert med rødt på figuren.



Da dukker dette vinduet opp og du lager et navn på variabelen din.



Dette gjentar du til du har mange nok variabler.

Opplegg 5 - Massetetthet og periodesystemet

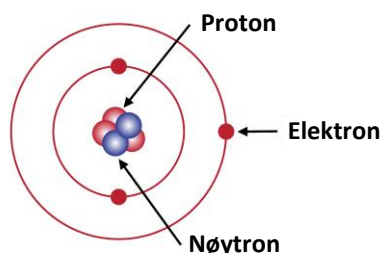
Hva er masse?

Masse er det vi måler når vi veier noe. Vi kaller det ofte tyngde, men tyngden er egentlig den krafta som jordkloden drar oss ned mot bakken med (tyngdekraft). Masse måler vi i kilogram.

Hva er massetetthet?

Den sier noe om hvor stor tetthet et stoff har. Hvis du holder en liten bit av et stoff med stor massetetthet, så vil det være tungt å holde. Enheten for massetetthet er enten kg/m^3 eller g/cm^3 .

$$\text{massetetthet} = \frac{\text{masse}}{\text{volum}}$$



Alle atomer er bygd opp av protoner, nøytroner og elektroner. Atomkjernen består av protoner og nøytroner. Protonene har en positiv ladning, men nøytronene har ingen ladning, de er nøytrale. Elektronene går i bane rundt atomkjernen og de har en negativ ladning. I et nøytralt atom, er det alltid like mange elektroner og protoner.

Visste du at?

En teskje med nøytronstjerne veier omtrent 500 millioner tonn!

Atommassene til grunnstoffene er i hovedsak bestemt av antall protoner og nøytroner i kjernen, siden elektronene har så liten masse i forhold. Atommassene øker når atomnummeret øker, for det sier hvor mange protoner atomet har i kjernen. For de letteste grunnstoffene har atomet omtrent like mange nøytroner som protoner, men for de tyngre grunnstoffene er det flere nøytroner enn protoner.

Atommassene bestemmer svært lite av massetettheten til grunnstoffene. Det er ikke slik at massetettheten automatisk blir høyere for grunnstoffene med høyere atomnummer. En del av grunnen er at for de fleste grunnstoffene, er atomene bundet til hverandre slik at det er molekyler av grunnstoffet. Måten disse er bundet sammen på, er med på å bestemme hvor stor plass molekylene tar. Altså hvor stort volum de har.

Innenfor hver kolonne (gruppe) i periodesystemet er det, likevel, slik at grunnstoffene med høyest atomnummer har størst massetetthet. Grunnen er at disse grunnstoffene har lignende kjemiske egenskaper siden de har like mange valenselektroner (elektroner i ytterste skall), som er det viktigste for å bestemme bindingstype og kjemiske egenskaper til stoffet.

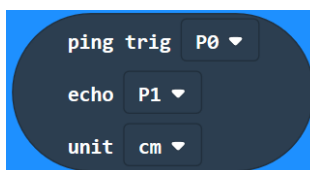
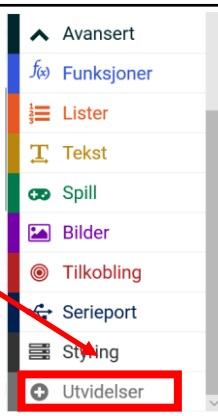
Diskuter

Hvilke forskjellige typer bindinger vet du om?

Ultralydsensor (sonar)

1. For å kunne bruke en ultralydsensor (sonar), så må du først trykke på avansert i menyen. Så må du trykke på utvidelser, nederst i menyen. Det er markert med rødt på figuren her.

2. Trykk så på ruta der det står sonar.



Da får dere tilgang til denne programmeringsklossen. Den viser avstanden fra sonaren målt i cm eller mikrosekunder.

Ultralydsensoren måler ikke avstand direkte. Den måler tiden det tar for ultralyd å gå fra sensoren, bort til en overflate, og så tilbake igjen. Siden sensoren vet at lydfarten er ca 340 m/s i luft, så kan den regne tiden om til avstand.

Dessverre oppgir sensoren avstanden i hele cm, altså ikke særlig nøyaktig. Hva kan vi gjøre med det? Kan det være mer nøyaktig å måle tiden, og heller regne om selv? (HINT: $s = v \cdot t$)

Lag en volummåler

- finn massetettheten til et stoff

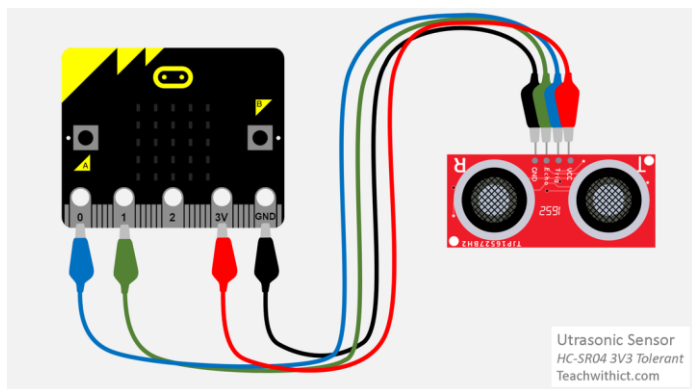
Oppgave

Lag et produkt som måler volumet til et stoff ved å bruke en ultralydsensor (sonar) til å måle hvor mye vannstanden i et begerglass øker når du putter i en bit av stoffet. Bruk massen av biten, og volumet, for å beregne massetettheten og finne ut av hvilket stoff det er.

Fase 1: Hvilke deler består en volummåler av? Hvordan ser disse delene ut? Hva må bygges sammen av disse delene?

Fase 2: Ha en idémyldring for deg selv. Hvordan vil du at din volummåler skal se ut? Tegn gjerne en skisse før du diskuterer med de andre.

Fase 3: Tid for å lage volummåleren og programmere micro:biten. Koble opp micro:bit som vist på figuren.



Nødvendig utstyr til volummåleren

Kalibrering av volummåleren (enkel regresjon)

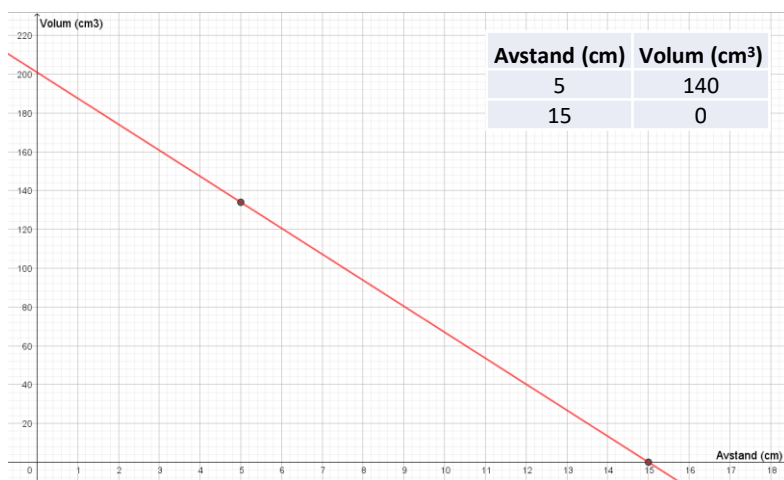
Vi skal finne massetettheten til diverse stoffer. Da må vi måle hvor stor masse og volum de har. Men det identifikatoren vår måler er avstand, og ikke volum. Heldigvis er det en sammenheng mellom dem!

Vi gjør først en måling av avstanden til overflaten på vannet uten noe oppi vannet, og noterer avstanden (volumet er 0 siden vi ikke har noe oppi vannet). Så legger vi oppi en halvstor gjenstand og leser av både avstanden til vannet og volumet på skalaen på siden av begerglasset. Da har vi fått to punkter som vi kan tegne inn i et koordinatsystem.

Dette kan du enten gjøre for hånd eller digitalt. Når du har tegnet de to punktene, trekker du en rett linje mellom dem. Da har du fått en graf du kan bruke til å lese av volumet for den avstanden som vises på micro:biten.

Dersom du vil lage en mer nøyaktig graf, gjør du flere målinger og tegner inn punktene i samme koordinatsystem. Da vil det ikke passe med en rett linje som treffer alle punktene dine, men du tegner opp den rette linja som treffer best mulig med alle punktene dine.

Da har du gjort en regresjon. Det vil si at du har laget en matematisk modell! Gratulerer! Det er det som er det største ønsket til mange forskere innen realfag. Det finnes enklere måter å gjøre det digitalt på, slik at du slipper å tegne inn alle punktene og linja for hånd. Det står det mer om i modellerings-kapitlet.



Diskuter

Hvorfor vil det ikke være mulig å finne massetettheten til salt på denne måten?

Gjør fase 4 – 7.

Opplegg 6 - Friksjon og omforming av energi

Friksjon hindrer overflater i å bevege seg i forhold til hverandre. Den virker slik at den alltid bremses en bevegelse. Hadde det ikke vært for friksjon og luftmotstand (som er en slags friksjon), hadde alle bevegelser fortsatt uten å stoppe opp!

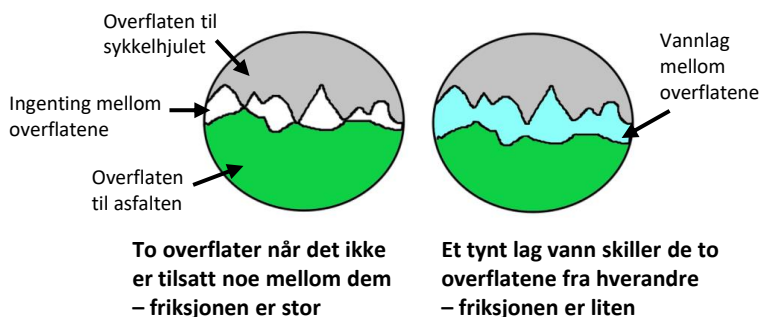
Tenk at du aker ned en bakke med akebrett. Ved enden av bakken er det en diger slette som er dekket med et islag med vann oppå. Hva skjer med farten din på sletta? Hva om det i stedet hadde vært snøslaps, eller kanskje gress? Forskjellen på is med vann, snøslaps og gress er hvor stor friksjonen blir. Jo glattere overflaten er, desto mindre friksjon har vi, og akebrettet fortsetter veldig langt før det stopper. Vi har uansett luftmotstanden som bremses bevegelsen, så den fortsetter ikke uendelig lenge.

Diskuter

Vet du om et sted uten luft, og dermed uten luftmotstand? Hvilke konsekvenser får det for bevegelse?

Det er hvor ujevne overflatene er, som bestemmer hvor stor friksjonen er. Vi kan tilsette smøremidler mellom ujevne overflater, slik som i bilmotorer og andre steder der overflater skal bevege seg i forhold til hverandre. Hva skjer om friksjonen er stor inni en bilmotor?

I andre tilfeller ønsker vi ikke å redusere friksjonen. Hva med når du sykler, vet du hva vannplaning er? Det er når det legger seg et lag med vann på asfalten, og det virker på samme måte som et smøremiddel. Skummelt!



Omforming av energi

Friksjon gjør at farten minker, og da blir bevegelsesenergien også mindre. Hvor blir resten av energien av? Den kan jo ikke bare forsvinne.

Energien går stort sett over fra bevegelsesenergi til varme- og lydenergi. Den kan også gå med til å deformere overflatene.

På bildet ser du en ball som slippes fra en høyde, og for hver gang den spretter, når den kortere opp. Det er friksjonen mellom ball og gulv som omformer (stjeler) noe av bevegelsesenergien til ballen, slik at den ikke når like høyt neste gang.

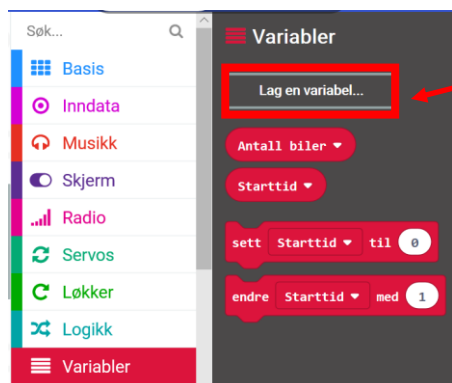


Variabler i MakeCode

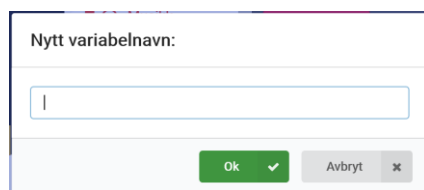
En variabel er som en liten skuff som kan inneholde en verdi. For å kunne utføre regnestykker, trenger vi skuffer til å ha tallene våre i. Hver gang du putter et nytt tall i skuffen, forsvinner det gamle, for skuffene er små og har bare plass til det nyeste tallet. Disse skuffene må vi først lage.



For å kunne lage variabler i MakeCode, så må du finne menyen for variabler – den er rød. Også må du trykke på den øverste knappen der det står lag en variabel. Det er markert med rødt på figuren.



Da dukker dette vinduet opp, og du lager et navn på variabelen din.



Dette gjentar du til du har mange nok variabler.

Lag et skråplan der vinkelen kan justeres

- og en stoppeklokke med micro:bit

Oppgave

Lag et justerbart skråplan som dere skal bruke til å la en kule trille ned med forskjellige vinkler. Mål tiden det tar for kula å trille en avmerket avstand på skråplanet med micro:biten. Lag en lineær modell for hvordan trilletiden avhenger av vinkelen til skråplanet.

Fase 1: Hva er et skråplan? Hva kan det bygges av? Hvordan få det regulerbart?

Fase 2: Det er viktig at dere er åpne for alle slags ideer og ikke er kritiske, da kan nyttige forslag bli avfeid for tidlig.

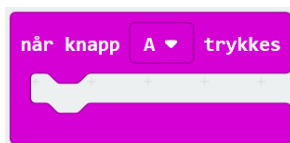
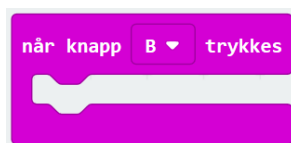
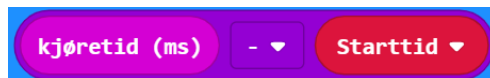
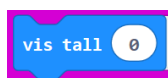
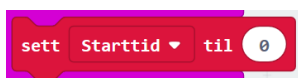
1. Tenk selv først og tegn gjerne skisser.
2. Forklar ideen din for de andre på gruppa.
3. Hele gruppa diskuterer de ulike ideene, og lager en felles hypotese for resultatet av målingene.

Fase 3: Tid for å lage skråplanet og programmere micro:biten. Under er det et puslespill-hint til stoppeklokke-programmet.

Kjøretid-klossen

Denne klossen måler hvor lang tid det har gått siden programmet begynte å kjøre. Den begynner å telle med en gang programmet er overført til micro:biten. Den finnes under kategorien inndata, der det står mer... Den måler tiden i millisekunder. Hva må du gjøre for å få tiden i sekunder?

kjøretid (ms)



Fase 4: Prøv ut skråplanet og stoppeklokka.

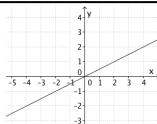
Fase 5: Virker det som planlagt? Må noe justeres? Kan du gjøre skråplanet eller stoppeklokka bedre på noen måte?

Fase 6: Lag det ferdige skråplanet og stoppeklokka.

Fase 7: Gjennomfør målinger av trilletid for minst 5 forskjellige vinkler. Skriv ned målingene og ta bilder.



Regresjonsoppgave – lineær modell



1. Plott alle de målte verdiene i Geogebra.
2. Finn en matematisk modell ved å foreta en regresjon for de målte dataene. Se modelleringskapitlet for fremgangsmåte.
3. Hvordan tror du modellen ville blitt om dere hadde gjort flere målinger?
 - Prøv!
 - Passet de nye målingene inn i modellen deres?
 - Var de nye målingene innimellom de gamle verdiene, eller utenfor? Tror du det har noe å si for resultatet?

Diskuter

Stemte hypotesen deres? Hvorfor/hvorfor ikke?

Oppgave

Modellen deres er en lineær funksjon som kan skrives på denne formen

$$f(x) = a \cdot x + b$$

Dersom dere setter deres funksjon lik en annen gruppes funksjon har dere laget en likning! Løs denne likningen, og diskuter hva dere har regnet ut. 42

Opplegg 7 - Solenergi

- Hvordan kan vi varme vann ved hjelp av solenergi?



Energiproduksjon fra sollys

Energien fra sollyset er fornybar energi siden vi ikke kan bruke den opp. Sollyset som treffer jordkloden tilsvarer en mengde energi som er 15 000 ganger høyere enn energiforbruket til alle menneskene på jordkloden. Dersom vi hadde klart å utnytte alle energien fra sola, hadde vi ikke trengt annen energiproduksjon.

Det er hovedsakelig to måter å produsere energi fra sollys. Vi kan bruke solceller eller solfangere. I en solcelle dannes det

elektrisitet siden sollyset kan slå løs elektroner. Disse elektronene beveger seg i en krets på grunn av måten solcellene er bygget opp, og vi får dannet elektrisitet.

Det finnes to typer solfangere. Begge virker slik at de varmer opp vann eller en annen væske. Forskjellen på de to typene er hvor høy temperatur de varmer væsken opp til, og hva de gjør med væsken etter den er oppvarmet.

1. **Solfanger til elektrisitetsproduksjon** – denne typen består av mange speil som reflekterer sollyset slik at det treffer en liten beholder med væske. Da vil temperaturen til denne væska øke kraftig og væska fordampes. Det vil si at den blir til gass. Da kan gassen drive en turbin som omformer bevegelsesenergien til gassen til elektrisitet. Dette er typisk større energikraftverk som produserer elektrisitet nok til mange tusen husstander, slik som på bildet over.
2. **Solfanger til oppvarming av vann** – denne typen er langt enklere, og noe privatpersoner kan ha hjemme. De bruker energi fra sollyset til å varme opp vann til opp mot kokepunktet. Mange bruker slike solfangere på hytta eller campingtur for å varme opp dusjvann. De består av en svart pose med vann oppi, siden svart farge absorberer mest mulig av strålingen fra sollyset, og en plastslange med et lite dusjhode. Her får vannet som regel ikke veldig høy temperatur, men det kan være lurt å sjekke litt forsiktig med en lillefinger før man dusjer hele seg.

Lignende solfangere kan også brukes av mennesker som bor steder uten tilgang til elektrisitet for å koke vannet sitt slik at bakterier og parasitter dør. Da er det mindre fare for å bli syke. I tillegg kan de få laget varm mat, selv om de ikke har tilgang på ved. Disse solfangerne består gjerne av papp eller tre, noe reflekterende, noe absorberende, noe som isolerer, og en beholder til vannet.



Materialene som en solfanger består av har alle en funksjon.



Papp - byggemateriale



Plastfolie - isolerer



Aluminiumsfolie - reflekterer sollys



Svart maling – absorberer sollys

Snakk om

All energi på jordkloden stammer til sist fra solenergi, både fornybare energikilder og ikke-fornybare energikilder. For eksempel vindkraft, vannkraft, olje, gass og kull, til og med energien i maten vi spiser!

1. Hvordan kan det stemme? Forklar hvordan det kan være slik.
2. Kan du komme på noen unntak? Er det noen energikilder som ikke har sitt opphav i solenergi?



Lage en effektiv solfanger

Oppgave

Lag en solfanger. Mål hvor mye temperaturen stiger for hvert minutt den står i sola. Lag en lineær modell for hvordan temperaturen avhenger av tiden.

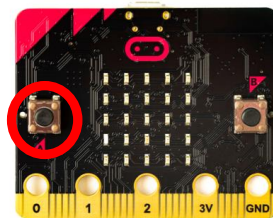
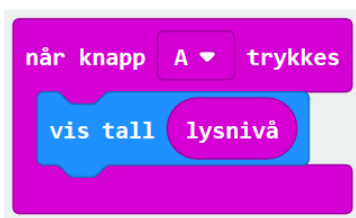
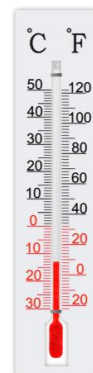
Fase 1: Undersøk litt mer om solfangere, dersom dere ønsker det.

Fase 2: Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan nyttige forslag bli avfeid for tidlig.

- Tenk selv først, og tegn gjerne skisser.
- Forklar ideen din for de andre på gruppa.
- Hele gruppa diskuterer de ulike ideene, og lager en felles hypotese for bygging.

Fase 3: Gjennomfør planen deres for å lage solfangeren og lag programmet for å måle temperaturen med micro:bit.

Lag et program som måler temperaturen når du trykker på den ene knappen. Under vises et programeksempel som måler lysstyrken når man trykker på knapp A, og skriver verdien til LED-lysene på framsiden av micro:biten (skjermen). Hva må du endre på?



Knapp A finner du her på micro:biten

Fase 4: Test hvor mye temperaturen øker.

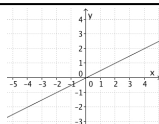
Fase 5: Sammenlign resultatene med andre i klassen.

Fikk noen andre større temperaturøkning? Hvorfor gjorde de det, tror du? Kan din gruppe gjøre noe av det samme?

Fase 6: Gå tilbake til de andre fasene for å gjøre de planlagte forbedringene.

Fase 7: Gjennomfør de siste målingene, disse skal dere bruke til å plote en graf med.

Regresjonsoppgave – lineær modell



1. Plott alle de målte verdiene i Geogebra.
2. Finn en matematisk modell ved å foreta en regresjon for de målte dataene. Se modelleringskapitlet for fremgangsmåte.
3. Hvordan ser funksjonen ut?
4. Passer den bra med datapunktene deres?
5. Tror dere modellen vil passe bra dersom dere lar solfangeren stå ute i sola en hel dag?

Ekstraoppgaver

1. Hva kan være mulige feilkilder for måling av temperaturen?
2. Forklar viktigheten av å teste-evaluere-forbedre.
3. Kan dere finne ut av hva faseoverganger er?
4. Koble resultatene deres mot faseoverganger, kanskje går en del av energien til at vannet fordampes. Hvordan kan man forhindre dette?

Oppgave

Modellen deres er en lineær funksjon som kan skrives på denne formen

$$f(x) = a \cdot x + b$$

Dersom dere setter deres funksjon lik en annen gruppes funksjon har dere laget en likning! Løs denne likningen, og diskuter hva dere har regnet ut.

Opplegg 8 - Formler, likninger og fart

Hva er fart?

Farten forteller oss hvor raskt vi beveger oss fra et sted til et annet. Når en gjenstand beveger seg, så er gjennomsnittsfarten lik strekningen dividert med tiden det tok. Dette kan vi skrive med formelen $v = \frac{s}{t}$ der v er farten, s er strekningen og t er tiden.



Snakk om

Enheten til fart er ofte km/t eller m/s . Vet du når de forskjellige enhetene brukes?

Vet du om andre enheter for fart og når de brukes?

Likninger

En likning består av en ukjent variabel (ofte bruker vi x), et likhetstegn og minst ett tall. Det som er hovedpoenget med likninger er at de inneholder et likhetstegn som forteller oss at det på venstre side er likt som det på høyre side av likhetstegnet.

Dette er like stort $\rightarrow 3x - 2 = x + 4 \leftarrow$ som dette

Siden vi vet at det som står på begge sidene av likhetstegnet er like stort, så må vi alltid gjøre det samme på begge sider av likhetstegnet. Dersom vi legger til et tall, så MÅ vi gjøre det på begge sidene. Det samme gjelder om vi trekker fra et tall, multipliserer eller dividerer. Når vi har to ting som er like, må vi alltid gjøre det samme med begge, for at de fortsatt skal være like. Er ikke det logisk?

Vi prøver å løse eksempelet vårt. Det finnes flere mulige rekkefølger for hva vi gjør, og dere kan velge helt fritt blant dem, så lenge dere ALLTID GJØR DET SAMME PÅ BEGGE SIDER.

Samle alle ledd med x på den venstre siden, da må vi trekke fra alle x -ene som er på høyre side:

$$3x - 2 - x = x + 4 - x$$

Også trekker vi sammen uttrykkene på hver side og får:

$$2x - 2 = 4$$

Det neste steget er å samle alle tallene på høyre side, da må vi finne ut hva vi må legge til eller trekke fra for å kvitte oss med tallene på venstre side:

$$2x - 2 + 2 = 4 + 2$$

Også trekker vi sammen uttrykkene på hver side og får:

$$2x = 6$$

Da er vi nesten i mål, men vi må bare kvitte oss med tallet foran x . Det står jo egentlig en gangetegn mellom tallet og x , så dersom vi deler på samme tallet, så kan vi stryke tallet. Dette fordi det blir som å multiplisere med en brøk som er lik 1, slik som $\frac{2}{2} = \frac{3}{3} = \frac{6}{6} = \frac{1000}{1000} = 1$.

Da får vi:

$$\frac{2x}{2} = \frac{6}{2}$$

Når vi til slutt forkorter, så står vi igjen med:

$$x = 3$$

Og vi har funnet svaret vårt! Så lenge vi gjør det samme på begge sidene av likhetstegnet, så trenger vi egentlig ikke huske noe mer for å løse likninger.

Formelregning

En formel er egentlig veldig lik en likning, og vi kan bruke akkurat de samme reglene for å omforme den. For $v = \frac{s}{t}$ som vi startet med, så kan vi omforme den til $s = v \cdot t$. Klarer du å se hva man må gjøre?

Lag det raskeste kjøretøyet

- med kontinuerlig servo

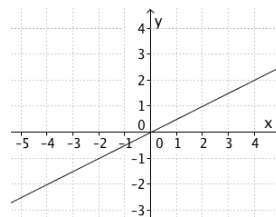
Oppgave

Lag et kjøretøy som bruker en kontinuerlig servo. Mål hvor lang tid det bruker på minst fem forskjellige strekninger. Lag en lineær modell for hvordan tiden avhenger av strekningen.

Fase 1: Undersøk gjerne litt for å få inspirasjon til kjøretøyet deres.

Fase 2: Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan nyttige forslag bli avfeid for tidlig.

- Tenk selv først, og tegn gjerne skisser.
- Forklar ideen din for de andre på gruppa.
- Hele gruppa diskuterer de ulike ideene, og lager en felles hypotese for bygging.



Fase 3: Gjennomfør planen deres for å lage kjøretøyet, og lag programmet for å styre servoen.

Ekstraoppgave

Lag en stoppeklokke for micro:bit for å måle tiden på strekningene.



Fase 4: Test hvor raskt kjøretøyet er.

Fase 5: Sammenlign resultatene med andre i klassen. Fikk noen andre større fart? Hvorfor tror du deres kjøretøy fikk større fart? Kan dere gjøre noe av det samme?

Fase 6: Gå tilbake til de andre fasene for å gjøre de planlagte forbedringene.

Fase 7: Gjennomfør de siste målingene for strekning og tid, disse skal dere bruke til å plote en graf med.



Regresjonsoppgave – lineær modell

1. Plott alle de målte verdiene i Geogebra. Tiden tilsvarer x-verdiene og strekningen tilsvarer y-verdiene.
2. Finn en matematisk modell ved å foreta en lineær regresjon for de målte dataene. Se modelleringskapitlet for fremgangsmåte.
3. Passer den bra med datapunktene deres?
4. For denne lineære modellen, vil stigningstallet være gjennomsnittsfarten til kjøretøyet. Det kjøretøyet med størst stigningstall, har størst fart. Hvordan kan dere se på grafen hvilket kjøretøy som har størst fart?
5. Tror dere modellen vil passe bra dersom dere lar kjøretøyet kjøre i 12 timer?

Oppgave

Modellen deres er en lineær funksjon som kan skrives på denne formen

$$f(x) = a \cdot x + b$$

Dersom dere setter deres funksjon lik en annen gruppes funksjon har dere laget en likning! Løs denne likningen, og diskuter hva dere har regnet ut.

Ekstraoppgave

Når dere har funksjonen for sammenhengen mellom tiden og strekningen til kjøretøyet, finn ut hva strekningen måtte vært for at kjøretøyet skulle kjørt i 5 timer.

Tror dere at svaret er realistisk?

Kan dere lage en annen ligning basert på kjøretøyet deres?

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 5

Innhold

Kapittel 5 – Musikkopplegg

- Opplegg 9 – Rytme.....3
 - Lag et programmerbart instrument.....4
- Opplegg 10 – Lyd og lydbølger.....5
 - Lag et musikk-kort.....6
- Opplegg 11 – Toner og frekvens.....7
 - Lag en ringeklokke.....8

Opplegg 9 - Rytme

Rytme er satt sammen av flere andre begreper, slik som betoning, puls og taktart. Hvordan man opplever tempoet varierer også med hvilke slag som blir markert med trommer eller et annet rytmeinstrument.

Det finnes mange ulike taktarter, og noe musikk har til og med flere taktarter i samme stykke! Taktarter skrives som brøker, og i vesten bruker vi i hovedsak disse taktartene: $\frac{2}{4}$, $\frac{3}{4}$, $\frac{4}{4}$ og $\frac{6}{8}$, men dette avhenger av kultur.

Marsjene til skolekorpset på 17.mai går typisk i $\frac{2}{4}$ -takt eller $\frac{4}{4}$ -takt, for at man skal kunne marsjere til dem.

Vals går i $\frac{3}{4}$ -takt, og det er ganske håpløst å danse vals til noe med annen taktart.

Noter og varighet

Notene har forskjellig utseende ut fra hvor lenge man skal holde dem når man spiller eller synger. Under vises en oversikt over varighet til noter og pauser.

	Helnote		Helpause
	Halvnote		Halvpause
	4.delsnote		4.delspause
	8.delsnote		8.delspause
	16.delsnote		16.delspause



Takt

Vi bruker takter for å dele opp musikken, og ut fra hvilken taktart, vil musikken ha mange forskjellige stiler. Taktarten er en brøk, og vi trenger faktisk brøkgregning for å skrive hvilke noter som kan være med i en takt med en bestemt taktart. For $\frac{4}{4}$ -takt, så vil vi ha enten en helnote, to halvnoter, fire 4.delsnoter eller andre kombinasjoner, som til sammen blir $\frac{4}{4}$ (eller en hel). Se eksempler på lovlige/ulovlige kombinasjoner under.

Ulovlig $\neq \frac{3}{4}$

$$\frac{3}{4} \text{ takt: } \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{4} = \frac{7}{8} < \frac{3}{4}$$

$$\frac{3}{4} \text{ takt: } \frac{1}{8} + \frac{1}{2} = \frac{5}{8} < \frac{3}{4}$$

$$\frac{3}{4} \text{ takt: } \frac{1}{16} + \frac{1}{16} + \frac{1}{16} + \frac{1}{2} + \frac{1}{8} = \frac{13}{16} > \frac{3}{4}$$

Lovlig $= \frac{3}{4}$

$$\frac{3}{4} \text{ takt: } \frac{1}{4} + \frac{1}{4} + \frac{1}{4} = \frac{3}{4}$$

$$\frac{3}{4} \text{ takt: } \frac{1}{2} + \frac{1}{8} + \frac{1}{8} = \frac{6}{8} = \frac{3}{4}$$

$$\frac{3}{4} \text{ takt: } \left(\frac{1}{4} + \frac{1}{8} \right) + \frac{1}{8} + \frac{1}{4} = \frac{6}{8} = \frac{3}{4}$$

Betoning

Det at man legger mer trykk på noen noter/toner enn andre og fremhever disse. De betonte tonene virker dermed tyngre enn de andre.

Hva er puls?



Den forteller oss hvor raskt musikken går, eller hvilken hastighet den har. Det kalles også tempo. Vi bruker gjerne et metronomtall for å si noe om tempoet, og det måles i BPM. BPM står for beats per minute, altså taktslag per minutt.

Det går an å bruke en metronom for å holde tempo. Før var disse analoge og hadde et slags urverk som gjorde dem i stand til å lage et visst antall klikkelyder per minutt. Nå er det vanligere med digitale metronomer, der det er elektronikk som gjør at vi får svake tikkelyder.

Det går også an å ha en metronom som ligner på et armbåndsur. Det kan gi en liten vibrasjon i stedet for lyd ved hver puls i musikken.

Lag et programmerbart instrument

Oppgave

Lag et rytmeinstrument som du må bruke minst en servo for å få lyd i.

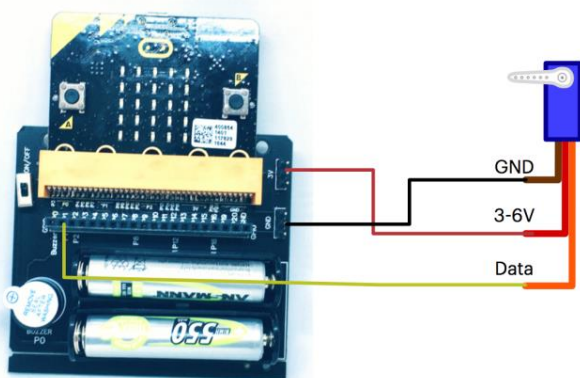


Fase 1: Vet du hva er et rytmeinstrument er? Kan du finne eksempler på forskjellige typer rytmeinstrumenter? Hva er forskjellen på et rytmeinstrument og et strengeinstrument eller et blåseinstrument?

Fase 2: Ha en idémyldring for deg selv. Hvordan vil du at ditt instrument skal se ut? Hvilke materialer vil du bruke? Hvordan kan du få akkurat den type lyd du har mest lyst å lage?

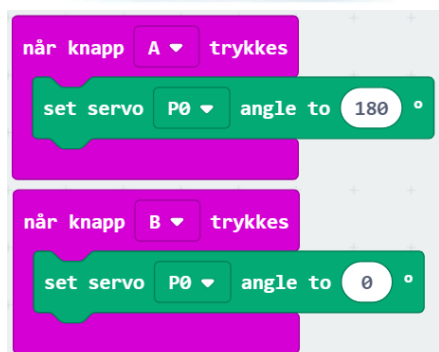
Det kan være lurt å tegne en skisse av instrumentet ditt.

Fase 3: Tid for å lage instrumentet. Bruk gjerne skissen fra fase 2. Så må du lage et program som får rytmeinstrumentet til å lage en rytmisk lyd. Hva slags type servo er den beste å bruke til ditt instrument?



Koble opp micro:bit og kretskort med servoen som vist på figuren.

Servoen har tre farger på ledningene der det er viktig at den riktige ledningen havner på det riktige stedet.



Eksempel på program

Det øverste programmet gjør at servoen dreier 180° i den ene retningen.

Det nederste programmet gjør at servoen dreier tilbake til utgangspunktet.

Kan du bruke noe av dette til ditt instrument?

Fase 4: Nå er det på tide å kombinere instrumentet du har laget, med programmet du har laget.

Fase 5: Virker instrumentet ditt slik du hadde tenkt? Ble lyden fin? Har noen andre i klassen kulere lyd? Hvorfor er den lyden kulere? Eller vil du forandre på rytmen?

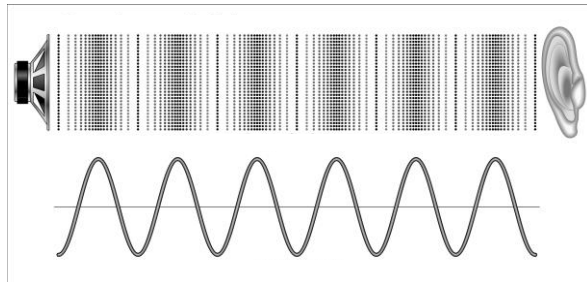
Fase 6: Kan du gå tilbake til en av de andre fasene for å lage et enda bedre instrument? Du kan endre både programmet og selve instrumentet ditt.

Fase 7: Til slutt kan du forklare hvordan du har gjort for å lage instrumentet ditt, og hvorfor du valgte akkurat din måte å lage rytme på. Skriv, tegn eller lag en lydfil med forklaringen.

Ekstraoppgaver

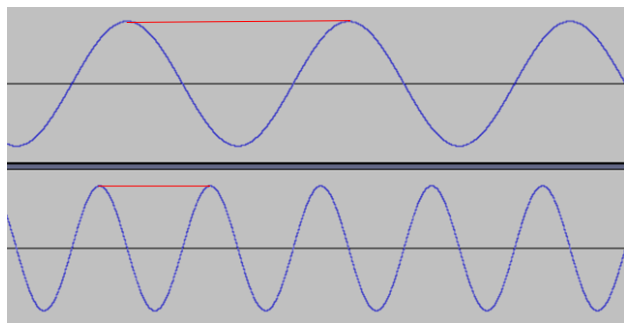
1. Kan du kombinere flere forskjellige måter å lage lyd på og lage et orkester?
2. Gå sammen med noen andre og lag musikk som spiller i takt.

Opplegg 10 - Lyd og lydbølger



Hva er lyd?

Lyd oppstår når luftmolekylene settes i bevegelse, og denne bevegelsen forplanter seg videre. Da vil luftmolekylene bevege seg på en slik måte at det dannes områder med større trykk og områder med mindre trykk. Disse områdene forflytter seg, og fører til at lyden forplanter seg videre. Etter hvert kan de svingende molekylene sette luftmolekylene inni ørene våre i bevegelse, og da vil trommehinna også begynne å svinge på samme måte som skinnet på en tromme. Denne svingningen forplanter seg til de små beina i det indre øret, og det blir sendt signaler til hjernen om disse svingningene. Hjernen oversetter disse signalene til lyd – enten det er fin musikk eller bare støy.

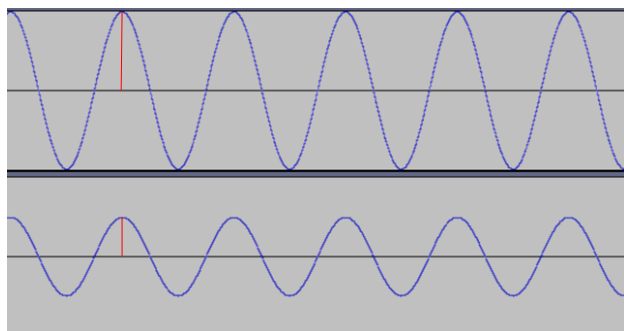


Bølgelengde

Avstanden mellom to bølgetopper. På figuren har den øverste bølgen dobbelt så stor bølgelengde som den nederste. Bølgelengde måles i meter.

Frekvens

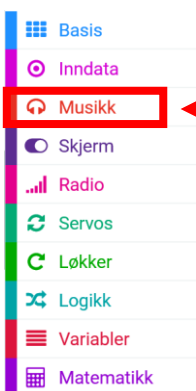
Antall svingninger per sekund. Når frekvensen øker blir tonen lysere. Frekvens måles i hertz (Hz).



Amplitude

Avstanden fra likevektsstillingen (nullpunktet) til toppunktet slik det er merket på figuren. Den øverste bølgen har dobbelt så stor amplitude som den nederste bølgen. Volumet på lyden øker når amplituden øker. Amplituden måles i forskjellige enheter alt etter hvilke bølger vi ser på. For lydbølger blir enheten den samme som for lufttrykk (Pascal eller Pa).

Musikk i MakeCode



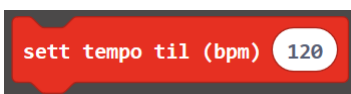
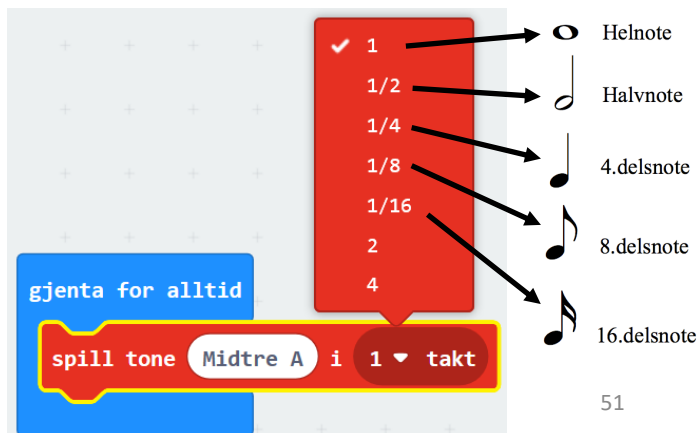
For å kunne bruke musikklossene, må du først trykke på musikk i menyen.

Da får du tilgang til en rekke klosser som du kan bruke for å komponere din egen melodi og spille den av.

Denne klossen spiller av en tone i så lang tid som du velger.



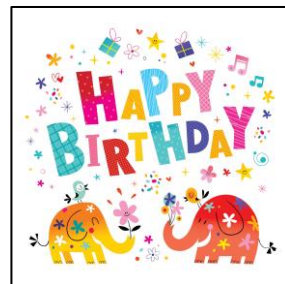
Valgene som kommer opp for hvor lenge tonen skal spille, tilsvarer de forskjellige notetyperne som vist under.



Denne klossen angir tempoet (pulsen) på melodien din.



Denne klossen lager en pause i melodien din.



Lag et musikkort

- som spiller av en egenkomponert melodi

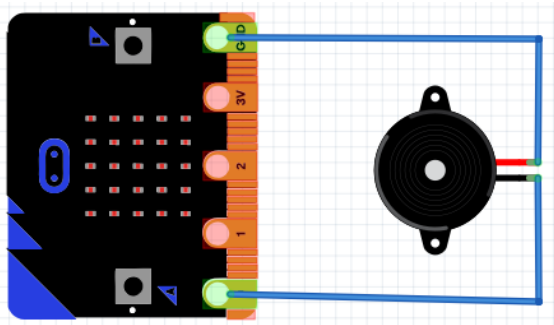
Oppgave

Lag et kort som spiller en egenkomponert melodi når det blir åpnet.

Fase 1: Finn informasjon og inspirasjon til et musikkort. Hva slags kort skal det være? Morsdags- eller farsdagskort? Julekort? Gratulerer-med-bursdagen-til-gullfisken-din-kort?

Fase 2: Ha en idémyldring for deg selv. Hvordan vil du at kortet skal se ut? Tegn gjerne en skisse før du begynner å lage det. Hvilke materialer skal du bruke? Kortet skal spille av en melodi, hvordan skal melodien din være?

Fase 3: Tid for å lage kortet og programmere micro:biten slik at den spiller en melodi når du trykker på knappen.



Koble opp micro:bit som vist på figuren.

Hvordan må vi koble om vi vil få tilkoblede LED-pærer til å lyse?



Dette programmet får høyttaleren til å spille C i én takt.

Hvordan kan du få den til å spille en hel melodi?



Fase 4: Test programmet ditt.

Fase 5: Virker det slik det skal? Sjekk gjerne om det er noen andre i klassen som har spennende ideer de har brukt til kortet sitt. Kan du bruke noe av dette til ditt kort?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få et best mulig kort. Gjør gjerne endringer i micro:bit-programmet ditt.

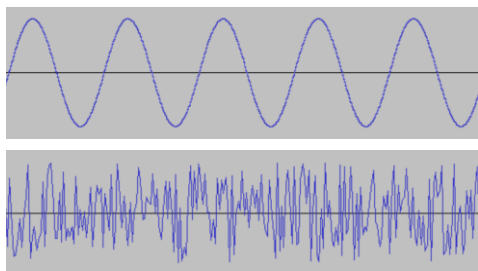
Fase 7: Forklar for en annen i klassen hvordan du gikk fram for å lage kortet ditt.

Ekstraoppgave

Kunne du fått kortet ditt til å spille en kjent melodi?

1. Søk gjerne opp noteark med melodien på internett.
2. Lag programmet til micro:biten.
3. Sjekk om de andre i klassen klarer å gjenkjenne melodien. 52

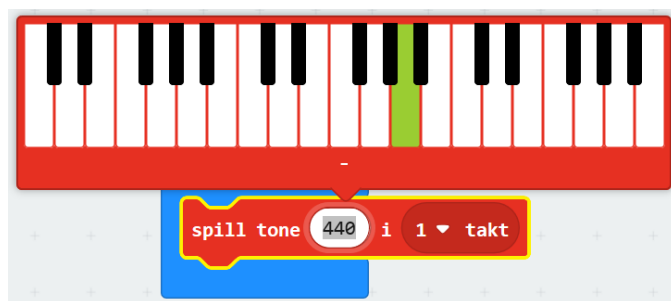
Opplegg 11 - Toner og frekvens



Forskjellen på en ren tone og støy eller annen lyd er at tonen har en fast bølgelengde og frekvens. Da vil lufttrykket variere på samme måte hele tiden. Da kan vi få en graf som viser en fin bølge når vi plotter lufttrykk som en funksjon av tiden.

For støy vil lufttrykket variere på en kaotisk måte, og vi får ingen fin bølgegraf. Dersom vi plotter lufttrykk som en funksjon av tiden, vil den ligne den nederste grafen til venstre.

I MakeCode kan du velge toner utfra navnet på tonene, frekvensen deres, eller fra tangentene på et slags piano. Du kan velge mellom tre forskjellige oktaver (frekvensintervaller) der de har kalt de mørkeste tonene for lav, og de lyseste tonene for høy. Innen musikkteori kalles gjerne disse mørke variantene for liten oktav, de midtre tonene for enstrøkne og de høye tonene for tostrøkne.

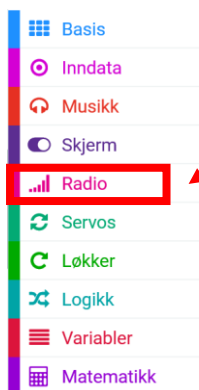


Sammenheng mellom tone og frekvens til lydbølgen					
Tone - lav	Frekvens (Hz)	Tone - midtre	Frekvens (Hz)	Tone - høy	Frekvens (Hz)
C	131	C	262	C	523
D	147	D	294	D	587
E	165	E	330	E	659
F	175	F	349	F	698
G	196	G	392	G	784
A	220	A	440	A	880
H	247	H	494	H	988

Snakk om

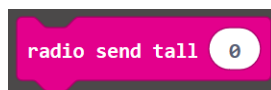
Kan dere se noen mønster for frekvensen til de forskjellige tonene i tabellen?

Radio i MakeCode

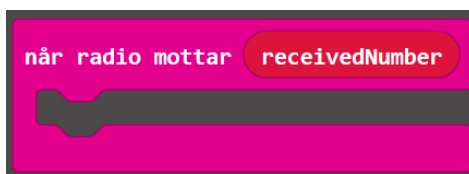


For å kunne bruke radioklossene, må du først trykke på radio i menyen.

Da får du tilgang til en rekke klosser som du kan bruke for å sende informasjon fra en micro:bit til en annen micro:bit.



Denne klossen sender et valgt tall til en annen micro:bit som er i samme radiogruppe som den micro:biten som sender tallet. Det går an å sende en variabel som man setter inn i denne blokken. Det finnes andre klosser for å sende tekster også.



Denne klossen mottar et tall fra en annen micro:bit som er i samme radiogruppe som den selv.



Denne klossen bestemmer hvilken «kanal» radioen bruker. Micro:bitene i samme radiogruppe bruker samme frekvens for å kommunisere.

Dette er en type hvis-kloss, og det som står inni klossen blir utført når den mottar et tall.

Radiogruppen må være lik på begge micro:bitene for at de skal kunne «snakke sammen». Pass på at det ikke er flere grupper i klassen som bruker samme kanal, for da vil alle micro:bitene «snakke i munnen på hverandre», og det kan bli litt kaotisk. Verdien for radiogruppen kan være mellom 0 og 255.

Lag en ringeklokke

- med et radiostyrt micro:bit-program



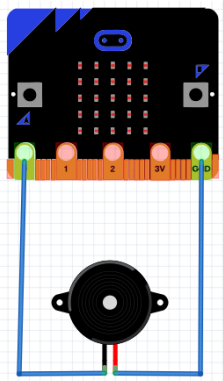
Oppgave

Lag en ringeklokke som spiller en egenkomponert melodi minst 5 meter borte når noen trykker på den.

Fase 1: Finn informasjon og inspirasjon en ringeklokke. Hvordan kan ringeklokker se ut?

Fase 2: Ha en idèmyldring for deg selv. Hvordan vil du at ringeklokka di skal se ut? Tegn gjerne en skisse før du begynner å lage den. Hvilke materialer skal du bruke? Ringeklokka skal spille av en melodi, hvilken melodi velger du?

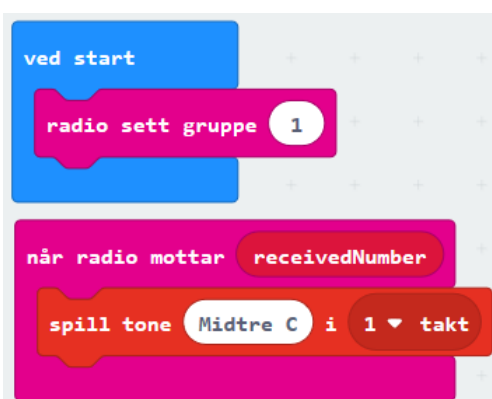
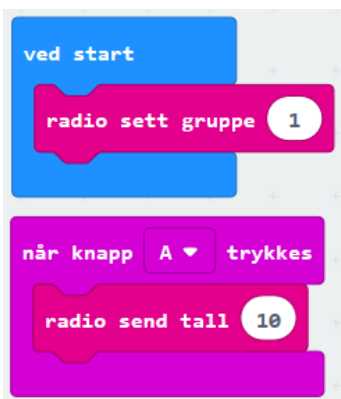
Fase 3: Tid for å lage kortet og programmere micro:biten slik at LED-pærene lyser.



Koble opp micro:bit som vist på figuren.

Hvordan må vi koble om vi vil bruke flere høyttalere samtidig?

Hvilke endringer må vi gjøre dersom vi ønsker å også få LED-pærer til å lyse?



Disse programmene får høyttaleren som er koblet til den ene micro:biten til å spille C i én takt når man trykker på knapp A på den andre micro:biten.

Hvordan kan du få den til å spille en hel melodi?

Fase 4: Test programmet ditt.

Fase 5: Virker det slik det skal?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få en best mulig ringeklokke. Gjør gjerne endringer i micro:bit-programmet ditt.

Fase 7: Forklar for en annen i klassen hvordan du gikk fram for å lage ringeklokka di.

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 6

Innhold

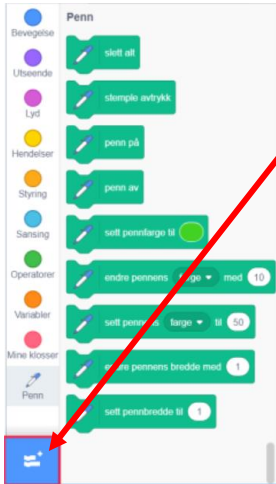
Kapittel 6 – Geometri, matematiske mønstre og modellering (9.trinn) og naturfag

- Opplegg 12 – Tegne geometriske former med programmering.....3
 - Tegning av hus til fargeleggingsbok.....4
- Opplegg 13 – Sirkler, kuler og naturressurser.....5
 - Tegning av sirkler til handlenett.....6
- Opplegg 14 – Volum for ulike geometriske former.....7
 - Utforsk volum med en volum-måler.....8
- Opplegg 15 – Matematiske mønstre.....9
 - Lag et putetrekke.....10

Opplegg 12 – Tegne geometriske former med programmering



Det går an å bruke programmering for å tegne geometriske figurer. I Scratch trenger vi å bruke noen tegnefunksjoner for å få til dette. De må importeres fra en tilleggskategori, og fremgangsmåten vises under.



1. For å finne tillegg i Scratch, må du trykke på den blå knappen nederst i det venstre hjørnet.

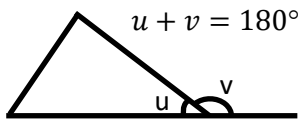
2. Da kommer du inn i en meny der man kan velge mellom mange forskjellige tillegg. Trykk så på ruta der det står penn.



3. Da blir det lagt til en meny med forskjellige muligheter for å tegne med Scratch.

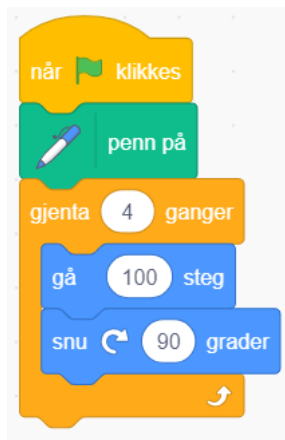
Tegning av geometriske former

Det er spesielt én ting man må tenke på når man skal tegne geometriske former med Scratch, og det er vinkler. Alle vinklene i Scratch er de ytre vinklene, det vil si supplementvinklene til de indre vinklene. To vinkler som til sammen danner 180° er supplementvinkler til hverandre. I trekanten under er u og v supplementvinkler til hverandre, og i Scratch er det vinkel v du skal bruke.



Diskuter

1. Hvilken geometrisk figur blir tegnet med de to programmene til høyre?
2. Hva er poenget med den siste delen av koden helt til høyre – må den alltid være med?
3. Hva betyr tallene i de to programmene? Blir det samme figurene, dersom noen av tallene forandres?
4. Hvordan ville du tegnet en sirkel med Scratch?



Tegning av hus til fargeleggingsbok

- Finn arealet av sammensatte figurer

Oppgave

Lag et program som tegner et hus til en fargeleggingsbok. Bruk minst tre forskjellige geometriske figurer i tegningen din. Ta skjermbilde av huset ditt og skriv det ut på papir slik at det blir en side i en matematisk fargeleggingsbok.

Fase 1: Hvilke geometriske former finnes? Sjekk at du vet om minst fem forskjellige, og hvilke kjennetegn de har. Kan du finne formlene for arealet for formene?

Fase 2: Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan nyttige forslag bli avfeid for tidlig.

Når du har klar en kladd over hvordan hustegningen din skal være, begynner planleggingen.

1. Del oppgaven opp i flere små deler
 - Hvilke geometriske figurer består tegningen din av?
 - Vet du hvordan du må programmere for å tegne de ulike delene?
 - Hvis du ikke vet, kan du finne det ut?
 - Hvis du ikke kan finne det ut, skal du endre tegningen din litt?
 - Hvordan skal du kombinere de forskjellige delene?
2. Planlegg gjennomføringen
 - Anslå hvor lang tid hver del tar.
 - Har du tid til å tegne alle delene?
 - Hvis ikke, hva skal du kutte ut?
 - Hvilke former *må* være med?

Fase 3: Lag første versjon av programmet ditt.

Fase 4: Test programmet ditt, og utforsk hva som skjer om du varierer hvilke tall du bruker i de forskjellige versjonene av programmet ditt.

Fase 5: Virker det slik det skal? Ble tegningen av huset slik du hadde planlagt?

Fase 6: Hopp gjerne tilbake til tidligere punkt, og gjør forandringer for å få en best mulig hustegning. Gjør gjerne endringer i programmet ditt.

Fase 7: Skriv ut bildet ditt slik at du kan bruke det i en matematisk fargeleggingsbok.



Oppgave

1. Finn arealet av gulvflaten i huset ditt.
2. Finn overflatearealet av huset ditt. Bruk formlene for areal av de forskjellige geometriske figurene.
3. Finn volumet av huset ditt. Må du gjøre noen antagelser?

Diskuter

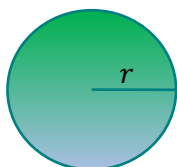
1. For at huset skal ha dobbelt så stort gulvareal, hva blir de nye målene?
2. For at huset skal få dobbelt så stort volum, hva blir de nye målene?
3. Hva blir svarene på spørsmål en og to, dersom husene må være formlike?

Opplegg 13 – Sirkler, kuler og naturressurser

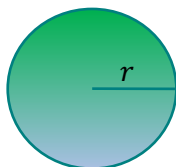


Under vises figurer av to sirkler med samme radius, og to kuler med samme radius. En sirkel er to-dimensjonal, mens en kule er tre-dimensjonal. Det betyr at vi kan ikke regne ut volumet for en sirkel, for den er helt flat, den har ingen høyde.

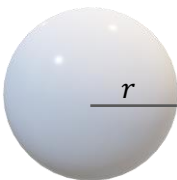
$$O = 2\pi r$$



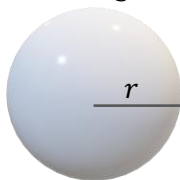
$$A = \pi \cdot r^2$$



$$A = 4\pi r^2$$



$$V = \frac{4\pi r^3}{3}$$



Diskusjonsoppgave

1. Kan vi sammenligne arealet av en sirkel med overflatearealet til en kule? Hvordan er denne sammenhengen?
2. Dersom du har en sirkel med radius r , hva blir formlene for omkretsen og arealet? Hva skjer med formlene dersom radiusen dobles, altså at den er lik $2r$?
3. Hva skjer ved en dobling av radius for volumet av en kule? HINT – se på formel for volum.
4. Sammenlign hvor mye omkrets og areal av sirkel og volum av en kule øker når radiusen blir dobbelt så stor. Kan dere si noe om hvorfor de øker på akkurat den måten?

Naturressurser

Jordkloden har mange naturressurser som mennesker har bruk for på forskjellige måter. Det kan for eksempel være metallressurser, drikkevann eller dyrkbare arealer. Eller oljeressurser som kan brukes til å lage plast eller bensin. Problemet er om mennesker bruker for mye av disse ressursene, slik at de blir brukt opp, eller at bruken av ressursene skaper problemer for naturen og mennesker. Dette kan blant annet føre til at dyre- og plantearter blir truet og dør ut, da får vi et tap av biologisk mangfold.

Diskuter

1. Hva er mest miljøvennlig av et bomullsnett og et plastbasert handlenett, eller en plastpose?
2. Hva menes med mest miljøvennlig? Se på noen av forslagene i lista og argumenter for hvilke perspektiver du mener er viktigst. Tenk gjerne ut flere perspektiver.
 - CO₂-utslipp
 - Dyrkingen/produksjonen påvirker dyre- og plantearter
 - Utslipp i naturen (forurensning)
 - Plast i havet
 - Bruk av dyrkingsareal (fortrenger dyrking av mat)
 - Vannforbruk
 - Mulighet for gjenbruk
3. Påvirker valg av perspektivene i spørsmål 2 svaret på spørsmål 1? Hva bør være med i vurderingen? Finnes det noe fasitsvar?



Tegning av sirkler til handlenett

- Måling av diameter og omkrets

Oppgave

Tegn minst 5 forskjellige sirkler ved hjelp av programmering, skriv dem ut, og mål diameter og omkrets. Bruk sirklene som mønster til dekorasjon på et handlenett. Finn en matematisk modell for sammenhengen mellom diameter og omkrets til sirklene.

Fase 1: Er det noen begreper fra oppgaven du er usikker på, prøv å finne en forklaring.

Fase 2: I denne oppgaven er det logisk å begynne på fase 2 i modellen, siden vi ikke trenger å finne ut så mye om sirkler på forhånd.

Fase 3: Tid for å lage sirklene, skrive dem ut, måle dem og bruke som mønster til stoffsirkler til handlenettet.

Del oppgaven opp i flere små deler

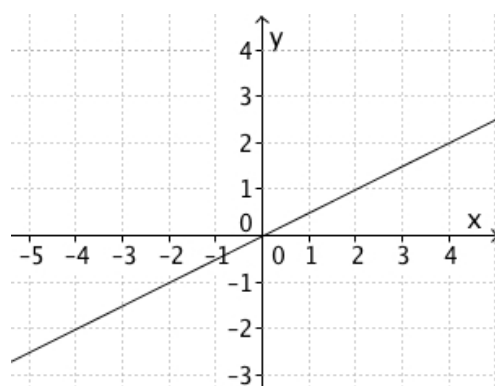
- Hvordan skal du lage minst 5 forskjellige sirkler?
- Hvordan bruke programmering?
- Har du noen lignende programmer fra før?
- Hvordan skal du måle diameter og omkrets?
- Trenger du noe utstyr for å gjennomføre målingene?



Gjør fase 4 – 7.

Oppgaver

1. Mål diameter og omkrets for de forskjellige sirklene du tegnet ved hjelp av programmering.
2. Plott alle de målte verdiene i Geogebra.
3. Finn en matematisk modell ved å foreta en regresjon for de målte dataene.
4. Hva er hovedforskjellen mellom andre modeller dere har laget tidligere og denne modellen?

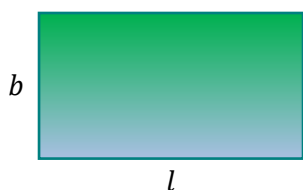


Opplegg 14 - Volum for ulike geometriske former

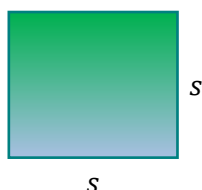
Lengde forteller oss hvor stor avstanden er mellom to punkter. Den er endimensjonal, og måles i for eksempel enhetene cm eller meter.

Areal gir oss informasjon om hvor stor en flate er, slik at vi kan sammenligne størrelsen på grunnflaten til forskjellige figurer, for eksempel om man skal kjøpe et hus. Areal er todimensjonalt, det vil si at det består av to lengder. Men som regel kaller vi det lengde og bredde, eller grunnlinje og høyde. Disse to avstandene må multipliseres for å få et areal, dermed blir enhetene for areal alltid en lengde opphøyd i andre, for eksempel cm^2 eller m^2 .

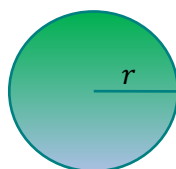
$$A = l \cdot b$$



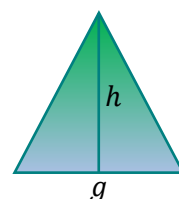
$$A = s^2$$



$$A = \pi \cdot r^2$$



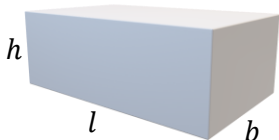
$$A = \frac{g \cdot h}{2}$$



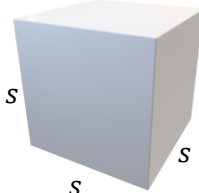
På samme måte gir volum informasjon om hvor stor plass noe tar, eller hvor stor plass det er inni noe. Volum er en 3-dimensjonal størrelse, og vi får derfor ikke tegnet den skikkelig på en 2-dimensjonal flate slik som i en bok.

For å beregne volum må vi alltid multiplisere tre lengder med hverandre. Sjekk gjerne med formlene under! Da blir enheten for volum alltid en lengde opphøyd i tredje, slik som cm^3 eller m^3 . Alle fysiske gjenstander har tre dimensjoner, men noen eller alle lengdene kan være veldig korte slik at volumet blir veldig lite. Dette gjelder for eksempel et hårstrå eller et ark silkepapir eller bladgull.

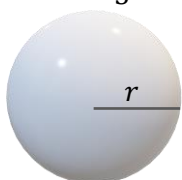
$$V = l \cdot b \cdot h$$



$$V = s^3$$



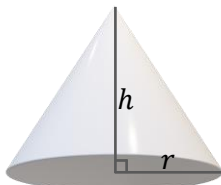
$$V = \frac{4\pi r^3}{3}$$



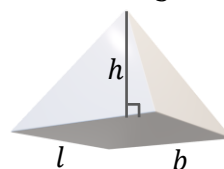
$$V = \pi r^2 \cdot h$$



$$V = \frac{\pi r^2 \cdot h}{3}$$



$$V = \frac{l \cdot b \cdot h}{3}$$



Diskuter

1. Har noen av 3D-figurerne noe til felles?
2. Hvordan vil dere fordele figurene i grupper? Er det noen figurer som passer i flere grupper?
3. Kan dere se noe felles for volumformlene for de forskjellige gruppene?

Utforsk volum med en volummåler

Oppgave

Lag en geometrisk form i minst 5 forskjellige størrelser. De skal brukes i en målesylinder, for å måle volumet på dem. Så skal dere bruke målingen for volum til å finne formelen for volum ved hjelp av regresjon i geogebra. Til slutt skal dere lage et program som dere kan bruke til å utforske hva som skjer med volumet til figuren hvis du dobler sidelengden, radius eller høyden.

Fase 1: Finn hvilke former det kan være lurt å lage? HINT: Det er en stor fordel med bare én variabel. Spør gjerne lærer om tips.

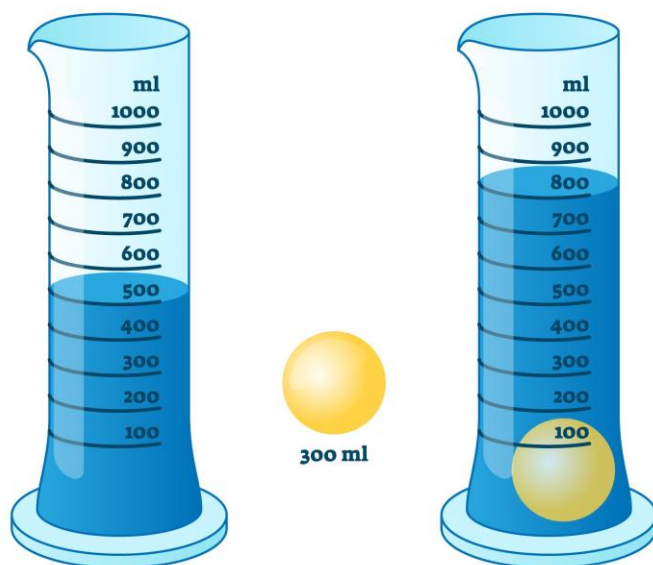
Fase 2: Ha en idémyldring for deg selv. Tegn gjerne en skisse før du diskuterer med de andre.

Fase 3: Tid for å lage figurene, og måle volum og den ene variabelen (sidelengde, radius eller høyde). Regresjonen og programmeringen må vi gjøre til slutt, for vi må ha alle målingene klare først.

Gjør fase 4 – 7.

Regresjon – vise volumformler

1. Mål volum og sidelengde, radius eller høyde.
2. Plott alle de målte verdiene i Geogebra.
3. Finn en matematisk modell ved å foreta en regresjon for de målte dataene.
 - Hvilken form tror du grafen vil ha?
4. Hva er hovedforskjellen mellom andre modeller dere har laget og denne modellen?



Programmering

- Bruk volumformelen dere kom fram til i regresjonen, og sjekk om den stemmer med bildet på forrige side.
- Lag et program som der dere skriver inn hvor stor variabelen skal være (sidelengde, radius eller høyde) og som regner ut hvor stort volumet blir i det tilfellet.
- Hva mer kan du utforske ved å forandre litt på programmet?

Programmeringstips

For å regne ut volumet til en kule, så vil dere trenge en kloss som er sammensatt på denne måten.



Opplegg 15 - Matematiske mønster

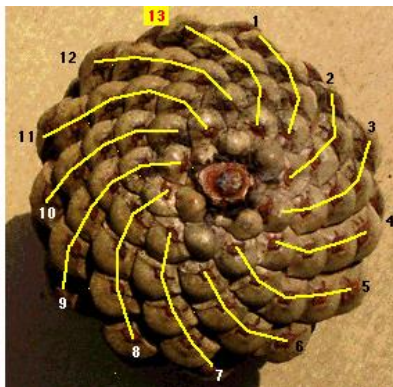
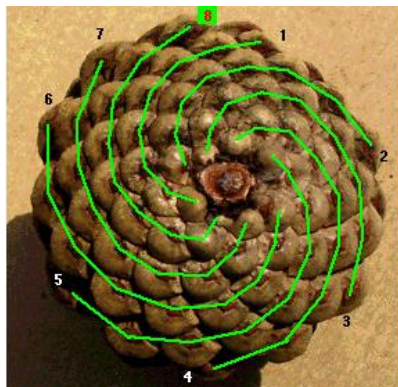
Det finnes mange matematiske mønster i naturen. Noen er basert på en tallrekke som kalles Fibonacci-tallene. Denne rekka består av tall der hvert tall er lik summen av de to foregående tallene i rekka.

Fibonacci rekka: 0 – 1 – 1 – 2 – 3 – 5 – 8 – 13 – 21 – 34



Fibonacci-tallene i naturen

Dersom man ser på spiralene i en konge eller ananas, kan man se at de ligner spiralene i solsikkefrøene eller romanesco-kålen. Antall spiraler i hver retning er to påfølgende Fibonacci-tall. Ved å se på frøkapselstrukturen til en moden hunnkonge kan man telle seg fram til 8 spiraler i den ene retningen (merket med grønt) og 13 spiraler i den andre retningen (merket med gult).



Illustrasjon: John R. Simmons, University of Georgia

Diskuter

1. Hva tror du de tre neste tallene i rekka blir?
2. Lag minst to egne tallrekker, og la noen annen i klassen finne det neste tallet.
3. Kan du finne ut hva trekantallene er?

Geometriske mønster

Vi kan også lage mønstre ved hjelp av geometriske figurer, og det passer veldig fint å gjøre det ved hjelp av programmering!

Ofte kan vi få mønstre med mange symmetrilinjer ved å tegne en geometrisk figur, og så forskyve den med for eksempel 20 grader, og gjenta dette helt til figuren har blitt dreid 360 grader. Da vil vi få en figur som kan ligne litt på en blomst.

Oppgaver

1. Hva er poenget med koden til høyre i Scratch-programmet på denne siden?
2. Hvor mange symmetrilinjer har «blomsten» tegnet av dette Scratch-programmet?
3. Hva er det i programmet som bestemmer hvor mange symmetrilinjer vi har?
4. Hvordan ville en trekantblomst se ut? Lag programmet og sjekk.

Dette programmet tegner denne «blomsten».

Lag et putetrekk

- med et matematisk mønster



Oppgave

Lag et program som tegner et matematisk mønster. Dette mønsteret skal overføres til et putetrekk ved bruk av brodering/symaskin, varmpresset vinyl, limt på med tråd, malt med stoffmaling eller andre metoder.

Fase 1: Undersøk ulike matematiske mønstre. Det kan være geometriske mønstre, tallmønstre eller andre typer mønstre. Hva kalles forskjellige typer matematiske mønstre? Hva har de til felles?

Fase 2: Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan nyttige forslag bli avfeid for tidlig.

Tips: Hvilket motiv kan få folk til å synes at matematiske mønstre er spennende? Kan du forenkle det slik at du klarer å tegne det med Scratch?

- Tenk selv først og tegn gjerne flere skisser.
- Forklar ideen din for de andre på gruppa.
- Hele gruppa diskuterer de ulike ideene, og alle bestemmer seg for hvilket matematisk mønster de vil bruke.

Tenk gjennom hvordan du skal få mønsteret overført til et putetrekk. Vil du helst brodere mønsteret for hånd, sy på symaskin, lime det på eller noe helt annet? Diskuter gjerne med de andre hva de har tenkt.

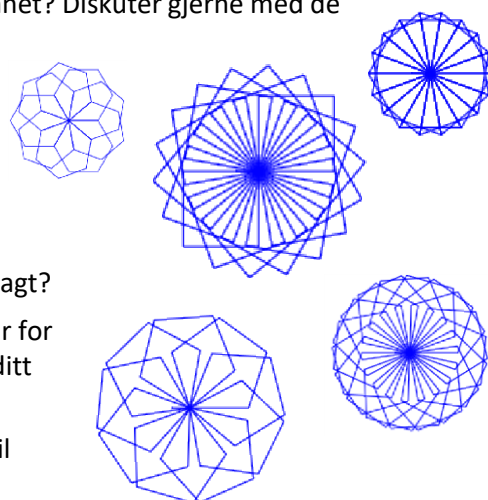
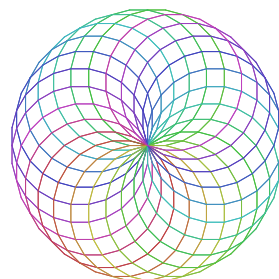
Fase 3: Lag første versjon av programmet ditt.

Fase 4: Test programmet ditt, og utforsk hva som skjer om du varierer hvilke tall du bruker i de forskjellige versjonene av programmet ditt.

Fase 5: Virker det slik det skal? Ble mønsteret slik du hadde planlagt?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få et best mulig mønster. Gjør gjerne endringer i programmet ditt og/eller din plan for å overføre det til putetrekket ditt.

Fase 7: Skriv ut bildet ditt slik at du kan bruke det som mønster til putetrekket ditt.



Oppgaver

1. Sammenlign mønsteret ditt med mønstrene til de andre i klassen. Har noen av dere formlike eller kongruente mønstre?
2. Kan du programmere noen formlike versjoner av ditt eget mønster?
3. Hva er likt, og hva er ulikt i programmene for de forskjellige formlike mønstrene?

Ekstraoppgave

1. Hva blir arealet av figuren din? Kan du justere størrelsen slik at figuren din dekker et område på 200 cm²?
2. Må du gjøre noen antagelser for å klare det? I så fall hvilke?

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 7

Innhold

Kapittel 7 – Sannsynlighet (9.trinn) og naturfag

• Forsøk, simuleringer og sannsynlighet.....	3
• Opplegg 16 – Sannsynlighet og representasjoner.....	4
• Lag en klimatilpasset art.....	5
• Opplegg 17 – Uniform sannsynlighet.....	6
• Lag en terning eller et lykkehjul.....	7
• Opplegg 18 – Antall mulige kombinasjoner.....	8
• Lag en magisk krystallkule.....	9
• Opplegg 19 – Sannsynlighet og valgtre.....	10
• Lag et spill.....	11

Forsøk, simuleringer og sannsynlighet

Hvordan finner vi sannsynligheten for at noe spesifikt skal skje? Dersom vi vil finne sannsynligheten for å få en sekser på et terningkast, kan vi kaste en terning flere ganger og telle opp hvor mange ganger vi får sekser. Hvis du kaster terningen 50 ganger, og 11 av gangene får du sekser. Da kan vi finne sannsynligheten for å få sekser ved å dele antall seksere på så mange ganger du kastet terningen.

$$P(\text{sekser}) = \frac{11}{50} \approx 0,22$$

Da har du gjort et forsøk, og funnet ut at sannsynligheten er 0,22 eller 22 % for å få sekser. Men dette hørtes vel litt rart ut? Burde det ikke være like stor sjanse for å få sekser som ener eller firer? Hvis det er like stor sjanse for å få de forskjellige verdiene på terningen, så burde hver av de seks mulighetene ha sannsynlighet på $\frac{1}{6} \approx 0,17 = 17\%$.

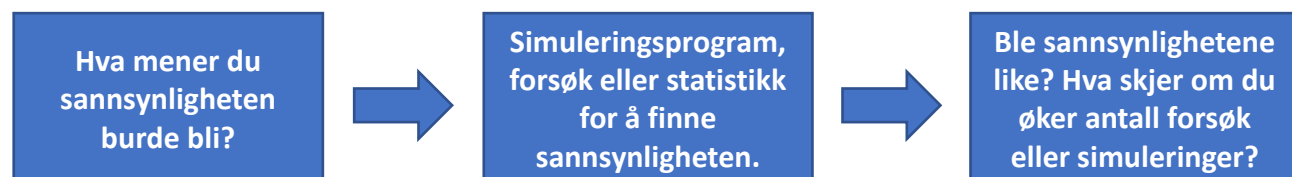
$$0,17 \neq 0,22$$

Hvorfor blir det ikke likt når vi gjør forsøket i virkeligheten? Er terningen vår en jukse-terning? Nei, det er slik at dersom vi gjør et forsøk i virkeligheten, så kan vi få litt rare svar om vi ikke gjør forsøket så mange ganger. Dette kommer av at terningen ikke vet hva den landet på forrige gang, og det er derfor helt tilfeldig hva den havner på neste gang. Vi sier at de to kastene er uavhengige av hverandre.

Dersom du kaster terningen 100 ganger i stedet, så vil sannsynligheten for å få sekser bli nærmere 0,17. Men du må kanskje kaste nesten 10000 ganger før den blir nøyaktig nok. Det blir nok litt kjedelig!

Da kan man bruke programmering i stedet! Når man lager et program som «later som» om det gjør et forsøk, så har vi laget en simulering. Denne simuleringen (programmet vi laget) kan vi enkelt bruke for å simulere terningkast, og sjekke at den simulerte sannsynligheten blir likere og likere den verdien den burde ha, jo flere simuleringer vi gjør.

Det finnes altså hovedsakelig tre måter for å finne hva en sannsynlighet er. Vi kan tenke logisk, også gjøre et forsøk (eller finne statistikk) eller en simulering for å sjekke om det stemmer. Ofte er det upraktisk med forsøk, det tar gjerne for lang tid, da kan man bruke simulering i stedet.



Utfall

De mulighetene som kan skje når vi gjør et statistisk forsøk. Det å kaste en ener på terningen er et utfall, og det å kaste en firer på terningen er et annet utfall. Sannsynligheten for alle mulige utfall vil alltid til sammen bli 1.

Hvordan beregne sannsynlighet?

Vi skal kaste en terning og finne sannsynligheten for å få ener eller toer. Først må vi telle opp hvor mange utfall som er gunstige for resultatet vårt, altså hvor mange utfall som gir oss ener eller toer. Det blir 2 utfall (ener er ett utfall og toer er ett til utfall, to til sammen). Så må vi finne hvor mange mulige utfall vi har, altså hvor mange forskjellige muligheter vi har når vi kaster en terning. Det er seks mulige utfall, siden en terning har seks sider. For å finne sannsynligheten må vi dividere gunstige utfall på mulige utfall:

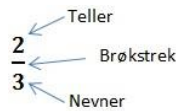
$$P(\text{ener eller toer}) = \frac{\text{gunstige utfall}}{\text{mulige utfall}} = \frac{2}{6} = \frac{1}{3}$$

Opplegg 16 - Sannsynlighet og representasjoner

Sannsynligheter kan oppgis som brøk, desimaltall eller prosent, og slik er sammenhengen mellom dem:

Brøk

En brøk består av en teller, en brøkstrek og en nevner. Brøkstreken kan vi se på som et deletegn.



Desimaltall

Siden brøkstreken kan ses på som et deletegn, kan vi enkelt regne ut desimaltallet fra brøken. Vi deler telleren på nevneren og får da desimaltallet: $\frac{2}{3} = 2 : 3 = 0,67$

Prosent – betyr hundredel

Siden desimaltallet viser hvor mange enere vi har, kan vi multiplisere med 100. Da får vi prosent, altså hvor mange hundredeler vi har: $0,67 \cdot 100 \% = 67 \%$

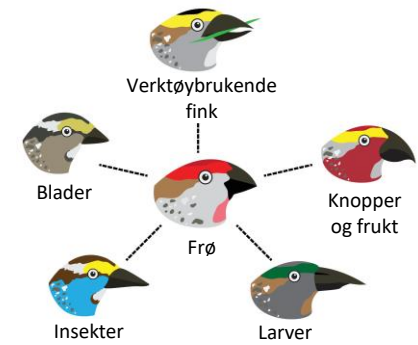
Evolusjonsteorien

I 1836 kom Charles Darwin hjem fra en jordomseiling med skipet Beagle. På turen hadde de besøkt Galapagosøyene, en øygruppe som tilhører Ecuador i Sør-Amerika. Darwin observerte dyrelivet på øyene og la merke til at finkene på forskjellige øyer hadde ulike nebb. Dette kom til å bli en viktig observasjon i hans arbeid med evolusjonsteorien.

Grunnen til at finkene hadde ulike nebb, var at de forskjellige øyene hadde tilgang til forskjellig mat for finkene. Finkene hadde litt forskjellige nebb når de ble klekket, og de finkene som hadde nebb som gjorde det enkelt å spise den maten det var størst tilgang til, ville også overleve lengst når det ble konkurranse om maten.

De finkene som overlevde lengst, var også de som fikk flest etterkommere. Over tid ville denne gradvise endringen føre til at det levde forskjellige typer finker på forskjellige øyer. Siden øyene hadde tilgang på forskjellige typer mat, ville det bare være den finketypen med nebb tilpasset akkurat den tilgjengelige maten, som levde på hver øy.

Det at den arten eller varianten av en art som er best tilpasset omgivelsene sine (skaffe seg mat, ly, få mange avkom o.l.), vil bli dominerende, er en viktig del av evolusjonsteorien. En art vil på den måten utvikle seg på en slik måte at den er tilpasset sine omgivelser. Siden dette kan gjøres på utrolig mange forskjellige måter, har vi stor variasjon blant de forskjellige artene.



Snakk om

Hva har klimaendringer med evolusjonsteori å gjøre?

Hvordan kan forskjellige arter tilpasse seg klimaendringene?

Artskjennetegn	Antall døde per 1000 dyr/fugler første leveår
Vinger	150
Nattaktiv	100
Gode graveklør	80
Tykk pels	75
Skarpe tenner	50
Lange ører	250

Dette er fiktive tall, da det ikke finnes noen ekte oversikt.

Sannsynlighet fra statistikk

I tabellen er det oversikt over dødelighet per 100 dyr eller fugler for deres første leveår. Dermed kan vi bruke de verdiene for å finne ut sannsynligheten for at dyret eller fuglen dør før den har ettårsdag.

Vi velger å se på fugler med vinger, og der dør 150 stykker per 1000 fugler.

Sannsynligheten for at fuglene med vinger vil dø i sitt første leveår blir da:

$$P(død) = \frac{150}{1000} = \frac{15}{100} = \frac{3}{20}$$

Lag en klimatilpasset art

- lag et program som regner om brøk til desimaltall og prosent

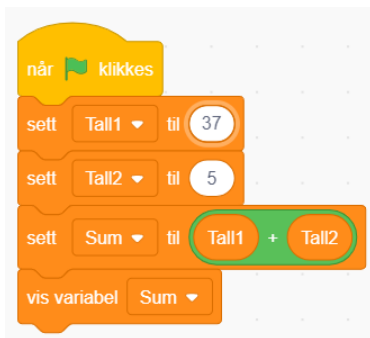
Oppgave

Lag en art som er bedre tilpasset klimaendringer. Du skal beregne sannsynligheten for at den dør i løpet av første leveår ved å bruke tabellen med artskjennetegn på forrige side, oppgitt som en brøk. Lag et program som regner om en brøk til desimaltall og prosent, og skriv sannsynligheten som både desimaltall og prosent.

Fase 1: Vet du hva mulige konsekvenser av klimaendringer kan være? Hvis ikke, finn det ut. Finn informasjon om hva klimaendringer er, og hvordan det påvirker forskjellige områder på jordkloden. Bruk dette som inspirasjon til en art som er bedre tilpasset klimaendringene.

Fase 2: Ha en idémyldring for deg selv. Hvordan vil du at arten din skal se ut? Tegn gjerne en skisse før du diskuterer med de andre. Deretter må gruppa bestemme hvordan arten skal se ut. Husk å ha med noe der dere kan skrive opp sannsynligheten som dere skal regne ut.

Fase 3: Tid for å lage et program som omgjør fra brøk til desimaltall og prosent, og å lage en modell av den nye arten.



Hva tror du disse programmene gjør?

Hva er forskjellen på dem?

Hvordan kan dere bruke noe av dette i deres program?

Lag omregningsprogrammet.

Del gjerne oppgaven opp i flere små deler

- Vet du hvordan du må programmere for å regne ut fra brøk til desimaltall?
- Vet du hvordan du må programmere for å regne ut til prosent?
- Hvis du ikke vet, kan du finne det ut?
- Hvordan skal du kombinere de forskjellige delene?



Fase 4: Test programmet ditt. Be andre i klassen om tilbakemeldinger på modellen. Kan de se hva det skal forestille?

Fase 5: Virker alt slik det skal? Er svaret fra programmet logisk?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få en best mulig klimatilpasset art. Gjør gjerne endringer i programmet ditt. Kan det gjøres mer effektivt eller ryddigere?

Fase 7: Skriv ned utregningen for sannsynligheten, og oppgi den som både brøk, desimaltall og prosent. Forklar hvordan din gruppe tenkte for en person fra en annen gruppe.

Oppgave

Hvilke konsekvenser av klimaendringene er arten deres spesielt godt tilpasset? Forklar hvorfor.

Fører denne tilpasningen til større overlevelse over alt, tror du? Hvorfor/hvorfor ikke?

Opplegg 17 - Uniform sannsynlighet

Uniform sannsynlighet

Alle muligheter har like stor sannsynlighet for å skje.

Eksempler på uniform sannsynlighet kan være å trekke et kort fra en kortstokk, eller å kaste mynt eller kron. Det er ingen grunn til at det skal være mer sannsynlig å trekke akkurat hjerter tre enn kløver fem. Det er heller ingen grunn til at det skulle være mer sannsynlig at en mynt lander med kronen opp, enn myntsiden opp (den med hodet). Vi ser bort fra den utrolig lille sannsynligheten for at mynten lander på kanten.



Diskuter

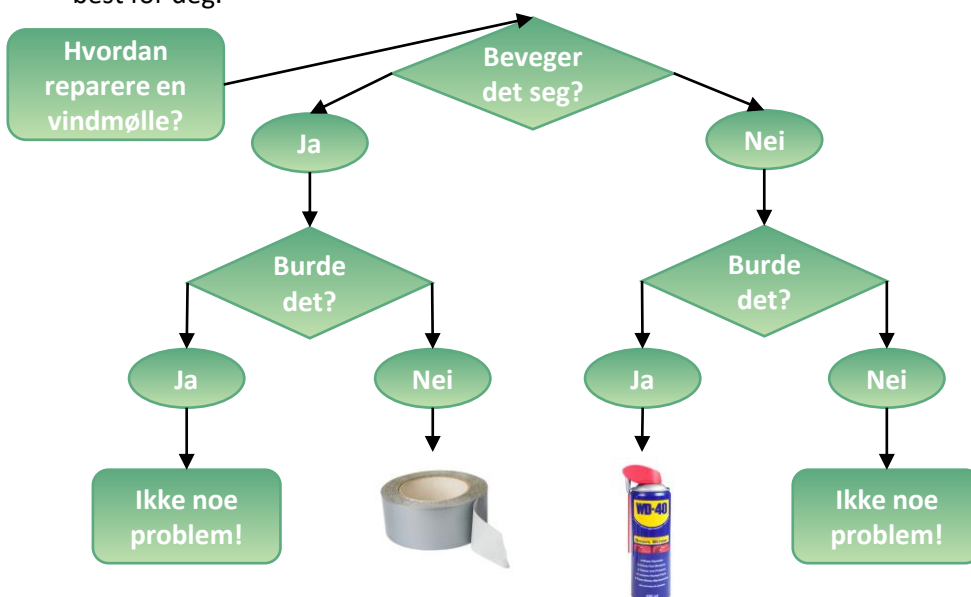
1. Kan du finne noen andre eksempler på uniform sannsynlighet
2. Kan du finne noen eksempler som ikke er uniforme?
3. Hvordan kan vi sjekke om eksemplene våre er uniforme eller ikke?

Pseudokode og flytskjema

I begynnelsen av boka er det to oppgaver der dere skal programmere hverandre, uten å bruke et programmeringsspråk. Da endte dere opp med å skrive en oppskrift, en algoritme, med vanlige ord. Dette kan kalles pseudokode. Altså det å skrive et program med det vanlige språket vårt, men gjerne med stikkord, men ikke med et programmeringsspråk.

Når man skal lage et litt større program, så kan det være lurt å bruke pseudokode først. Da skriver du hva programmet skal gjøre med egne ord, og etterpå oversetter du dette til enten Scratch- eller Pythonkode.

Et alternativ er å lage et flytskjema først. Et flytskjema er et slags oversikt over programmet. Man bruker forskjellige små figurer som betyr forskjellige ting. En form for spørsmål, en form for svar og en form for begynnelse og slutt. Du kan bruke både flytskjema og pseudokode når du skal lage et program, men som regel er det greit å velge det som passer best for deg.



Begynnelse
eller slutt

Svar

Spørsmål

Oppgaver

1. Hva viser dette flytskjemaet?
2. Lag et flytskjema for hvordan rydde rommet ditt (søk på «funny flowcharts» for inspirasjon).
3. Lag en pseudokode for å rydde rommet ditt.



Lag en terning eller et lykkehjul

- og et program som simulerer sannsynligheten



© Inter IKEA Systems B.V. 2020

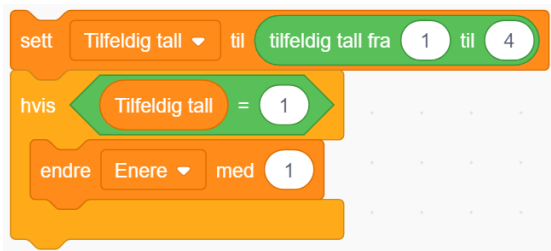
Oppgave

Lag en egendesignet terning som ikke har seks sider, eller et lykkehjul som senere skal være en del av et spill. Du skal lage et program som du kan bruke til å simulere terningkast eller lykkehjul-snurr og finne sannsynligheten for de forskjellige utfallene.

Fase 1: Finn informasjon og inspirasjon til en terning eller et lykkehjul. Hvilken form kan en terning ha? Hvilke deler består et lykkehjul av? Hvordan ser disse delene ut?

Fase 2: Ha en idèmyldring for deg selv. Tegn gjerne en skisse før du diskuterer med de andre. Deretter må gruppa bestemme hvordan terningen/lykkehjulet skal se ut.

Fase 3: Tid for å lage simuleringsprogrammet, og å lage terningen eller lykkehjulet. Se på programmeringseksempelene for et firedelt lykkehjul under:



Hva tror du disse programmene gjør?
Hvordan kan du bruke det i ditt simuleringsprogram?

Lag et simuleringsprogram – det finnes ark med HINT (spør lærer)

Del oppgaven opp i flere små deler

- Vet du hvordan du må programmere for å få et tilfeldig terningkast eller lykkehjul-spinn?
- Vet du hvordan du må programmere for å telle opp hvor mange det blir av hvert utfall?
- Hvis du ikke vet, kan du finne det ut?
- Kan du bruke noe fra programmeringseksempelene over?
- Hvordan skal du kombinere de forskjellige delene?

Kan du lage et flytskjema her?



Fase 4: Test terningen eller lykkehjulet og programmet ditt.

Fase 5: Virker alt slik det skal?

Fase 6: Hopp gjerne tilbake til tidligere punkt, og gjør forandringer for å få en best mulig terning eller lykkehjul. Gjør gjerne endringer i modelleringsprogrammet ditt. Kan det gjøres mer effektivt eller ryddigere?

Fase 7: Lag et ark med bilder fra hele prosessen sammen med resultatet for sannsynligheten til de forskjellige utfallene.

Opplegg 18 - Antall mulige kombinasjoner



Regel for sannsynlighet

Vi har alltid denne regelen for beregning av sannsynlighet:

$$P(\text{utfall}) = \frac{\text{gunstige utfall}}{\text{mulige utfall}}$$

Hendelse

Satt sammen av ett eller flere utfall.

Gunstige utfall

Utfallene som er med i hendelsen du skal finne sannsynligheten til.

Mulige utfall

Alle utfallene som er mulige å få.

For å bruke denne regelen, må vi kunne finne ut hvor mange gunstige utfall vi har, og hvor mange mulige utfall vi har. En måte å gjøre dette på, dersom man ikke har altfor mange utfall, er å tegne opp alle utfallene i en tabell og telle opp hvilke som er gunstige.

Vi skal finne ut hvor stor sannsynlighet det er for at summen av øyne på terningene blir 9, når vi kaster to forskjellige terninger. En av terningene er vanlig og har seks sider, mens den andre har åtte sider. Vi lager en tabell (som en krysstabell for gener, men med flere utfall):

+	1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8	9
2	3	4	5	6	7	8	9	10
3	4	5	6	7	8	9	10	11
4	5	6	7	8	9	10	11	12
5	6	7	8	9	10	11	12	13
6	7	8	9	10	11	12	13	14

For å finne hvor mange gunstige utfall vi har, må vi telle alle utfallene der vi får at summen blir 9. Det blir 6 utfall.

For å finne hvor mange mulige utfall vi har, så må vi telle alle utfallene i tabellen vår. Eller må vi det? Dersom vi skal finne ut hvor mange ruter det er i tabellen vår, så kan vi vel bare multiplisere antall utfall for hver av terningene?

Den ene terningen har 6 sider, og dermed 6 mulige utfall. Den andre terningen har 8 sider, og dermed 8 mulige utfall. For hver av de 6 sidene på den ene terningen, kan vi kombinere den med hver av de 8 sidene på den andre terningen. Altså blir det 8 mulige summer for hver utfall på den 6-sida terningen. Dermed kan vi multiplisere 8 med 6 for å finne alle de mulige utfallene: $6 \cdot 8 = 48$. Da kan vi regne ut sannsynligheten for at summen på terningene blir 9:

$$P(\text{summen er 9}) = \frac{\text{gunstige utfall}}{\text{mulige utfall}} = \frac{6}{48} = \frac{1}{8}$$

Regel for antall mulige utfall

For å beregne antall mulige utfall for en hendelse, kan vi multiplisere antall mulige hendelser for hvert av utfallene hendelsen består av.



Oppgaver

1. Finn ut hvor stor sannsynlighet det er for å trekke et billedkort (inkludert ess) fra en vanlig kortstokk. Trenger du å tegne en tabell?
2. Finn ut hvor stor sannsynlighet det er for at produktet av øynene på to vanlige terninger blir 14. Trenger du å tegne en tabell?
3. Hva gjorde at du trengte å tegne tabell på en av oppgavene, men ikke den andre? Kan du lage en regel for når det er lurt å tegne en tabell?

Lag en magisk krystallkule

- og tilpass programmet som simulerer sannsynligheten

Oppgave

Bruk tilgjengelige materialer til å lage en «krystallkule» med plass til micro:biten inni. Den skal bruke et program som tilfeldig velger blant minst 8 spådommer. Du skal også lage et program som du kan bruke til å simulere trekning av spådommer og finne sannsynligheten for de forskjellige utfallene.

Fase 1: Finn inspirasjon til spådommene og hvordan «krystallkula» skal se ut. Hvilke materialer skal du bruke? Hvor mange spådommer skal du lage?

Fase 2: Ha en idèmyldring for deg selv. Hvordan skal micro:bit-programmet ditt vise hvilken spådom som blir tilfeldig trukket? Tegn gjerne en skisse før du diskuterer med de andre. Deretter må gruppa bestemme hvordan krystallkula skal se ut.

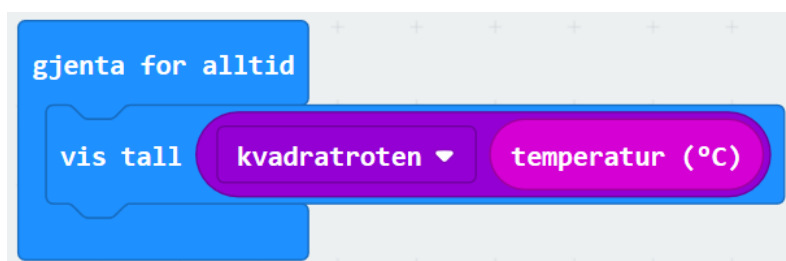
Fase 3: Tid for å lage «krystallkula» og spådommene, i tillegg til programmene:

- Lage et program til micro:biten som velger tilfeldig blant spådommene dine, og viser resultatet.
- Endre det tidligere simuleringsprogrammet ditt (fra opplegg 17) slik at det kan simulere trekking av spådommer, og beregning av sannsynligheten for de forskjellige utfallene.



Hva tror du dette programmet gjør?

Hvordan kan du bruke noe av det i ditt program på micro:biten?



Fase 4: Test krystallkula og programmene dine.

Fase 5: Virker alt slik det skal?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få en best mulig krystallkule og micro:bit-program. Gjør gjerne endringer i simuleringsprogrammet ditt. Kan det gjøres mer effektivt eller ryddigere?

Fase 7: Dokumenter hele prosessen sammen med resultatet for sannsynligheten til de forskjellige utfallene på en valgfri måte.

Opplegg 19 - Sannsynlighet og valgtre



Regel for multiplikasjon av sannsynligheter

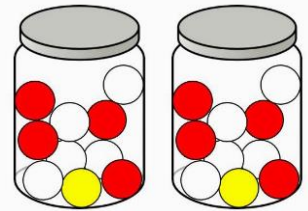
Dersom vi har en hendelse som består av utfall som er uavhengige av hverandre, kan vi finne sannsynligheten for hvert av utfallene, og multiplisere dem sammen for å finne sannsynligheten til den sammensatte hendelsen:

$$P(\text{hendelse}) = P(\text{utfall 1}) \cdot P(\text{utfall 2})$$

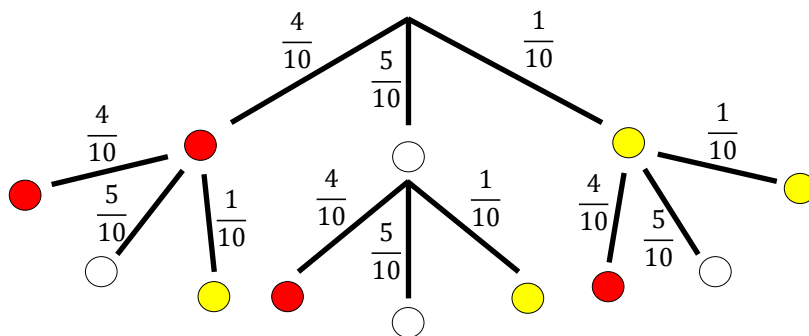
Uavhengige utfall

Et utfall påvirker ikke et annet utfall. Eksempler er å trille en terning flere ganger eller å slå mynt eller kron. Det gjelder også når man trekker flere kort fra en kortstokk dersom man legger kortene tilbake mellom hver gang man trekker.

Et valgtre brukes når man skal finne sannsynligheten til en sammensatt hendelse, som består av flere tilfeldige delforsøk. Vi kan se på to glass med 10 farga klinkekuler i hvert. 4 er røde, 5 er hvite og en er gul i hvert av glassene. Vi skal finne sannsynligheten for å trekke en rød kule fra hvert av glassene.



Først kan vi starte med å få oversikt over hvilke mulige utfall vi har, og da kan vi bruke et valgtre. Her har vi to uavhengige delforsøk, siden hva vi trekker opp av det ene glasset ikke påvirker hva vi trekker opp fra det andre glasset. Et valgtre begynner med en grein for hvert mulig utfall for det ene delforsøket. Her har vi tre mulige utfall; rød, hvit eller gul klinkekule. Så for hver grein deler vi opp i de mulige utfallene i delforsøk 2, som også er rød, hvit eller gul. Da blir valgtreet slik:



Til slutt må vi sette på hvor stor sannsynlighet det er for hver grein, slik at det blir enkelt å regne ut sannsynligheten for å trekke en rød fra hvert glass. Ser du hvordan sannsynligheten er regnet ut? HINT: $\frac{\text{gunstige utfall}}{\text{mulige utfall}}$

For å finne ut sannsynligheten, så finner vi ut av hvor mange sammensatte greiner som gir et gunstig utfall, altså rød kule fra begge glass. Vi ser av figuren at det er bare én sammensatt grein som gir to røde kuler. Vi vet at de to delforsøkene er uavhengige av hverandre, og dermed kan vi bare multiplisere sannsynligheten for å få rød fra den ene krukka med sannsynligheten for å få rød fra den andre krukka. Vi får da:

$$P(2 \text{ røde kuler}) = \frac{4}{10} \cdot \frac{4}{10} = \frac{16}{100} = \frac{4}{25}$$

TIPS:

Det kan være lurt å forkorte brøkene før man setter dem inn i valgtreet, slik at det ikke blir så store tall i regnestykkene. Da blir det enklere å forkorte etter utregning også.

Diskuter

Hvordan vil valgtreet bli dersom du bare har én krukke du skal trekke to røde kuler fra?

1. Hva blir forskjellen fra eksempelet over?
2. Hvordan vil sannsynlighetene og valgtreet da bli?

Lag et spill

- og tilpass programmet som simulerer sannsynligheten

Oppgave

Bruk terningen eller lykkhjulet fra opplegg 17 for å lage et spill. Du skal også lage et program som du kan bruke til å simulere at du triller to terninger eller snurrer to ganger på lykkhjulet og beregner sannsynligheten for de forskjellige mulige utfallene.



Fase 1: Finn informasjon om forskjellige typer spill som finnes, og bruk det som inspirasjon til ditt eget spill. Hvilke materialer skal du bruke? Husk at det må inkludere minst én terning eller ett lykkhjul.

Fase 2: Ha en idèmyldring for deg selv. Tegn gjerne en skisse (eller mange skisser) før du diskuterer med de andre. Deretter må gruppa bestemme hvordan spillet skal se ut.

Fase 3: Tid for å lage spillet, i tillegg til simuleringsprogrammet:

- Endre det tidligere simuleringsprogrammet (fra opplegg 17 og 18) til å brukes for å simulere trilling av to terninger, eller å snurre to ganger på lykkhjulet, og beregne sannsynligheten for de forskjellige utfallene.

HINT

Det kan være lurt å tegne et valgtre for alle utfallene dine før du begynner å jobbe med simuleringsprogrammet.

Kan du lage et flytskjema her?



Fase 4: Test programmet ditt. Ta en liten prøverunde med spillet.

Fase 5: Virker alt slik det skal? Var spillet gøy?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få et best mulig spill. Gjør gjerne endringer i simuleringsprogrammet ditt. Kan det gjøres mer effektivt, enklere eller ryddigere?

Fase 7: Dokumenter hele prosessen sammen med resultatet for sannsynligheten til de forskjellige utfallene på en valgfri måte.

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 8

Innhold

Kapittel 8 – Statistikk (9.trinn) og naturfag

- Opplegg 20 – Statistikk og sentralmål.....3
 - Lag en gjennomsnittskalkulator.....4
- Opplegg 21 – Frekvenstabell og sektordiagram.....5
 - Lag en spørreundersøkelse.....6
- Opplegg 22 – Statistikk og diagrammer.....7
 - Lag en statistisk fremstilling.....8
- Opplegg 23 – Statistikk og «fake news».....9
 - Lag en villedende statistisk fremstilling.....10
- Opplegg 24 – Statistikk og spredningsmål.....11
 - Lag en standardavvikskalkulator.....12
- Opplegg 25 – Statistikk i naturfag og samfunnsfag.....13
 - Lag en statistisk fargeleggingsnok.....14

Opplegg 20 - Statistikk og sentralmål



På Bore skole er det 23 lærere, og disse har alle notert hvilken skostørrelse de bruker. Resultatet ble: 39, 40, 37, 37, 39, 38, 39, 36, 43, 38, 37, 38, 39, 42, 43, 38, 38, 38, 44, 40, 37, 39 og 43.

Dette er ganske uoversiktlig, derfor finnes det flere måter å få bedre oversikt over innsamlede data på. Vi kan for eksempel se på ulike sentralmål. De sier noe om hvor midten av målingene er.

Sentralmål

Gjennomsnitt – er summen av alle målingene dividert med antall målinger.

I vårt eksempel blir det:

$$\frac{39 + 40 + 37 + 37 + 39 + 38 + 39 + 36 + 43 + 38 + 37 + 38 + 39 + 42 + 43 + 38 + 38 + 38 + 44 + 40 + 37 + 39 + 43}{23} \approx 39$$

Typetall – er den målingen det er flest av.

I vårt eksempel blir det 38.

Median – er det tallet som er i midten av målingene når de er stilt opp i stigende rekkefølge.

I vårt eksempel blir det

36, 37, 37, 37, 37, 38, 38, 38, 38, 38, 38, **39**, 39, 39, 39, 40, 40, 42, 43, 43, 43, 44

Snakk om

Vi har samlet inn informasjon om hvor mye 100 voksne får i årslønn. Tre av dem er direktører og tjener over 5 millioner kroner. De andre tjener mellom 200 000 og 700 000 kroner. Hvilket tall tror dere er størst, gjennomsnittslønna eller medianlønna? Hvorfor?

Lag en gjennomsnittskalkulator

- og en del av en statistikkutstilling

Oppgave

Dere skal lage en program som regner ut et gjennomsnitt. Dere skal bruke data samlet inn fra solfangeren fra opplegg 7 i programmet. Dere skal måle solfanger-temperaturen etter hvert minutt i 10 minutter. Programmet skal regne ut gjennomsnittet av målingene dine, og gjennomsnittet av temperaturstigningen etter 10 minutter for alle klassens grupper.

Lag en del av en statistikkutstilling ut fra første del av denne oppgaven. Det kan blant annet være en plakat, en tredimensjonal modell, en beskrivelse av hvordan dere gjorde det, bilder og tekst. Tenk gjennom hvordan dere skal utforme utstillingen for å få fram det dere gjorde på en god og effektiv måte.



Fase 1: Undersøk gjerne hvilke måter man kan lage en utstilling på.

Fase 2: Hvilke farger vil dere bruke i utstillingen? Skal dere bruke noen spesielle materialer? Vil dere bare bruke papir i flere farger, eller materialer som silkepapir, stoffbiter, garn, papp? Skal dere bygge en modell av noe? Lag en skisse over hvordan dere ønsker at utstillingen skal bli til slutt.

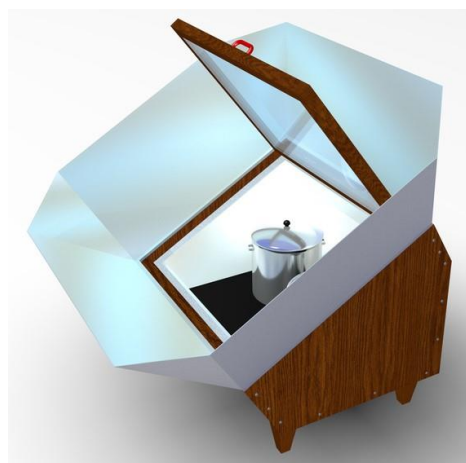
Fase 3: Lag første versjon av programmet for å beregne gjennomsnitt. Lag det dere trenger til utstillingen.

Fase 4: Test programmet ditt, får dere et svar som virker fornuftig? Spør andre i klassen om utstillingen deres er tydelig og god.

Fase 5: Sammenlign gjerne med de andre i klassen, er det noen som har en bedre utstilling? Hvorfor mener dere at den er bedre? Kan dere bruke noe av de samme grepene i deres utstilling?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få en best mulig utstilling. Gjør gjerne endringer i programmet deres om det trengs.

Fase 7: Pass på å ta vare på bilder og notater dere har gjort underveis, slik at dere kan vise hva dere har tenkt. I denne oppgaven går dokumenteringen ut på å lage selve utstillingen.



Opplegg 21 - Frekvenstabell og sektordiagram



Nå skal vi se på ulike måter å vise fram data, ved hjelp av frekvenstabell og sektordiagram.

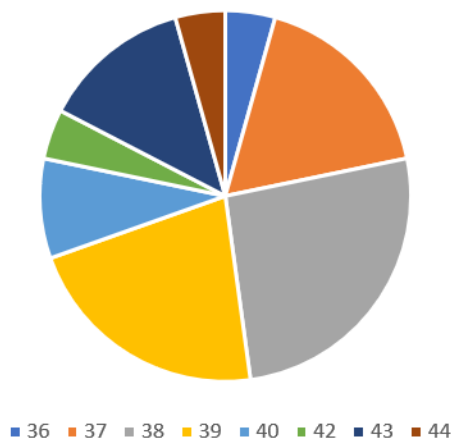
Frekvenstabell – Passer for å vise fram et datasett med et begrenset antall verdier. Vi teller opp hvor mange vi har av hver måling og ordner det i en tabell. Hvor mange vi har av hver måling kalles frekvensen til målingen. I vårt sko-eksempel er frekvensen til skostørrelse 39 lik 5. Frekvenstabellen for sko-eksempelen blir slik.

Måling	Frekvens
36	1
37	4
38	6
39	5
40	2
42	1
43	3
44	1

Sektordiagram – Passer for å vise fram et datasett med et begrenset antall verdier eller kategorier. Det passer i vårt sko-eksempel, der alle verdiene er fra 36 til 44. Det viser hvor stor andel hver måleverdi tilsvarer.

Vi bruker en frekvenstabell som utgangspunkt for å lage et sektordiagram. Man regner ut hvor stor brøkdel hver måling tilsvarer – hvor stor del av lærerne som har hver skostørrelse. Denne brøkdelen kan vi multiplisere med vinkelsummen for en sirkel, for da finner vi ut hvor mange grader av sirkelen hver skostørrelse tilsvarer. Da kan vi til slutt tegne sektordiagrammet, og merke av størrelsen på alle kakestykkene vi regnet ut størrelsen på, ved hjelp av en gradskive.

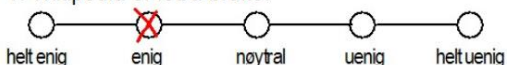
Måling	Frekvens	Brøkdel/andel	Vinkel
36	1	$\frac{1}{23} \approx 0,043$	$0,043 \cdot 360^\circ \approx 15^\circ$
37	4	$\frac{4}{23} \approx 0,174$	$0,174 \cdot 360^\circ \approx 63^\circ$
38	6	$\frac{6}{23} \approx 0,261$	$0,261 \cdot 360^\circ \approx 94^\circ$
39	5	$\frac{5}{23} \approx 0,217$	$0,217 \cdot 360^\circ \approx 78^\circ$
40	2	$\frac{2}{23} \approx 0,087$	$0,087 \cdot 360^\circ \approx 31^\circ$
42	1	$\frac{1}{23} \approx 0,043$	$0,043 \cdot 360^\circ \approx 15^\circ$
43	3	$\frac{3}{23} \approx 0,130$	$0,130 \cdot 360^\circ \approx 47^\circ$
44	1	$\frac{1}{23} \approx 0,043$	$0,043 \cdot 360^\circ \approx 15^\circ$
SUM	23	1	360



Spørreundersøkelser – Er en metode for å samle inn data som vi kan behandle statistisk. Når man utvikler en spørreundersøkelse, er det flere ting man bør tenke på.

- **Ja-siing:** Mennesker har en tendens til å være enig i utsagn de får presentert. Det er derfor best å spør om samme ting minst to ganger, men med motsatt vinkling.
 - Jeg liker å lese bøker. / Jeg misliker å lese bøker.
 - Jeg synes det er spennende å finne ut av ting jeg ikke vet. / Jeg er ikke spesielt nysgjerrig.
- **Skala:** Ofte bruker man det som kalles en likert-skala eller graderingsskala. Da har man gjerne svaralternativer som uenig, litt uenig, nøytral, litt enig og enig, i tillegg til vet ikke.

1. Wikipedia er lett å bruke.



Diskusjonsoppgave: Hvorfor tror dere at det er vanlig med likert-skala? Hva synes dere om eksempelet? Er det et godt eller dårlig eksempel?

Lag en spørreundersøkelse

Oppgave

Design en spørreundersøkelse som samler inn data, og samle resultatene i en frekvenstabell og et sektordiagram for minst ett spørsmål. Lag et program som beregner vinkler til sektordiagrammet ut fra tallene i frekvenstabellen. Bruk resultatene som en del av statistikkutstillingen deres med en beskrivelse av undersøkelsen og resultatet.

Fase 1: Undersøke og finne informasjon

Undersøk gjerne hvilke ulike typer data som kan være fornuftig å samle inn? Er det noe spesielt dere ønsker å finne ut av? Hva slags type spørsmål kan være bra å bruke?

Forslag til datainnsamling – ulike typer spørsmål

Hvor mange man har av noe (antall):

- Søsken, skjermer i husstanden, bøker i husstanden o.l.

Hvor fornøyd/misfornøyd man er med noe

- Fritidsaktiviteter, leksene, klimapolitikk o.l.

Hvor villig man er til å forandre på noe

- Antall flyreiser, kaste mindre mat, senke sitt forbruk o.l.



Fase 2: Idémyldring og planlegging

Ha en idémyldring for deg selv. Hvilke data ønsker du å samle inn? Hvorfor akkurat dette? Se etikkramma før dere bestemmer dere. →

Tegn gjerne en skisse over statistikkutstillingen før du diskuterer med de andre. Deretter må gruppa samlet bestemme hva dere skal spørre om og hvordan deres utstilling skal se ut.

Planlegg gjennomføringen

- Anslå hvor lang tid hver del tar.
- Har dere tid til å spørre alle spørsmålene?
- Hvis ikke, hva skal dere kutte ut?
- Hvordan skal dere dele inn svaralternativene?

Følg resten av fasene for å gjøre oppgaven ferdig.

Etikk

Pass på at du ikke spør om noe som oppleves som negativt eller sensitivt for noen personer.

Det må også være helt frivillig å svare på spørreundersøkelsen.

Kanskje kan undersøkelsen være anonym?

Hvorfor er dette så viktig? Hva har etikk med forskning å gjøre?

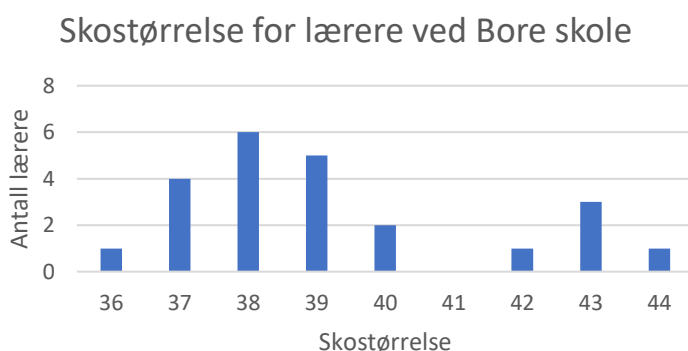


Opplegg 22 - Statistikk og diagrammer

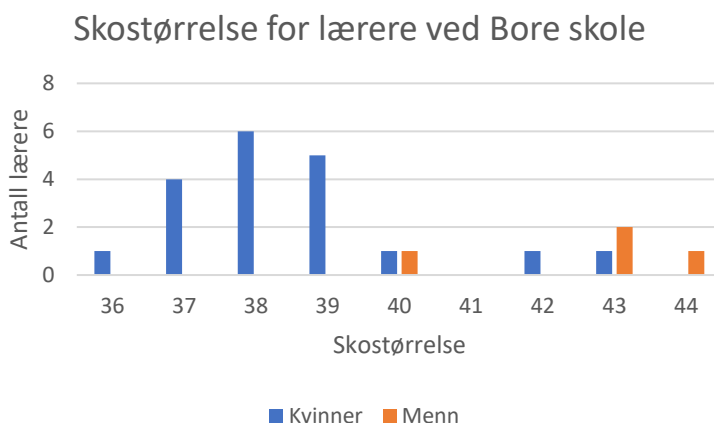


Nå skal vi se på ulike måter å vise fram data ved hjelp av diagrammer, og det finnes flere ulike typer diagrammer. Noen av de mest vanlige er:

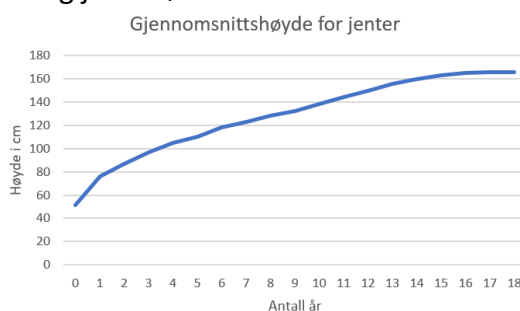
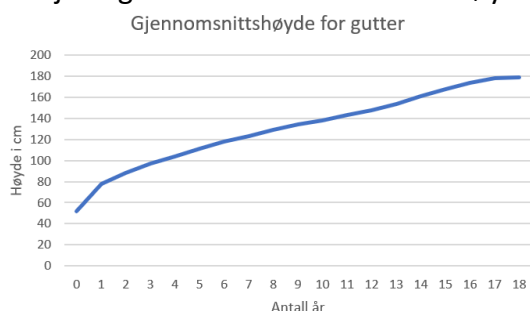
Stolpediagram (søylediagram) – Passer for å vise fram et datasett med et begrenset antall verdier eller kategorier. Det passer i vårt sko-eksempel, der alle verdiene er fra 36 til 44.



Gruppert stolpediagram – Passer for å sammenligne to ulike grupper som du har samme type data for. Dersom vi visste hvilke skostørrelser som tilhørte de kvinnelige og de mannlige lærerne, kunne vi laget et gruppert stolpediagram som dette:



Linjediagram (kurvediagram) – Passer godt for å vise hvordan noe endrer seg over tid. Det vet vi ikke noe om i vårt skostørrelse-eksempel, så vi må finne et annet eksempel. I disse to linjediagrammene ser vi hvordan høyden til gutter og jenter øker når de blir eldre.



Lag en statistisk fremstilling

- som skal brukes i en nyhetssak, og videre som en del av statistikkutstillingen deres

Oppgave

Bruk micro:biten til å samle inn data, og samle resultatene i en frekvenstabell og et diagram. Lag et nyhetsoppslag utfra målingene deres. Dette nyhetsoppslaget skal være en del av utstillingen deres.

Fase 1: Undersøke og innhente informasjon

Undersøk gjerne hvilke ulike typer data som kan være fornuftig å samle inn? Siden dere skal bruke micro:bit til datainnsamling, bør dere finne ut hva den kan brukes til å måle.

Fase 2: Idémyldre og planlegge

Ha en idémyldring for deg selv. Hvilke data ønsker du å samle inn? Hvorfor akkurat dette? Tegn gjerne en skisse over nyhetssaken før du diskuterer med de andre. Deretter må gruppa samlet bestemme hva dere skal måle og hvordan deres nyhetssak og utstilling skal se ut.

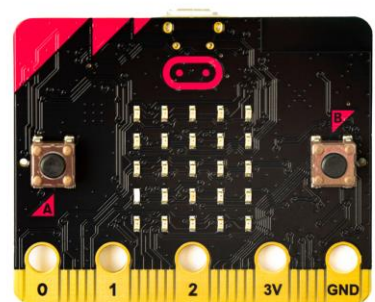
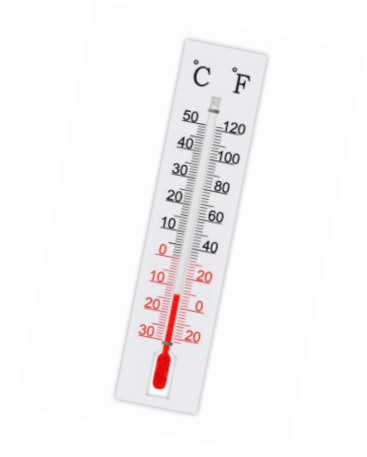
Fase 3: Lag første versjon av programmet for å gjøre målingene deres. Lag førsteutkast av nyhetssaken og det dere ellers trenger til utstillingen.,

Fase 4: Test programmet, får dere målinger som virker fornuftige? Spør andre i klassen om nyhetssaken deres er tydelig og god.

Fase 5: Sammenlign gjerne med de andre i klassen, er det noen som har en bedre nyhetssak? Hvorfor mener dere at den er bedre? Kan dere bruke noe av de samme grepene i deres sak?

Fase 6: Hopp gjerne tilbake til tidligere punkt, og gjør forandringer for å få en best mulig nyhetssak og utstilling. Gjør gjerne endringer i programmet deres om det trengs.

Fase 7: Pass på å ta vare på bilder og notater dere har gjort underveis, slik at dere kan vise hva dere har tenkt. I denne oppgaven går dokumenteringen ut på å lage selve nyhetssaken og utstillingen.



Opplegg 23 - Statistikk og «fake news»

Det finnes mange eksempler på dårlige eller villedende fremstillinger av data. Dette kan være gjort med vilje for å fremme et spesielt synspunkt, eller bare komme av dårlig utformede tabeller og diagrammer. Vi skal se på noen måter å lage dårlige og feil fremstillinger på.

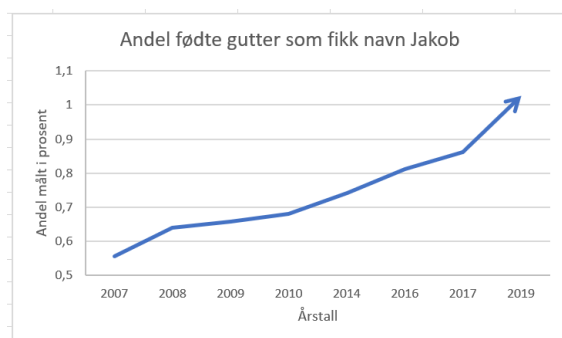
Kutte akser – fører til overdrevne forskjeller om man kutter på loddrett akse og feil inntrykk av utviklingen om man kutter på vannrett akse.

Kutte ut enkelte målinger – kan få statistikken til å vise noe helt annet enn den faktisk gjør.

Avslutte med pil – ser ut som om utviklingen fortsetter i samme retning, selv om vi ikke kan vite noe om dette.

Utelate relevant informasjon – gjør det vanskeligere å danne seg et helhetlig bilde av årsaker til utvikling.

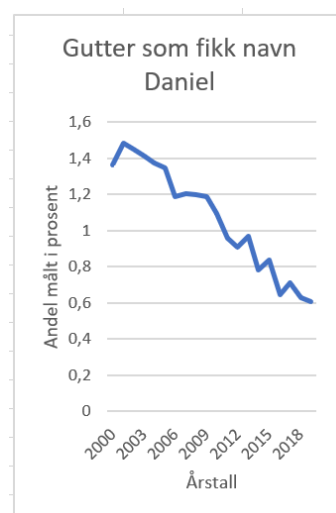
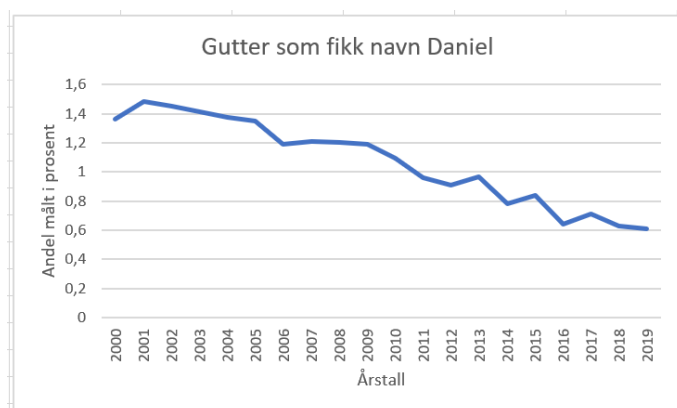
Ujevn avstand i aksene – kan føre til overdrevne eller underdrevne forskjeller. Da kan man få dataene til å gi inntrykk av akkurat hva man selv ønsker.



Data hentet fra: <https://www.ssb.no/statbank/table/10467/>

I diagrammet til venstre ser det ut til at andelen guttebabyer som blir kalt Jakob øker dramatisk og er på videre opp, siden vi avslutter med en pil oppover. Den loddrette akse er kuttet, og det gjør at det ser ut til at endringene er veldig store. Dette diagrammet gir inntrykk av at nesten alle guttebabyer vil få navn Jakob om noen få år, og det er ikke spesielt sannsynlig. Dersom vi ser på diagrammet til høyre, så ser det ikke like dramatisk ut. Her har vi med noen flere målinger som viser at utviklingen ikke er like dramatisk. Hvis du vil se hvordan utviklingen er for ditt navn, så kan du gå inn hit: <https://www.ssb.no/statbank/table/10467/>

Proporsjoner – dersom man bytter om på formatet, kan diagrammer gi inntrykk av større endringer enn det er grunn til. Standard er et liggende format. Se eksempel under:



Lag en villedende statistisk fremstilling

- som skal brukes i en falsk nyhetssak, og videre som en del av statistikkutstillingen deres

Oppgave

Bruk micro:biten til å samle inn data, og samle resultatene i et diagram, lag et falskt nyhetsoppslag utfra målingene deres. Dette falske nyhetsoppslaget skal være en del av utstillingen deres.

Fase 1: Undersøke og innhente informasjon

Undersøk gjerne hvilke ulike typer data som kan være fornuftig å samle inn? Siden dere skal bruke micro:bit til datainnsamling, bør dere finne ut hva den kan brukes til å måle. Husk at dere skal lage en falsk nyhetssak til slutt, så det kan være lurt å tenke på hvilken vinkling dere ønsker å ha i saken.

Fase 2: Idémyldre og planlegge

Ha en idémyldring for deg selv. Hvilke data ønsker du å samle inn? Hvorfor akkurat dette? Tegn gjerne en skisse over nyhetssaken før du diskuterer med de andre. Deretter må gruppa samlet bestemme hva dere skal måle og hvordan deres nyhetssak og utstilling skal se ut.

Fase 3: Lag første versjon av programmet for å gjøre målingene deres. Lag førsteutkast til nyhetssaken, og det dere ellers trenger til utstillingen.

Fase 4: Test programmet, får dere målinger som virker fornuftige? Spør andre i klassen om nyhetssaken deres er tydelig og god.

Fase 5: Sammenlign gjerne med de andre i klassen. Er det noen som har en bedre nyhetssak? Hvorfor mener dere at den er bedre? Kan dere bruke noe av de samme grepene i deres sak?

Fase 6: Hopp gjerne tilbake til tidligere punkt og gjør forandringer for å få en best mulig nyhetssak og utstilling. Gjør gjerne endringer i programmet deres om det trengs.

Fase 7: Pass på å ta vare på bilder og notater dere har gjort underveis, slik at dere kan vise hva dere har tenkt. I denne oppgaven går dokumenteringen ut på å lage selve nyhetssaken og utstillingen.



Opplegg 24 - Statistikk og spredningsmål



På Bore skole er det 23 lærere, og disse har alle notert hvilken skostørrelse de bruker. Resultatet ble: 39, 40, 37, 37, 39, 38, 39, 36, 43, 38, 37, 38, 39, 42, 43, 38, 38, 38, 44, 40, 37, 39 og 43.

Dette er ganske uoversiktlig, derfor finnes det flere måter å få bedre oversikt over innsamlede data på. Vi kan for eksempel se på ulike spredningsmål. De sier noe om hvor mye målingene varierer.

Spredningsmål

Variasjonsbredde – er forskjellen mellom høyeste og laveste måling. I vårt eksempel blir det:

$$44 - 36 = 8$$

Varsians – den sier noe om hvor langt fra gjennomsnittet alle målingene er. Først må vi regne ut gjennomsnittet, og det i opplegg 20, og fikk at gjennomsnittet var 39.

Deretter må vi ta differansen mellom målingen og gjennomsnittet, og opphøye det i andre. Dette multipliserer vi med frekvensen til målingen, for den sier hvor mange av målingen vi har. Til slutt summerer vi svarene vi får for alle målingene våre og dividerer med antall målinger. Det er lurt å bruke en frekvenstabell for å gjøre dette, ellers er det fort gjort å gå i surr. I vårt eksempel blir det slik:

Måling	Frekvens	$(\text{Måling} - \text{Gjennomsnitt})^2 \cdot \text{Frekvens}$
36	1	$(36 - 39)^2 \cdot 1 \approx 9$
37	4	$(37 - 39)^2 \cdot 4 \approx 16$
38	6	$(38 - 39)^2 \cdot 6 \approx 6$
39	5	$(39 - 39)^2 \cdot 5 \approx 0$
40	2	$(40 - 39)^2 \cdot 2 \approx 2$
41	0	$(41 - 39)^2 \cdot 0 \approx 0$
42	1	$(42 - 39)^2 \cdot 1 \approx 9$
43	3	$(43 - 39)^2 \cdot 3 \approx 48$
44	1	$(44 - 39)^2 \cdot 1 \approx 25$
SUM	23	115

Der vi til slutt må ta summen og dele på antall målinger: $\frac{115}{23} \approx 5$

Standardavvik – er kvadratroten av variansen. I vårt eksempel blir det:

$$\sqrt{5} \approx 2,2$$

Lag en standardavvikskalkulator

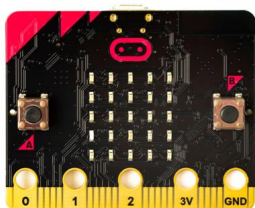
- og en del av statistikkutstillingen deres

Oppgave

Design en statistisk undersøkelse som samler inn data i naturfag, samle resultatene i en frekvenstabell og lag et program som regner ut standardavviket.

Lag en del av utstillingen med en beskrivelse av undersøkelsen og resultatet. Det kan være en plakate, en tredimensjonal modell, en beskrivelse eller noe helt annet. Tenk gjennom hvordan dere skal utforme utstillingen for å få fram det dere gjorde på en god og effektiv måte.

Fase 1: Undersøk gjerne hvilke ulike typer data som kan være fornuftig å samle inn. Vil dere bruke micro:bit til dette, eller en spørreundersøkelse? Hvordan skal denne delen av utstillingen se ut? Let etter inspirasjon fra forskjellige typer kilder.



Fase 2: Hvilke farger vil dere bruke i utstillingen? Skal dere bruke noen spesielle materialer? Vil dere bare bruke papir i flere farger, eller materialer som silkepapir, stoffbiter, garn, papp? Skal dere bygge en modell av noe? Lag en skisse over hvordan dere ønsker at utstillingen skal bli til slutt.

Fase 3: Lag første versjon av programmet for å beregne standardavvik. Lag eventuelt et program for å bruke micro:biten til å foreta målinger. Lag det dere trenger til utstillingen.

Fase 4: Test programmet, får dere et svar som virker fornuftig?



Fase 5: Sammenlign gjerne med de andre i klassen, er det noen som har en bedre utstilling? Hvorfor mener dere at den er bedre? Kan dere bruke noe av de samme grepene i deres utstilling?

Fase 6: Hopp gjerne tilbake til tidligere punkt, og gjør forandringer for å få en best mulig utstilling. Gjør gjerne endringer i programmet deres om det trengs.

Fase 7: Pass på å ta vare på bilder og notater dere har gjort underveis, slik at dere kan vise hva dere har tenkt. I denne oppgaven går dokumenteringen ut på å lage selve utstillingen.

Opplegg 25 - Statistikk i naturfag og samfunnsfag

- hvilke typer data finnes, og hvordan bør de representeres?

Bruk av statistikk i naturfag og samfunnsfag følger gjerne en inndeling som dette. Det betyr ikke at grensene for de ulike typene data er fullstendig klare, eller at noen av fremstillingene ikke kan brukes i begge fagområdene.

Naturfag

1) Data som skal «bli én verdi» og fordeler seg rundt denne.

Dette er typiske målinger vi gjør i naturfag, der vi holder alle andre variabler mest mulig fast. Da vil vi få forskjellige verdier på grunn av feilkilder og måleusikkerhet. Feilkildene får vi ikke gjort så mye med, men vi kan diskutere hva vi tror påvirker målingene våre. Måleusikkerheten kan vi finne ut av størrelsen på, ved å beregne gjennomsnitt og standardavvik for målingene våre.

Eksempler: Måling av lengden på en gjenstand, måling av farten til en bil, eller måling av vindstyrke.

2) Data som utvikler seg over tid.

Mange av målingene vi gjør i naturfag går ut på å måle en variabel som forandrer seg over tid. Disse målingene er det som regel nyttig å vise med et linjediagram.

Eksempler: Temperaturen i løpet av et døgn eller blodsukkernivået i løpet av en uke.



Samfunnsfag

1) Data som fordeler seg på forskjellige verdier med et begrenset antall verdier eller kategorier.

Dette er typiske målinger vi gjør i samfunnsfag gjennom spørreundersøkelser. Der er det gjerne lagt inn et begrenset antall svaralternativer med en likert-skala. Denne typen data passer veldig godt å vise fram i en frekvenstabell eller et stolpediagram. Et sektordiagram er fint å bruke i de tilfellene at fordelingen av de ulike verdiene er viktig i forhold til hverandre.

Eksempler: Karakterfordelingen på en prøve eller partitilhørighet for folk.

2) Data som har så mange forskjellige verdier at vi må dele dem inn i intervaller.

Mye av dataene vi samler inn i samfunnsfag har så mange forskjellige verdier at vi ikke kan ta med alle enkeltverdiene. Da lager vi grupper som inneholder flere av verdiene, vi lager intervaller. Denne typen data kan presenteres ved tabeller eller histogrammer.

Eksempler: Høydefordeling for gutter på 10 år eller lengden på skoleveien for en klasse.

3) Data som utvikler seg over tid.

Mye av dataene vi samler i samfunnsfag går ut på å måle én variabel som forandrer seg over tid. Disse målingene er det som regel nyttig å vise med et linjediagram, akkurat som i naturfag. I samfunnsfag blir disse dataene gjerne samlet inn i spørreundersøkelser, og ikke målt direkte med måleinstrumenter og sensorer.

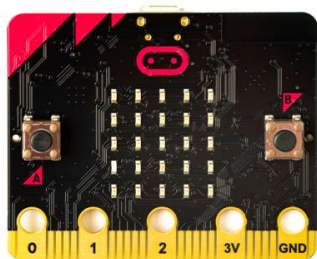
Eksempler: Antall personer som stemmer på et bestemt parti i løpet av 10 år, eller hvordan befolkningstallet endres over tid.

Lag en statistisk fargeleggingsbok

- Bruk micro:bit til å samle inn data

Oppgave

Design en datainnsamling i naturfag (forsøk) der dere bruker micro:bit for å samle inn data, og vis fram resultatene på en ryddig måte der dere selv velger hvilke tabeller og diagram dere vil bruke. Lag en side til en fargeleggingsbok med en enkel beskrivelse av undersøkelsen og resultatet. Denne skal siden skal også benyttes i statistikkutstillingen.



Fase 1: Undersøke og finne informasjon

Undersøk gjerne hvilke ulike typer data som kan være fornuftig å samle inn? Er det verdier som endrer seg over tid? En verdi som skal være konstant, som dere måler flere ganger? Eller er det verdier som bør deles inn i intervaller? Tenk gjennom hvordan dere best kan vise fram dataene dere samler inn – hvilken fremstilling som blir tydeligst.

Siden dere skal bruke micro:bit til dette, må dere finne ut hva den kan brukes til å måle. Husk at det går an å koble til andre sensorer og elektronikk.

Fase 2: Idémyldre og planlegge

Ha en idémyldring for deg selv. Hvilke data ønsker du å samle inn? Hvorfor akkurat dette? Tegn gjerne en skisse over siden til fargeleggingsboka før du diskuterer med de andre. Deretter må gruppa samlet bestemme hva dere skal måle og hvordan deres side i fargeleggingsboka skal se ut.

Planlegg gjennomføringen

- Anslå hvor lang tid hver del tar.
- Har du tid til å gjøre alle målingene?
- Hvis ikke, hva skal du kutte ut?
- Hvordan skal du dele inn målingene?

Følg resten av fasene i innovasjonsmetoden for å gjøre oppgaven.

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 9

Innhold

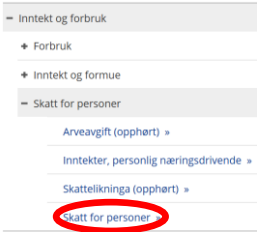
Kapittel 9 – Modellering og statistikk (10.trinn)

- Opplegg 26 – Last ned data fra statistisk sentralbyrå.....3
 - Lag figurer som illustrerer dataene du bruker.....4
- Opplegg 27 – Personlig økonomi.....5
 - Forsk på personlig økonomi.....6
- Opplegg 28 – Statistikk og korrelasjon.....7
 - Lag en villedende nyhetssak.....8

Opplegg 26 - Last ned data fra statistisk sentralbyrå

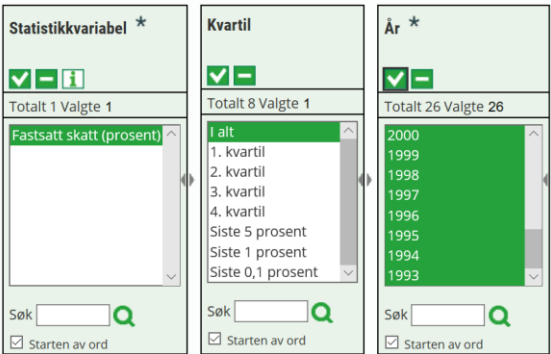
Mange forskere bruker data fra statistisk sentralbyrå (SSB) til å forske på mange forskjellige temaer. Det kan du også gjøre! Men da trenger du å vite hvordan du kan laste ned data fra SSB, og hvordan du kan bruke dem videre. Det får du en oversikt over her (eller en algoritme, om du vil):

1. Gå inn på www.ssb.no og trykk på statistikkbanken.
2. Velg hvilken kategori du vil finne data for, og så hvilken tabell du vil se på.

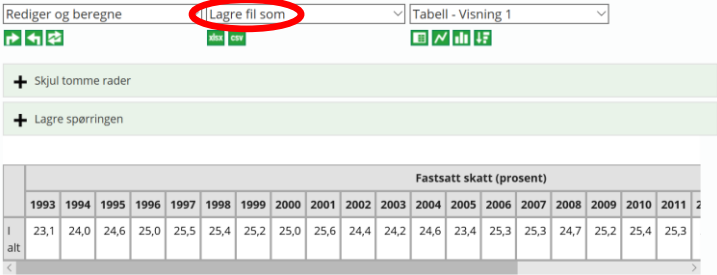


08564	Oversiktstall fra skatteoppgjøret for alle personer. Antall, mill. kr og gjennomsnitt (F)	2006 - 2018
06593	Skatt i prosent av bruttoinntekt for ulike bruttoinntektsgrupper (prosent)	1993 - 2018
05802	Formuesskatt (før nedsettelse), etter alder og kjønn. Antall personer og gjennomsnitt, alle personer	1997 - 2018

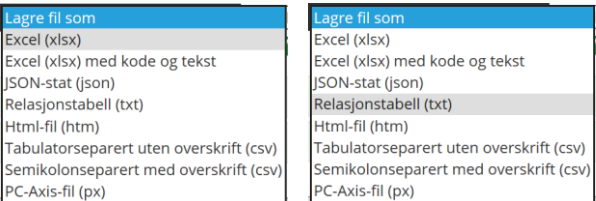
3. Velg hvilke variabler du vil ha med i tabellen din, og trykk på fortsett-knappen.
4. Da vises tabellen, og du kan laste den ned. Trykk på «Lagre fil som», og velg filtype.



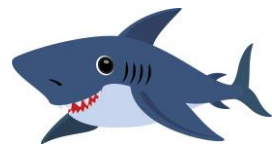
Antall valgte celler 26 (maksimalt tillatt er 300 000)
Presentasjon på skjermen er begrenset til 1 000 rader og 30 kolonner



5. Dersom du skal bruke data i
- Geogebra eller Excel → bruk .xlsx-filer
 - Python → bruk .txt-filer
6. Lagre filen i samme mappe som du skal lagre Geogebra, Excel- eller Pythonfila di.



Lag figurer som illustrerer dataene du bruker



- Bruk statistikk fra SSB

Oppgave

Finn et datasett hos statistisk sentralbyrå (se oppskrift på forrige side), og plott en graf av dataene. Lag en lineær matematisk modell for datasettet ditt og vurder om modellen er god. Til slutt skal dere lage figurer som illustrer hvilke data dere har brukt, som brukes i en utstilling sammen med grafen og modellen.

Fase 1: Tenk gjerne hvilke data som kan være fornuftig å bruke. Siden dere skal bruke SSB som kilde for data, må det være et datasett som finnes der.

Husk at dere skal lage figurer som illustrerer datasettet til slutt, så det kan være lurt å tenke på hvilken type figurer som kan passe til datasettene dere ser på.

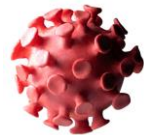
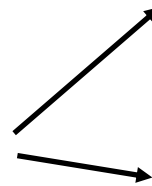
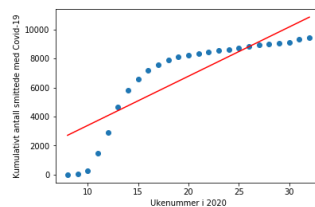
Kanskje kan en smeltende snømann illustrere utslipp av CO₂ eller stigende gjennomsnittstemperaturer?



Fase 2: Ha en idémyldring for deg selv. Hvilke data ønsker du å samle inn? Hvorfor akkurat dette? Tegn gjerne en skisse over figurene før du diskuterer med de andre. Deretter må gruppa samlet bestemme hvilke data dere skal bruke, og hvordan deres figurer og utstilling skal se ut.

Fase 3: Lag første versjon av grafen og den matematiske modellen. Lag figuren/figurene dere trenger til utstillingen.

Fase 4: Test grafen og modellen, får dere en modell som virker fornuftig? Spør andre i klassen om utstillingen deres er tydelig og god.



Fase 5: Sammenlign gjerne med de andre i klassen. Er det noen som har en bedre utstilling? Hvorfor mener dere at den er bedre? Kan dere bruke noe av de samme grepene i deres utstilling?

Fase 6: Hopp gjerne tilbake til tidligere punkt, og gjør forandringer, for å få en best mulig utstilling.

Fase 7: Pass på å ta vare på bilder og notater dere har gjort underveis, slik at dere kan vise hva dere har tenkt. I denne oppgaven går dokumenteringen ut på å lage selve utstillingen.

Opplegg 27 - Personlig økonomi



Personlig økonomi handler om de pengene folk tjener, låner og bruker på forskjellige varer og tjenester, og større ting som bolig og bil. Her kommer en liten forklaring på noen viktige begreper.

Innskudd – Penger vi har stående på konto i banken.

Lån – Vi kan låne penger fra banker eller kredittkortselskaper, og da må vi betale en viss prosentdel av lånesummen (det vi har lånt), i tillegg til selve lånesummen tilbake. Det finnes flere typer lån, blant annet serielån og annuitetslån.

Renter – En viss prosentdel av innskuddene eller lånene våre. Dersom det er renter på innskudd, er det penger vi får fra banken, men dersom det er renter på lån, er det penger vi må betale til banken.

Forbruk – Det vi kjøper. Et høyt forbruk betyr at vi bruker mye penger, mens et lite forbruk betyr at vi bruker lite penger.

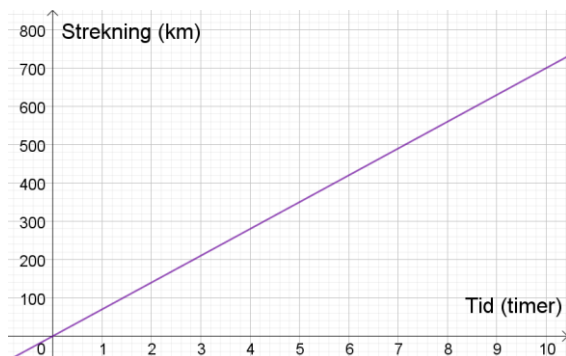
Lønn – De pengene vi tjener når vi jobber.

Skatt – Når vi får lønn, må vi betale en viss prosentdel av pengene til staten. Det kalles skatt.

Budsjett – En oversikt over hvor mye penger vi tror vi kommer til å bruke på forskjellige ting.

Regnskap – En oversikt over hvor mye penger vi faktisk har brukt på forskjellige ting.

Stigningstall til lineære funksjoner



Denne lineære grafen viser strekningen en bil har kjørt, som funksjon av tiden den bruker. Vi skal finne funksjonsuttrykket til grafen.

Alle lineære funksjoner kan skrives på formen

$$f(x) = a \cdot x + b$$

der a er stigningstallet og b er konstantleddet.

Stigningstallet forteller hvor bratt grafen er, og konstantleddet forteller hvor grafen skjærer y-aksen.

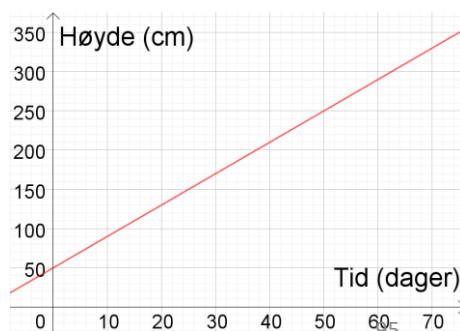
For vår graf blir konstantleddet $b = 0$ siden grafen krysser y-aksen i origo, altså når $f(x) = 0$. Når vi skal finne stigningstallet a må vi finne hvor mye grafen går oppover for hver enhet på x-aksen. Da velger vi to punkter på grafen vår, og regner ut hvor stor endringen er på y-aksen mellom de to punktene. Så regner vi ut hvor stor endringen er på x-aksen mellom de samme to punktene. Etterpå deler vi endringen på y-aksen på endringen på x-aksen, og da finner vi stigningstallet.

$$a = \frac{\text{endring i } y}{\text{endring i } x} = \frac{700 \text{ km} - 0 \text{ km}}{10 \text{ t} - 0 \text{ t}} = \frac{700 \text{ km}}{10 \text{ t}} = 70 \text{ km/t}$$

Stigningstallet til en lineær funksjon sier oss hvor mye y øker for hver økning i x . I en lineær graf med strekning og tid, slik som vi har, vil stigningstallet alltid gi oss farten, altså hvor mye strekningen i kilometer endrer seg per time. Funksjonsuttrykket til grafen vår blir dermed $f(x) = 70 \cdot x$.

Diskusjonsoppgaver

- Grafen til høyre viser høyden til en solsikke som funksjon av antall dager etter den ble satt ut.
 - Hva er funksjonsuttrykket for denne grafen?
 - Hva betyr stigningstallet for denne grafen?
 - For hvilke x -verdier tror du denne grafen stemmer med målingene?
- Hva betyr det at en graf har negativt stigningstall?
- En lineær graf viser antall firere som en funksjon av antall terningkast.
 - Hva blir stigningstallet (for store x -verdier)?
 - Hva betyr stigningstallet for denne grafen?



Forsk på personlig økonomi

- Bruk statistikk fra SSB

Oppgave

Finn data innen personlig økonomi som varierer over tid (<https://www.ssb.no/statbank/>), og utfør en lineær regresjon (gjærne med Python) for å finne en matematisk modell. Lag et produkt som viser hvilke data du har brukt, og hva du har gjort med dem. Husk begrunnelse for hvilke metoder du har brukt, og beskriv hva du har funnet ut.



Fase 1: Undersøk hvilke data som kan være fornuftig å bruke. Hvilke data har med personlig økonomi å gjøre? TIPS: Lån, renter, forbruk, arbeidsledighet og lønn.

Tenk gjennom hvordan dere best kan vise fram dataene dere samler inn – hvilken fremstilling som blir tydeligst. Vil dere lage en plakate, en film, en utstilling, eller noe annet?

Fase 2: Ha en idémyldring for deg selv. Hvilke data ønsker du å bruke? Hvorfor akkurat dette? Tegn gjerne en skisse over produktet før du diskuterer med de andre. Deretter må gruppa samlet bestemme hvilke data dere skal bruke og hvilket produkt dere skal lage.

Planlegg gjennomføringen, og anslå hvor lang tid hver del tar.

Fase 3: Lag første versjon av grafen for den matematiske modellen. Lag det dere trenger til produktet deres.

Fase 4: Test produktet ved å spørre de andre i klassen om produktet deres er informativt og tydelig.

Fase 5: Sammenlign gjerne med de andre i klassen. Er det noen som har brukt interessante data eller gjort noe spennende med produktet sitt? Kan dere bruke noe av de samme grepene til deres produkt?

Fase 6: Hopp gjerne tilbake til tidligere punkt, og gjør forandringer for å få en best mulig produkt. Gjør gjerne endringer i grafen deres, om det trengs.

Fase 7: Pass på å ta vare på bilder og notater dere har gjort underveis, slik at dere kan vise hva dere har tenkt. I denne oppgaven går dokumenteringen ut på å lage selve produktet.

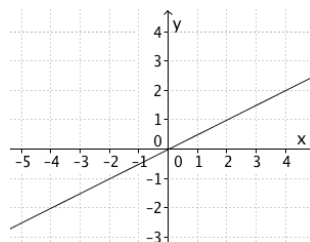


Oppgave

Hva er stigningstallet for den lineære funksjonen din?

Forklar hva stigningstallet ditt betyr.

Forklar hva resultatet hadde blitt om stigningstallet ditt ble mindre.



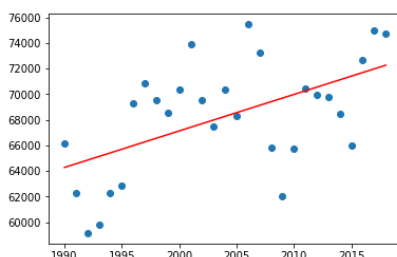
Opplegg 28 - Statistikk og korrelasjon

Korrelasjon betyr ikke nødvendigvis at vi har kausalitet!

Dersom vi har to datasett (for to variabler), kan vi finne ut hvor godt de korrelerer med hverandre. Altså om de varierer på samme måte. Det vil si at dersom den ene variabelen øker, så øker den andre variabelen tilsvarende, eller kanskje den minker tilsvarende. Vi kan si at korrelasjon er det samme som samvariasjon.

For å sjekke hvor godt to variabler korrelerer, utfører vi en lineær regresjon. Da kan vi få beregnet korrelasjonen, det er nemlig et mål på hvor store avstandene er mellom regresjonsgrafen og hvert av punktene i grafen. Vi regner ikke dette for hånd, men geogebra eller Python kan gjøre det for oss.

Litt forenklet vil geogebra eller Python i en lineær regresjon lage en rett linje som gjør at de samla avstandene mellom alle punktene og selve regresjonslinja blir minst mulig. Altså at summen av avstandene mellom de blå punktene og den røde lineære grafen blir minst mulig.

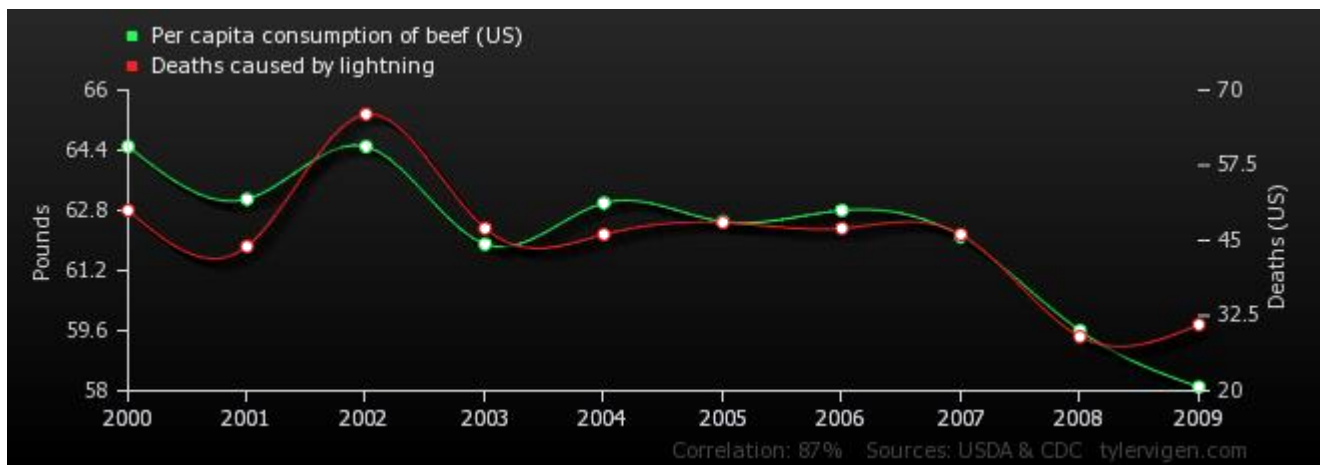


Hva er kausalitet?

Det betyr at det er en årsak-virkning sammenheng mellom variabler. Dersom man endrer på én variabel, så vil den andre variabelen også endre seg. Et eksempel på en kausal sammenheng kan være at antall timer du sover om natta påvirker hvor trøtt du er neste dag. Et annet ord for kausalitet er årsakssammenheng.

Man kan sammenligne to variabler som varierer over tid, og sjekke om de varierer på samme måte. For eksempel kan man velge to variabler som det er gjort målinger av årlig i perioden 1990 til 2020. For å sjekke hvor godt de to variablene korrelerer, velger man én av variablene som x-verdi og den andre som y-verdi. Det er viktig å passe på at x-verdien og y-verdien er for samme årstall. Da kan man utføre en lineær regresjon på x- og y-verdiene og finne korrelasjonen.

Det er ikke dette som er gjort med de to grafene under. Her er begge variablene tegnet inn som funksjon av årstall, slik at vi har to separate funksjoner. Men vi kan se at begge grafene har lignende form.



Korrelasjonskoeffisient: 0,87

Kilde: www.tylervigen.com

Korrelasjonskoeffisienten – kan variere mellom -1 og 1.

Dersom den er 0, eller nær 0, betyr det at vi har ingen eller liten korrelasjon.

Dersom den er nær -1, minker den ene variabelen når den andre øker.

Dersom den er nær 1, vil begge variablene øke/minke på samme måte.

Det er tilsvarende for R^2 , men den kan ikke være negativ.

Noen mulige årsaker til korrelasjon:

- En variabel kan direkte forårsake eller avhenge av verdien til en annen variabel.
- En variabel kan være løst knytta til en annen variabel.
- To variabler kan begge avhenge av en ukjent tredje variabel.
- Det kan være tilfeldig – størst sjanse for at dette skjer ved få målinger.

Lag en villedende nyhetssak

- ved bruk av korrelasjon

Oppgave

Finn to datasett hos statistisk sentralbyrå (<https://www.ssb.no/statbank/>), og finn korrelasjonen ved bruk av Geogebra eller Python. Lag et nyhetsoppslag som spekulerer i kausalitet (at den ene variabelen er årsak til at den andre endres).

Fase 1: Undersøk hvilke data som kan være fornuftig å bruke. Hvilke data kan tenkes å variere på samme måte over tid? Husk at dere skal lage en falsk nyhetssak til slutt, så det kan være lurt å tenke på hvilken vinkling dere ønsker å ha i saken.

Fase 2: Ha en idémyldring for deg selv. Hvilke data ønsker du å bruke? Hvorfor akkurat dette? Tegn gjerne en skisse over nyhetssaken før du diskuterer med de andre. Deretter må gruppa samlet bestemme hvilke data dere skal bruke og hvordan deres nyhetssak skal se ut.

Fase 3: Lag første versjon av grafen for den matematiske modellen. Hva blir korrelasjonen mellom variablene? Lag nyhetssaken basert på korrelasjonen.

Fase 4: Får dere en korrelasjon som virker fornuftig? Spør andre i klassen om nyhetssaken deres er tydelig og god.

Noen eksempler på villedende nyhetssaker og konspirasjonsteorier

- basert på korrelasjoner

Ny studie:

Mye egg koblet til hjertesykdom og død

Kanskje er ikke egg så ufarlig likevel?

[Ny studie: Mye egg koblet til hjertesykdom og død \(forskning.no\)](#)

Folk som drikker brus – både sukkerfri og vanlig – lever kortere

Brus hører ikke til i hverdagen, sier forsker.

[Folk som drikker brus – både sukkerfri og vanlig – lever kortere \(forskning.no\)](#)

Håndskrift hjelper hukommelsen

Barn som lærer seg å skrive for hånd, har lettere for å huske og utvikle egne ideer enn barn som lærer seg å skrive via et tastatur, viser ny forskning.

[Håndskrift hjelper hukommelsen \(aftenposten.no\)](#)

One is a 5G Map, One is the CoronaVirus Map in the USA
Always a bigger picture 🕵️



Theodore R. Smith på Twitter: «Correlation ≠ Causation. Tho this probably will correlate with population density which will correlate with mass infections, and 5G could be ruining people's immune systems. <https://t.co/r5as5Wfm2e> / Twitter

Fase 5: Sammenlign gjerne med de andre i klassen. Er det noen som har en bedre nyhetssak? Hvorfor mener dere at den er bedre? Kan dere bruke noe av de samme grepene i deres sak?

Fase 6: Hopp gjerne tilbake til tidligere punkt, og gjør forandringer for å få en best mulig nyhetssak. Gjør gjerne endringer om det trengs.

Fase 7: Pass på å ta vare på bilder og notater dere har gjort underveis, slik at dere kan vise hva dere har tenkt. I denne oppgaven går dokumenteringen ut på å lage selve nyhetssaken.

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 10

Innhold

Kapittel 10 – Naturfag og ikke-lineær modellering (10.trinn)

- Opplegg 29 – Varmer, isolasjon og energieffektivitet.....3
 - Lag et energieffektivt hus.....4
- Opplegg 30 – Populasjoner og populasjonsvekt.....5
 - Lag en øy.....6
- Opplegg 31 – Vaksiner og immunforsvar.....7
 - Programmer og strikk et smittetall.....8
- Opplegg 32 – Forbrenningsreaksjoner.....9
 - Lag en utskytningsmekanisme.....10

Opplegg 29 - Varme, isolasjon og energieffektivitet

Varme er energi som blir overført fordi noen ting har forskjellig temperatur. Ingen gjenstander kan inneholde varme, det er kun noe som kan bli overført fra et system til et annet. Et slikt system med høy temperatur kan for eksempel være deg selv, og et system med lavere temperatur kan være lufta rundt deg. Da vil det bli overført varme fra deg til lufta rundt deg. Som oftest ønsker vi ikke å overføre så mye varme, da minker temperaturen vår og vi fryser. Da kan vi sørge for å bruke en form for

isolasjon. Den vanligste isolasjonen for oss mennesker, er klærne vi bruker. De forhindrer at temperaturen vår synker for mye.



Grunnen til at klær virker isolerende, er at de inneholder mye stillestående luft. Luft leder varme veldig dårlig, og virker dermed som en varmeisulator. Men om lufta beveger seg, så isolerer den ikke spesielt godt. Tenk bare på hvor mye kaldere det kjennes ut ute når det blåser kraftig sammenlignet med når det ikke er vind. De fleste metaller leder varme veldig godt. Tenk bare på grytene dere har hjemme, de er typisk laget av metall siden vi ønsker at maten skal bli oppvarmet raskest mulig.

Hvor gode varmeisolatorer materialer er, henger sammen med deres varmeledningsevne. Varmeledningsevnen sier noe om hvor godt et materiale leder varme. Dersom et materiale har god varmeledningsevne, er det en dårlig varmeisulator.

I en termos er hele poenget å sørge for at minst mulig varme slipper ut til omgivelsene. Vanligvis består termosar i hovedsak av glass, plast og stål.

Snakk om

Har du hørt om leder og isolator i andre sammenhenger enn varmeledning?



Papp



Plastfolie



Aluminiumsfolie



Glass



Garn



Plast



Kakelys

Snakk om

Hvilken funksjon kan disse forskjellige materialene ha?

Er det noen som isolerer bedre enn andre?

Er det noen andre grunner til å bruke disse materialene i et energieffektivt hus enn varmeledningsevne?



Foto: Takako Picture Lab / Shutterstock.com

Et passivhus er et hus som bruker lite energi til oppvarming sammenlignet med vanlige hus. Det er med andre ord veldig energieffektivt. For å kalles et passivhus må det tilfredsstille fire krav:

- 1) Lite varmetap.
- 2) Lite oppvarmingsbehov.
- 3) Minst halvparten av varmtvannet skal varmes opp uten bruk av strøm eller fossile brensler.
- 4) Ventilasjonsanlegget og materialene må være energieffektive.

Energiproduksjon

Siden energi ikke kan oppstå eller forsvinne, bare omdannes til andre typer energi, så er all elektrisitetsproduksjon egentlig bare en omforming. Ved vannkraft omgjøres stillingsenergien til vannet til elektrisitet, ved vindkraft omgjøres bevegelsesenergien til lufta (vinden) til elektrisitet. For fossile brensler er det kjemisk energi som omgjøres til elektrisitet eller varme, men da frigis det store mengder CO₂. Dette fører til globale klimaendringer. Dersom vi klarer å bruke mindre energi, blant annet ved å ha mest mulig energieffektive hus, vil vi kunne redusere våre CO₂-utslipp.

Lag et energieffektivt hus

- mål temperaturen med radiostyrt micro:bit

Oppgave

Lag et mest mulig energieffektivt hus. Dere skal brenne ned ett kakelys inni huset og så måle hvordan temperaturen synker etterpå. Mål hvor mye temperaturen endrer seg på 60 minutter med å bruke en micro:bit inni huset som sender temperaturen til en annen micro:bit.

Fase 1: Hvordan virker et energieffektivt hus? Dere kan undersøke litt mer om teorien bak passivhus.

Fase 2: Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan nyttige forslag bli avleid for tidlig.

1. Tenk selv først og tegn skisser fra forskjellige vinkler.
2. Forklar ideen din for de andre på gruppa. Bruk gjerne skissene i forklaringen.
3. Hele gruppa diskuterer de ulike ideene, og lager en felles hypotese for bygging.

Fase 3: Gjennomfør planen deres for å lage huset og lag programmet for å måle temperaturen. Se tips under.



Lag en hypotese

Lag en hypotese som beskriver hvordan det mest energieffektive huset ser ut.

Micro:bit nummer 1

Lag et program som sender temperaturen ved å bruke følgende kommandoer:

```
radio.send_bytes(x)    radio.on()
radio.config(channel=40)
temperatur = temperature() while True:
    sleep(500)    from microbit import *
import radio    x = str(temperatur)
```

Micro:bit nummer 2

Lag et program som mottar og skriver ut temperaturen ved å bruke følgende kommandoer:

```
x = radio.receive_bytes()    sleep(500)
radio.on()    import radio    print(x)
while True:    radio.config(channel=40)
from microbit import *
```

Fase 4: Test hvor mye temperaturen endrer seg.

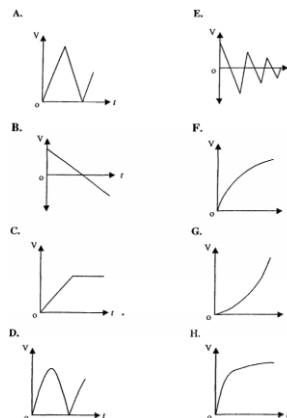
Fase 5: Sammenlign resultatene med andre i klassen. Fikk noen andre mindre temperaturnedgang?

Fase 6: Gå tilbake til de andre fasene for å gjøre planlagte forbedringer.

Fase 7: Gjennomfør de siste målingene og dokumenter prosessen på en valgfri måte.

Modelleringsoppgaver

1. Mål temperaturen hvert minutt i 30 minutter.
2. Plott alle de målte verdiene i Geogebra eller med Python
3. Finn en matematisk modell ved å foreta en regresjon for de målte dataene.
4. Hvordan kan dere avgjøre hvilken modell som passer best?
5. Passer modellen bra med målingene deres?
6. Hva kan gyldighetsområdet til modellen kan være? Begrunn svaret.



Opplegg 30 - Populasjoner og populasjonsvekst

Hva er en populasjon?

Det er alle individene av en bestemt art som lever innenfor et avgrenset område eller økosystem. Det kalles også en bestand.

Hva er et økosystem?

Det består av alle populasjonene innenfor et begrenset område, samt de abiotiske faktorene i området. Noen eksempler kan være en myr, en skog eller et korallrev.

Ekspontialfunksjoner og prosentvis endring

En eksponentialfunksjon kan alltid skrives på denne formen $f(x) = a \cdot b^x$ der a er startverdien og b er vekstfaktoren som alltid er et positivt tall.

Vi har en populasjon av rådyr og vi vet at den øker med 20 % årlig. Hva blir vekstfaktoren til denne populasjonen?

Vi begynner med å regne om 20 % til desimaltall.

$$20 \% = \frac{20}{100} = 0,20$$

Siden hele den opprinnelige populasjonen alltid tilsvarer 100 %, må vi legge til dette

$$100 \% + 20 \% = 1,00 + 0,20 = 1,20$$

I dette tilfellet blir vekstfaktoren til rådyrpopulasjonen 1,20.

Diskuter

Hva blir vekstfaktoren om populasjonen minker med 20 % årlig?

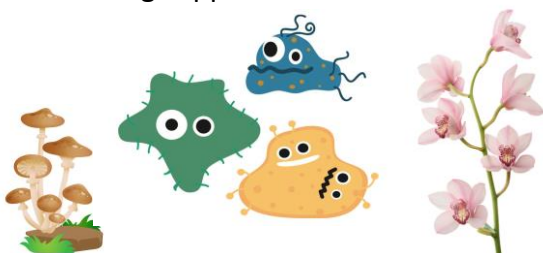


Startverdien er antall rådyr i den opprinnelige populasjonen. Dersom vi har 250 i en rådyrpopulasjon med en årlig vekst på 20 % vil funksjonen vår se slik ut:

$$f(x) = 250 \cdot 1,20^x$$

Hva er en biotisk faktor?

Det er alle de levende organismene i et økosystem, slik som alle planter, dyr, bakterier og sopp.



Hva er en abiotisk faktor?

Det er alle de ikke-levende omgivelsene som påvirker organismene som lever der, slik som jordsmonn, pH-verdi, klima og temperatur.



Lag en øy!

- og programmer populasjonsveksten

Oppgave

Lag en modell av en liten øy, med en oversikt over de forskjellige biotiske og abiotiske faktorene. Tenk deg at det ble satt ut et kaninpar på øya. Lag et program som beregner hvor mange kaniner det blir på øya hvert år etterpå, for de 10 første årene. Lag en graf som viser populasjonsveksten for kaninene.

Fase 1: Hvilke materialer skal dere bruke for å lage øya? Hvilke abiotiske/biotiske faktorer skal dere illustrere?

Fase 2: Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan morsomme forslag bli avfeid for tidlig.

1. Tenk selv først og tegn gjerne skisser.
2. Forklar ideen din for de andre på gruppa.
3. Hele gruppa diskuterer de ulike ideene, og lager en felles plan for øymodellen.



Fase 3: Lag en modell av øya deres, med biotiske og abiotiske faktorer. Lag programmet som beregner populasjonsveksten. Spør gjerne læreren om ark med tips til programmeringen.

Puslespill-programmering

```
print(populasjon)

populasjon = populasjon * vekstfaktor

for i in range(0,10):
    vekstfaktor = 1 + økning/100
```

økning = 20 populasjon = 1000

Fase 4: Hvordan skal dere teste modellen av øya deres? Kanskje kan det være fornuftig å sammenligne resultatene med andre i klassen.

Fase 5: Er det noe dere vil endre?

Fase 6: Gå tilbake til de andre fasene for å gjøre planlagte forbedringer på modellen deres.

Fase 7: Dokumentér det dere har gjort med en liten film og begrunn valgene deres. Vis fram resultatet for resten av klassen med en liten utstilling.

Refleksjonsoppgaver

1. Hva ville skjedd med de forskjellige artene på øya om det flyttet mennesker dit?
2. Menneskene ville trenge å dyrke mat, hvilke konsekvenser ville dette få for det biologiske mangfoldet på øya?
3. Hva er viktigst av at mennesker kan benytte naturressursene, og å bevare det biologiske mangfoldet?
4. Hva om menneskene ville drive gruver eller lage et kraftverk? Hvilke konsekvenser ville det få?

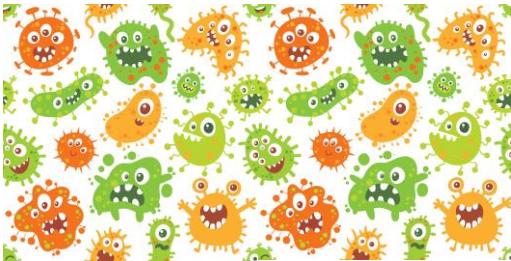
Diskuter

1. Hvilken type matematisk modell har vi brukt for populasjonen på øya?
2. Er denne realistisk?
3. Hvorfor/hvorfor ikke?
4. Klarer du å finne en modell som er mer realistisk?

Opplegg 31 - Vaksiner og immunforsvar



Hvordan vet kroppen at noen bakterier og virus er farlige, men lar yoghurtbakteriene være? Hva er det som gjør at kroppen aktiverer immunforsvaret til den store sykdomsjakten? Alle de farlige virusene og bakteriene har noen kjennetegn som gjør det mulig for immunforsvaret å oppdage dem. Immunforsvaret har en slags hukommelse som gjør at når det blir aktivert, så husker det kjennetegnene i lang tid, slik at det kan angripe sykdommen neste gang vi blir smitta. Vaksiner inneholder viruset, eller bakterien vi skal vaksineres mot, men de er enten døde eller svekket slik at de ikke kan gi sykdom. Når de reduserte organismene kommer inn i kroppen, vil de aktivere immunforsvaret, slik at immunforsvaret husker kjennetegnene til akkurat den organismen. Da kan kroppens immunforsvar effektivt kjempe mot sykdommen, dersom man blir smitta senere.



Hva er smittetall?

Det forteller hvor mange som gjennomsnittlig blir smittet av én person mens den er syk. Smittetall er det samme som vekstfaktor i en eksponential matematisk modell.

Vaksinemotstand mot meslingvaksinen (og andre)

Det finnes personer som er motstandere av vaksiner, og argumenterer for at andre heller ikke skal vaksineres. Dette er hovedgrunnen til at sykdommer som meslinger har fått utbrudd der sykdommen tidligere var utslettet.

Det finnes gjerne tre hovedargumenter de bruker:

- Meslingvaksinen forårsaker autisme.
- Meslinger er bra for immunforsvaret.
- Vaksinene inneholder farlige stoffer som aluminium, kvikksølv og formaldehyd.

En studie som ble utført av Andrew Wakefield, og publisert i 1998 hevdet at det var en forbindelse mellom meslingvaksinen og autisme. Studien ble tilbakevist og trukket i 2004. Studien var basert på forfalskede data, og Wakefield ble fradømt retten til å praktisere som lege.

Meslinger er en sykdom som kan få dødelig utfall. Det er noen som får milde bivirkninger av vaksinen, men dette er mye mindre alvorlige bivirkninger enn de som får meslinger. Det finnes mange studier som viser at immunforsvaret virker dårligere i lang tid etter sykdommen. I slutten av 2019 kom det en studie som viste at et utbrudd av meslinger får kroppens immunforsvar til å «glemme» andre sykdommer den har hatt før og miste immuniteten sin.

Aluminium finnes i enkelte vaksiner, men i mye mindre dose enn at det kan utgjøre en risiko. Det blir brukt for å gjøre noen vaksiner mer effektive. Kvikksølv (thiomersal) har blitt fjernet fra barnevaksiner, selv om det ikke er vist noen helserisiko når det ble brukt. Formaldehyd har vi i blodet fra før, i langt større mengde enn det som blir tilført ved en vaksine.

Kildekritikk

Sjekk påstandene og svarene på påstandene om vaksinemotstand fra forskjellige internettkilder.

Sjekk hvilke personer og institusjoner som står bak de forskjellige synspunktene.

1. Hvem mener du er mest troverdige?
2. Hvilke kilder oppgir de for sine påstander?
3. Stemmer det de skriver med det du kan om immunforsvaret fra før?



Oppgave

1. Hvor stor er sannsynligheten for de fire forskjellige alvorlige bivirkningene fra figuren over?
2. Hvor stor er sannsynligheten for å få meslinger i Norge dersom du ikke er vaksinert?

Programmér og strikk et smittetall

Oppgave

Lag et program som beregner hvor mange personer som blir smitta ved et valgfritt smittetall i 10 smittegenerasjoner. Start med 10 masker og la hver maske tilsvare en person. Hver nye rad tilsvarer en ny smittegenerasjon. Strikk smittelappen for de 10 første smittegenerasjonene. La hver gruppe lage en blomst av smittelappene de lager (en «smitteblomst»).

Fase 1: Hvordan kan man bruke programmering til å beregne smittespredning? Hva betyr smittegenerasjon? Vet du hvordan man kan øke antall masker når man strikker? Hvis det er noe i oppgaven du ikke forstår, prøv å finne det ut. Diskuter gjerne med andre på gruppa, søk på internett eller spør hva læreren mener.

HINT

Smittetall tilsvarer vekstfaktor i eksponentialfunksjoner. Har dere programmert vekst for eksponentialfunksjoner før?

Fase 2: Idémyldre og planlegge

Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan nyttige forslag bli avfeid for tidlig.

1. Tenk selv først, og tegn gjerne skisser av både smittelappene og smitteblomsten.
2. Forklar og vis ideen din for de andre på gruppa.
3. Hele gruppa diskuterer de ulike ideene, og lager en felles plan for smitteblomsten.

Hva er det som gjør at smittelappene vil se forskjellige ut? Hvilke typer smittelapper vil bidra til en fin smitteblomst? Hvor mange smittelapper skal gruppa lage? Hvilke farger og hvilket type garn skal dere bruke? Skal alle bruke de samme materialene og fargene?



Fase 3: Programmering av populasjonsvekst og lage blomst

Lag en strikkeoppskrift ved hjelp av programmering og strikk minst en lapp hver. Gjennomfør planen deres for å strikke populasjonsendringene, og lag en plakate med blomsten deres.

Fase 4: Siden det er vanskelig å teste hvor god smitteblomsten er, så kan dere hoppe over denne fasen.

Fase 5: Sammenlign smitteblomsten med de andre smitteblomstene i klassen. Er det noe dere vil endre på?

Fase 6: Gå tilbake til de andre fasene for å gjøre forbedringene dere kom fram til i fase 5.

Fase 7: Dokumenter det dere har gjort og begrunn valgene deres med en film- eller lydfil. Presenter gjerne smitteblomsten for resten av klassen og ha blomstene på utstilling.



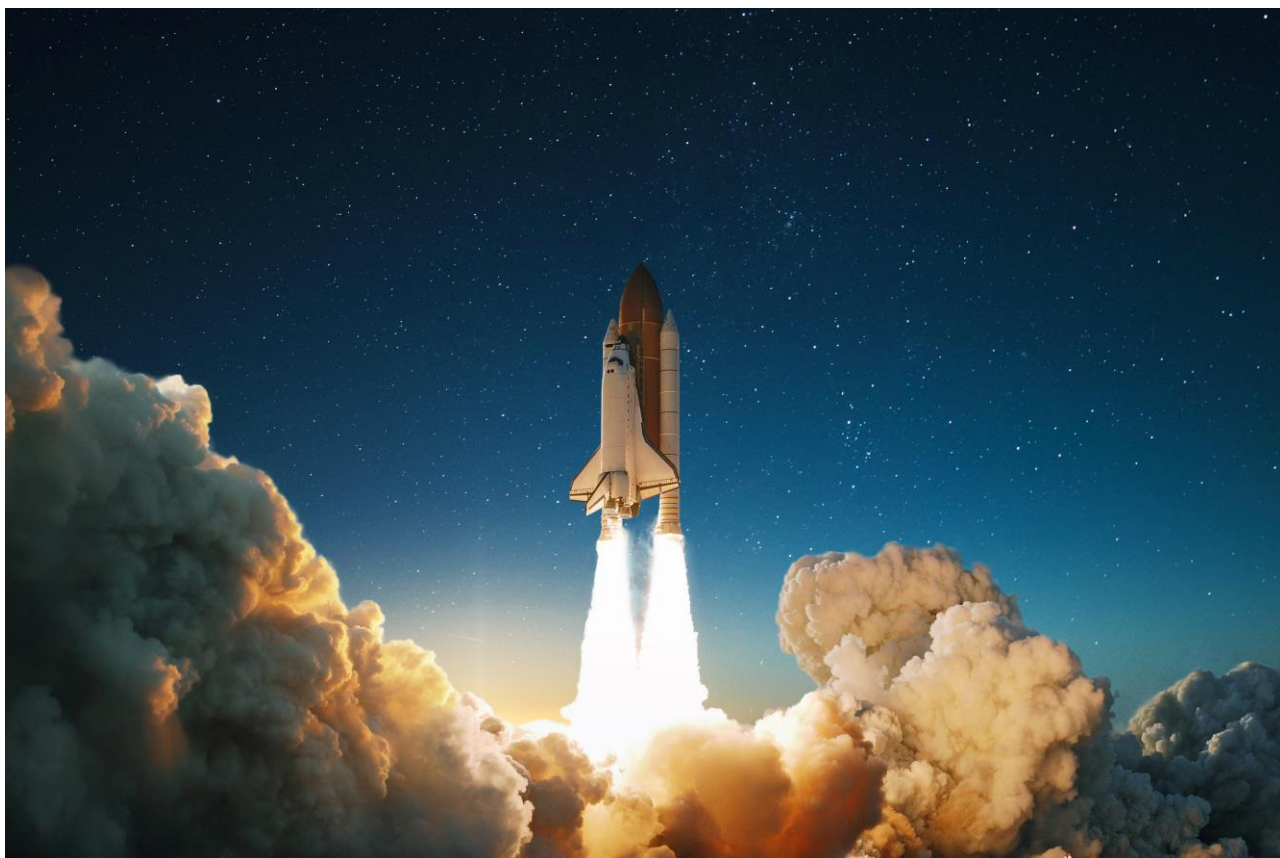
Diskuter

1. Hvordan henger smittetallet sammen med smittelappene?
2. Hvordan kan man bruke smittelappene til å se hvor mange personer som blir smittet av en sykdom?
3. Hva skjer med antall smittede dersom smittetallet blir dobbelt så stort?

Opplegg 32 - Forbrenningsreaksjoner

Hva er en forbrenningsreaksjon?

En reaksjon mellom et stoff og oksyngengass, og som frigir energi.



For at en rakett skal kunne skytes opp i verdensrommet, må den gjøre om en energiform til kinetisk energi (bevegelsesenergi). Dagens raketter bruker kjemisk energi lagret i et drivstoff og et **oksiderende** stoff i fast form, flytende form eller en blanding av disse formene. Disse stoffene blandes og antennes, slik at vi får en forbrenningsreaksjon. I disse reaksjonene frigis det store nok mengder energi til å gi raketten stor nok fart til å nå verdensrommet.

I organiske forbrenningsreaksjoner er det et organisk stoff som reagerer med oksygen. Dersom man har god tilgang til oksygen, vil man ha en fullstendig forbrenning. Da blir det frigitt mye energi, vann og karbondioksid. Om det ikke er nok oksygen, får man en ufullstendig forbrenning. Da frigis bare litt energi, vann og karbonmonoksid. Karbonmonoksid er en luktfri og usynlig gass som er giftig å puste inn.

Diskusjonsoppgaver

1. Hvilke energiformer har vi i en rakettoppskyting?
2. Hva har forbrenningsreaksjoner med klimaendringer å gjøre?
3. Skjer det forbrenningsreaksjoner i menneskekroppen?

Fullstendig forbrenning: Karbonholdig stoff + $O_2 \rightarrow H_2O + CO_2 + \text{ENERGI}$

Ufullstendig forbrenning: Karbonholdig stoff + $O_2 \rightarrow H_2O + CO + \text{energi}$

Lag en utskytningsmekanisme

Oppgave

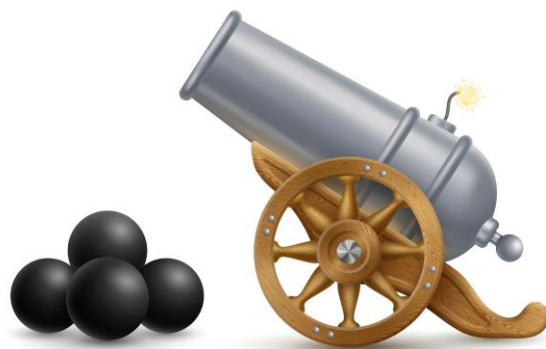
Lag en utskytningsmekanisme som skyter en kule lengst mulig. Den må bruke en servo som kan stille inn vinkelen på utskytingen i den retningen dere programmerer. Mål hvor lang tid det tar før kula treffer bakken for forskjellige vinkler, og lag en matematisk modell for sammenhengen mellom vinkel og tid.

Fase 1: Gjennomfør informasjonsinnhenting for å få idéer, dersom dere ønsker det.

Fase 2: Idémyldre og planlegge

Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan nyttige forslag bli avfeid for tidlig.

1. Tenk selv først og tegn gjerne skisser.
2. Forklar ideen din for de andre på gruppa.
3. Hele gruppa diskuterer de ulike ideene, og lager en felles hypotese for bygging.



Fase 3: Gjennomfør planen deres for å lage utskytningsmekanismen, og lag programmet for å stille inn vinkelen.

Fase 4: Test om utskytningsmekanismen virker som planlagt.

Fase 5: Sammenlign resultatene med andre i klassen.

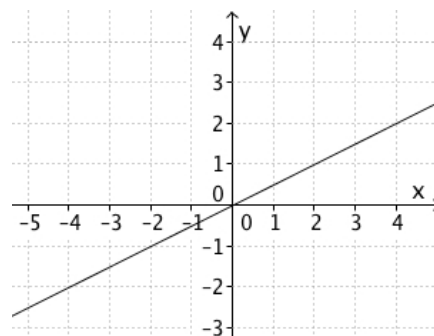
- Fikk noen andre større avstand?
- Hvorfor det?

Fase 6: Gå tilbake til de andre fasene for å gjøre de planlagte forbedringene.

Fase 7: Gjennomfør de siste målingene, disse skal dere bruke til å plotte en graf i Geogebra og lage en matematisk modell, se oppgaveboks under. Dokumentér prosjektet med et valgfritt produkt.

Oppgaver

1. Mål hvor lang tid kula bruker på å nå bakken med minst fem ulike vinkler.
2. Plott alle de målte verdiene i Geogebra.
3. Finn en matematisk modell ved å foreta en regresjon for de målte dataene.
4. Hvordan kan dere avgjøre hvilken modell som passer best?
5. Hva betyr det når x-verdien er negativ?



Modellering

Passer modellen bra med målingene deres?

Hva kan gyldighetsområdet til modellen kan være?
- hvorfor?

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 11

Innhold

Kapittel 11 – Kunst og håndverksopplegg

- Opplegg 33 – Geriljabroderi.....3
 - Lag et teknologisk geriljabroderi.....4
- Opplegg 34 – Lag en teknologisk 17.mai-fane.....5
- Opplegg 35 – Lag et produkt som bruker hvis-klosser.....6

Opplegg 33 - Geriljabroderi



Therese Susan Isaksen



Mathilde Ellingsen Skjæveland



Irene Skjølberg



Sissel Madsø
Granvang



Ira Pettersen



Ellen Strand Caspersen



Ingrid Langemyhr



Mathilde Ellingsen Skjæveland

Lag et teknologisk geriljabroderi

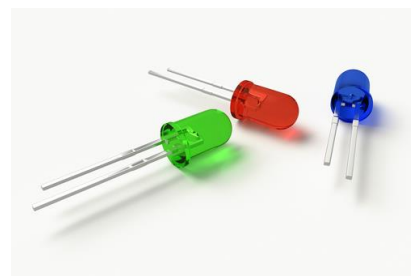
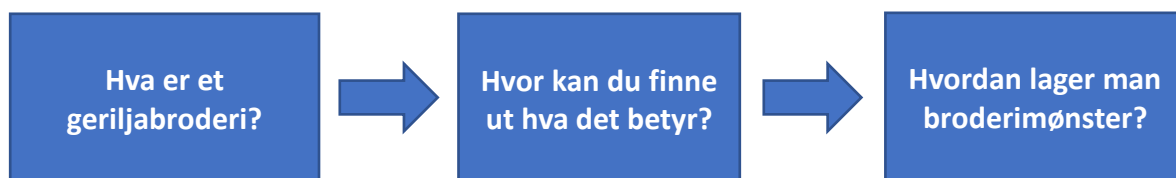
Oppgave

Lag et geriljabroderi der du også bruker elektrisk ledende sytråd, minst én LED-pære og en micro:bit.

Bruk innovasjonsmetoden fra opplegg 1 for å lage broderiet.



Fase 1: Undersøke og finne informasjon



Fase 2: Idémyldre og planlegge

Det finnes ledende sytråd som leder elektrisitet på akkurat samme måte som ledninger.

Hvordan kan du bruke den i broderiet ditt?

Hvordan kan du dele opp oppgaven? Hvilke ulike deler består den av?



Opplegg 34 - Lag en teknologisk 17. mai-fane

Oppgave

Lag en 17.mai-fane med elektroniske komponenter og minst én micro:bit.



Bruk innovasjonsmetoden fra opplegg 1 for å lage fanen.

Idémyldre

Idéer og plan

Utføre

Teste

Evaluerer

Forbedre

Dokumentere

Fase 1: Undersøke og finne informasjon

Hva slags teknologi kan brukes?



Hvor kan du finne ut om forskjellige elektroniske komponenter?



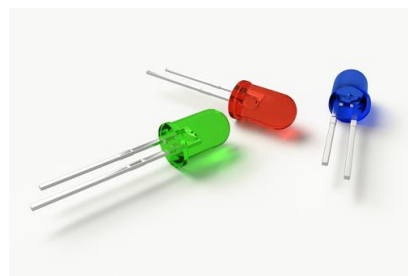
Hvordan passer de sammen med 17.mai-designet?



Fase 2: Idémyldre og planlegge

Hvordan kan du bruke de ulike elektroniske komponentene i fanedesignet?

Hvordan kan du dele opp oppgaven? Hvilke ulike deler består den av? Har dere gjort noe lignende før?



Opplegg 35 - Lag et produkt som bruker hvis-klosser

Oppgave

Konstruer et produkt som bruker en måling til å utføre en handling. Du kan bruke alle tilgjengelige materialer, inkludert servomotorer og elektronikk.

Bruk innovasjonsmetoden med de syv fasene fra opplegg 1.

Idémyldre

Idéer og plan

Utføre

Teste

Evaluerer

Forbedre

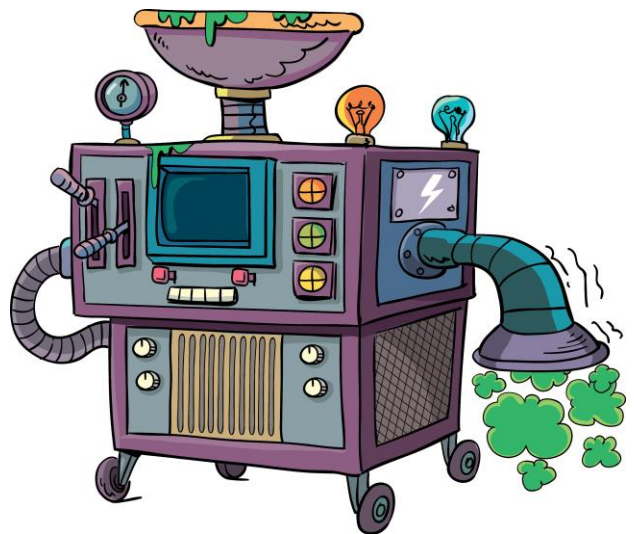
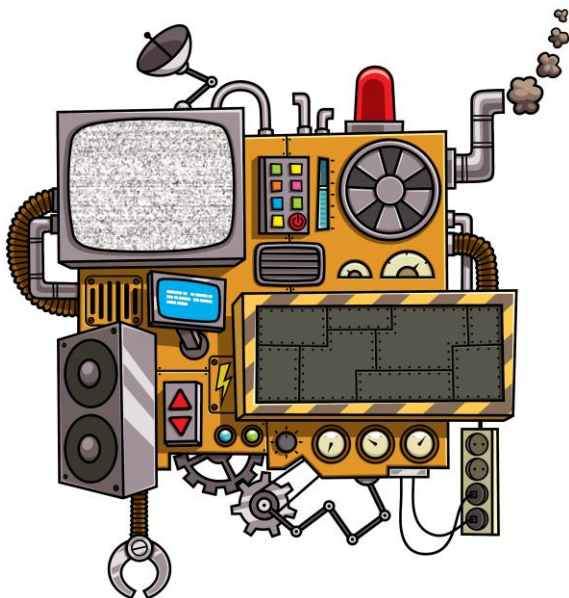
Dokumentere

Mulige målinger:

Temperatur
Lysstyrke
Spennings/motstand
Koble til andre sensorer?

Mulige handlinger:

LED-pære som lyser/blinker
Servo som beveger seg
Lage lyder/spille musikk



Fase 1: Undersøke og finne informasjon

Kan du lage et produkt som er nyttig for noen? Eller et produkt som gjør noe positivt for miljøet? Hvordan kan dere sjekke hva folk har bruk for? Skal dere bare google, eller kanskje spørre folk?

Fase 2: Idémyldre og planlegge

Hvordan skal dere kombinere sensorer med handlinger?

- Hvilket utstyr skal dere bruke?
- Hvordan blir programmeringen?

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 12

Innhold

Kapittel 12 – Introduksjon til Pythonprogrammering

• Introduksjon.....	3
• Variabler.....	4
• Hvis-setninger.....	5
• While-løkker.....	6
• For-løkker.....	7
• Lister.....	8
• Tabeller.....	9
• Funksjoner.....	10
• Tegning.....	11
• Python for micro:bit	
• Inndata.....	12
• Utdata.....	13
• Musikk.....	14
• Logging til pc/MAC.....	15
• Sonar.....	16
• Servo.....	17
• Skrive til fil.....	18

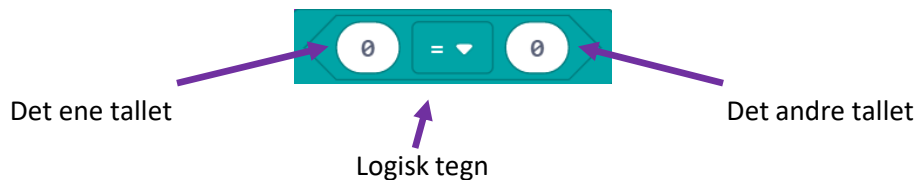


Introduksjon til Python-programmering

Vilkår (conditions)

Vilkår i Python er det samme som vilkår i Scratch eller MakeCode. Forskjellen mellom dem er at i stedet for å ha ferdige sekskanta klosser med vilkår, må vi skrive vilkårene selv i Python. Et vilkår er et utsagn som kan være sant eller usant, og den typen vilkår vi kan bruke i programmering vil alltid sammenligne noe. Ofte sammenligner vi tallverdier, og ser om de er like store eller det første er mindre enn det andre, eller omvendt. Det går også an å sammenligne tekster for å se om de er like eller ikke. Men for tekster kan man ikke finne ut om den ene teksten er større enn en annen. Hvordan skulle vi målt det?

Et vilkår i python må bestå av tre deler, akkurat som i Scratch og MakeCode. To tall som skal sammenlignes på en eller annen måte, og et tegn som sier hvordan de skal sammenlignes. Dette tegnet kalles ofte for et logisk tegn eller en logisk operator. Dersom sammenligningen stemmer, vil vilkåret gi oss verdien «sann» tilbake. Dersom sammenligningen ikke stemmer, vil vi i stedet få verdien «usann» tilbake. Sann og usann kalles boolske verdier. Det går an å ha boolske variable, det vil si at variabelen kan lagre verdien sann eller usann.



Under kommer en oversikt over hvordan man kan skrive forskjellige vilkår. Se om du klarer å finne ut om eksemplene returnerer verdien sann eller usann.

Vilkår		
Tegn	Forklaring	Eksempler
>	Større enn	4 > 5
<	Mindre enn	2 < 7
>=	Større enn eller lik	-2 >= 0
<=	Mindre enn eller lik	7 <= 12
==	Lik	9 == 19
!=	Ikke lik	3 != 6
or	Ett av vilkårene må være sanne	3 == 4 or 3 > 1
and	Alle vilkårene må være sanne	3 == 4 and 3 > 1

Regning

En del ganger har vi behov for å utføre regnestykker i Python. Dette er spesielt nyttig der vi har noen variabler vi, for eksempel, skal multiplisere eller legge sammen. Men det går fint an å utføre regnestykker direkte med tall. Under er en oversikt over forskjellige regneoperatorer for Python. Se om du klarer å se hva verdien av regnestykke-eksemplene blir.

Regneoperatorer		
Tegn	Forklaring	Eksempler
+	Addisjon	(-4) + 5
-	Subtraksjon	(-1) - (-4)
*	Multiplikasjon	3 * 0
/	Divisjon	12 / (-3)
**	Opphøyd i (eksponent)	2**3

Variabler i Python

Ulike typer variabler er viktige innen programmering. Ett av poengene med å lage programmer, er at vi skal kunne bruke det samme programmet på å løse flere lignende oppgaver. Da er det mye greiere å lagre alle tall vi skal bruke i variabler, slik at vi bare trenger å bytte ut tallene ett sted, for at programmet likevel skal gi oss riktig svar.

En variabel kan inneholde én verdi, og hver gang vi legger en verdi i en variabel, erstatter vi den verdien som eventuelt ligger i variabelen fra før. Når vi lager en variabel i Python, trenger vi ikke skrive hvilken type det er, det gjenkjenner Python ut fra den første verdien vi legger inn i variabelen. Det finnes fire typer variabler:



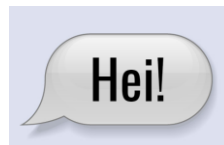
Heltall
- integer

1234
5678
90

Desimaltall
- float

0.02

Tekst
- string



Boolsk verdi
- boolean

True
False

Heltall kalles integer i Python. En integer er et positivt eller negativt tall som ikke har noen desimaler. 0 er også en integer.

Desimaltall kalles float i Python. En float er et tall som inneholder desimaler. Det kan være både negativt eller positivt. Tallet 2,0 er en float.

Tekst kalles string i Python. (På norsk blir også ordet «streng» brukt.) En tekstverdi er ett eller flere ord, det kan også være hele setninger. For at python skal forstå at det er en tekstverdi, må vi bruke apostrofer (') rundt teksten. Dersom vi vil legge ordet hei inn i variabelen a, må vi skrive kommandoen slik: `a = 'hei'`

Boolsk verdi kalles boolean value i Python. Denne verdien kan være enten sann eller usann. Den blir brukt i forbindelse med vilkår, siden et vilkår kan være enten sant eller usant.

Av og til trenger vi å gjøre om en type variabel til en annen type variabel. Slik som når det er en funksjon i Python som bare tar inn heltall. Dersom vi da har et desimaltall, må vi gjøre det om til et heltall, og da må vi avrunde desimaltallet vårt. Da bruker vi kommandoen `round()`. Det går også an å bruke kommandoen `int()`, men den avrunder ikke desimaltallet, den bare kutter ut alt bak kommaet. Det betyr i praksis at desimaltallet alltid blir rundet ned til nærmeste heltall.

For å kunne sende tall mellom to eller flere micro:biter, må verdiene være gjort om til tekstverdier. For å gjøre om en tallverdi til en tekstverdi kan vi bruke kommandoen `str()`.

Oppgave

Hva vil vises på skjermen for disse programmene? Kan du lage noen programmer selv? Få en i klassen til å gjette hva som vil skje i programmene dine, eller kanskje læreren?

```
a = 10
print(a)
a = a + 1
print(a)
```

```
a = 1.7
a = round(a)
print(a)
```

```
a = '1.7'
a = float(a)
print(a+a)
```

```
a = 1.7
a = str(a)
b = float(a)
print(a+'b')
```

```
a = 1.7
a = int(a)
print(a)
```

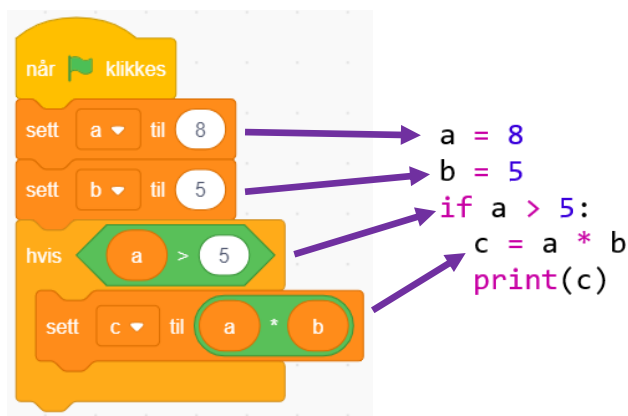
```
a = '1.7'
a = float(a)
print(a)
```

```
a = '1.7'
print(a+a)
```

```
a = 1.7
#a = str(a)
b = float(a)
print(a+b)
```

Hvis-setninger (if-setninger)

Hvis-setninger tilsvarer hvis-klosser i Scratch og MakeCode. De må inneholde et vilkår, og dersom vilkåret er sant, vil det som «tilhører» hvis-setningen bli utført. I de klossbaserte programmeringsspråkene putter vi inn det som skal skje ved hjelp av klossene som passer slik som puslespillbrikker. I Python må vi gjøre det samme, bare at vi må skrive alt selv. Siden vi nå ikke har klosser for å vise hva som står «inni» hvis-setningen, bruker vi innrykk i stedet. Alt som står etter en hvis-setning, med et innrykk foran, vil «stå inni» hvis-setningen, og blir bare utført om vilkåret i hvis-setningen er sant.



Ofte har vi behov for å ha flere alternativer som kan være sanne. Det finnes to forskjellige varianter, og de kalles gjerne «ellers» eller «ellers hvis».

Ellers – det som skal skje, dersom vilkåret i hvis-setningen er usant.

else:

Ellers hvis – det som skal skje dersom vilkåret i hvis-setningen er usant og følgende vilkår er sant.

elif:

Disse to programmene gjør det samme, og hver blokk i Scratch tilsvarer én linje i Python, med to unntak.

I Python trenger vi ikke å trykke på et flagg, vi trykker på en «play-knapp» som får koden til å begynne å kjøre. Dermed trenger vi ikke noe som tilsvarer den første blokka i Scratch-koden.

I Python viser ikke verdien på variablene, med mindre vi skriver dem ut til skjermen ved å bruke print-kommandoen. I Scratch vises verdien til alle variablene hele tiden, så der trenger vi ikke å få vist fram c på noen spesiell måte.

Snakk om

Hva måtte du endre i Python-programmet for at det bare skulle regne ut c når verdien i b var større enn verdien i a?

Hvilken forskjell hadde det blitt dersom vi flyttet linja med print(c) helt til venstre i Pythonkoden, slik at den ikke sto med innrykk?

Oppgave

Hva vil vises på skjermen for disse programmene? Kan du lage noen programmer selv? Få en i klassen til å gjette hva som vil skje i programmene dine, eller kanskje læreren?

```
a = 8
b = 5
if a > 5:
    c = a * b
    print(c)
else:
    print(a)
    print(b)
```

```
a = 8
b = 5
if a > b:
    c = a * b
    print(c)
elif a < b:
    print(a)
    print(b)
```

```
a = 5
b = 8
if a > b:
    c = a * b
    print(c)
elif a == b:
    print(a)
else:
    print(a)
    print(b)
```

```
a = 5
b = a
if a > b:
    c = a * b
    print(c)
elif a == b:
    print(a)
else:
    print(a)
    print(b)
```

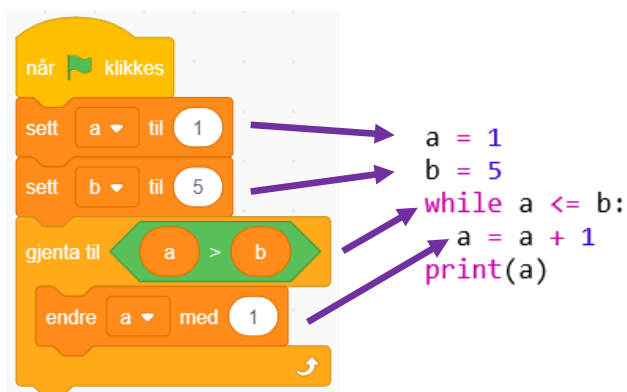
```
a = 5
b = 8
if a > b:
    c = a * b
    print(c)
elif a == b:
    print(a)
elif a < b:
    print(a)
    print(b)
```

Hvordan skrive hvis-setninger – en oppskrift

```
if betingelse:
    kode som utføres når betingelsen er sann
elif betingelse:
    kode som utføres når betingelsen er sann og betingelsen over er usann
else:
    kode som utføres når ingen av de andre betingelsene er sanne
```

While-løkker i Python

While-løkker (vilkårløkker) tilsvarer «gjenta til-løkkene» i Scratch. De må ha med et vilkår som forteller om det som står inni løkka skal gjøres. For Scratch er det slik at det som står inni løkka utføres helt til vilkåret blir sant (altså at løkka går mens vilkåret er usant). I Python er det motsatt, at det som står inni løkka blir utført helt til vilkåret ikke lenger er sant. Det gjør at selve vilkåret må skrives på forskjellige måter, dersom man skal oppnå det samme med Scratch og Python.



Snakk om

Hva vil vise på skjermen for Python-programmet?

Vilkåret i de to programmene er ikke like. Hva ville Pythonprogrammet vist, dersom man brukte vilkåret fra Scratch-programmet direkte?

Hvilken verdi ville a hatt, dersom vi brukte vilkåret fra Pythonprogrammet direkte i Scratch-programmet?

Disse to programmene gjør det samme, og hver blokk i Scratch tilsvarer én linje i Python, men med to unntak.

I Python trenger vi ikke å trykke på et flagg, vi trykker på en «play-knapp» som får koden til å begynne å kjøre. Dermed trenger vi ikke noe som tilsvarer den første klossen i Scratch-koden.

I Python vises ikke verdien på variablene, med mindre vi skriver dem ut til skjermen ved å bruke print-kommandoen. I Scratch vises verdien til alle variablene hele tiden, så der trenger vi ikke å få vist fram a på noen spesiell måte, men vi rekker ikke å se hvilke verdier a har underveis, siden programmet går så raskt.

Hvordan skrive while-løkker – en oppskrift

while vilkår:

kode som utføres mens vilkåret er sant

Pass på å sette alt som skal være inni while-løkka med innrykk. Dersom det ikke står med innrykk, vil det ikke skje før løkka er ferdig, og programmet fortsetter videre.

Oppgave

1. Hva vil vises på skjermen for disse programmene?

Skriv ned hva du tror vil vises på skjermen, og skriv inn programmene med Pythonkode etterpå. Kjør koden, og sjekk om du får det svaret du trodde du skulle få.

- Dersom du ikke fikk svaret du trodde, hva var grunnen til det, tror du? Diskuter gjerne med andre.
- Kan du lage noen programmer selv? Få en i klassen til å gjette hva som vil skje i programmene dine, eller kanskje læreren?

```
a = 1
b = 5
while a < b:
    a = a + 1
    print(a)
```

```
a = 1
b = 5
while a > b:
    a = a + 1
    print(a)
```

```
a = 10
b = 0
while a > b:
    a = a - 1
    print(a)
```

```
a = 10
b = 0
while a > b:
    a = a - 1
    print(a)
```

```
a = 1
b = 5
while a < b:
    a = a + 1
    print(a)
    print(b)
```

For-løkker i Python

For-løkker (telleløkker) tilsvarer «gjenta et visst antall ganger-løkkene» i Scratch. Disse løkkene gjentar det som står inni dem så mange ganger som vi ønsker. I Scratch kan vi bare skrive direkte hvor mange ganger vi vil gjenta løkka, men i Python er det litt annerledes. Der blir det opprettet en variabel som heter *i*, vi kan gjerne kalle det en tellevariabel. Den begynner på den tallverdien som står først i parentesen til for-løkka, i vårt eksempel er det 0. Det andre tallet i parentesen viser hvilket tall tellervariabelen i skal telle seg opp til (men ikke inkludert). I vårt tilfelle skal *i* telle seg opp til verdien 5. Det siste tallet inni parentesen sier hvor store steg vi skal øke verdien til tellevariabelen i med. I vårt eksempel skal vi telle med 1, altså sier vi at tellervariabelen begynner på 0, vi teller opp til 5 ved å øke med 1 hver gang. Da vil *i* være 0, 1, 2, 3, 4 før løkka blir avslutta.



```
for i in range (0, 5, 1):  
    print(i)
```

Snakk om

Hva ville skjedd om vi brukte variabel *a* i stedet for tallet 4 i løkka i Scratch?

Går både Scratch-programmet og Pythonprogrammet gjennom løkka like mange ganger?

Hvordan skrive for-løkker – en oppskrift

`for i in range (fra og med-verdi, til-verdi, steglengde):`
kode som utføres mens *i* teller gjennom verdiene den skal ha

Disse to programmene gjør det samme, men her tilsvarer ikke hver kloss i Scratch én linje i Python. Her er tre av blokkene i Scratch innbakt i for-løkka i python. I tillegg er det to klosser i Scratch som ikke tilsvarer noen linjer i Python.

I Python trenger vi ikke å trykke på et flagg, vi trykker på en «play-knapp» som får koden til å begynne å kjøre. Dermed trenger vi ikke noe som tilsvarer den første klossen i Scratch-koden.

I Python viser ikke verdien på variablene med mindre vi skriver dem ut til skjermen ved å bruke print-kommandoen. I Scratch vises verdien til alle variablene hele tiden, så der trenger vi ikke å få vist fram *a* på noen spesiell måte, men vi rekker ikke å se hvilke verdier den har underveis, siden det går så raskt. For å kunne se hvilke verdier variabelen i Scratch har, så har vi denne gang lagt inn en kloss som sier «vent 1 sekunder», og den gjør at programmet tar en pause i ett sekund. Da rekker vi å se verdien på variabelen før programmet går videre igjen.

Oppgave

1. Hva vil vises på skjermen for disse fem programmene?

Skriv ned hva du tror vil vises på skjermen, og skriv inn programmene med Pythonkode etterpå. Kjør koden, og sjekk om du får det svaret du trodde du skulle få.

2. Dersom du ikke fikk svaret du trodde, hva var grunnen til det, tror du? Diskuter gjerne med andre.

3. Kan du lage noen programmer selv? Få en i klassen til å gjette hva som vil skje i programmene dine, eller kanskje læreren?

```
for i in range (0, 5, 2):  
    print(i)
```

```
for i in range (0, 5, 0):  
    print(i)
```

```
for i in range (1, 10, 2):  
    print(i+i)
```

```
for i in range (1, 10, 2):  
    print(i-i)
```

```
for i in range (1, 10, 1):  
    print(i**2)
```

Lister i Python



Ofte trenger vi å bruke mange variabler, og da kan det være greit å lage en liste eller en tabell (array i Python). Dersom én variabel tilsvarer én skuff som har et navn og en verdi, så er en liste som en kommode med flere skuffer. Hver variabel er ett element i lista, og hvert element har et navn og en verdi. Navnet på elementet er navnet på hele lista sammen med plasseringen på lista, altså hvilken rad det tilhører.

Vi trenger dermed ikke å lage et eget navn til hvert element. Det er også enkelt å bruke en løkke for å gå gjennom alle elementene for å bruke dem i, for eksempel et regnestykke.

En liste har et navn og elementer med hvert sitt nummer. Det som er litt spesielt med lister innen programmering, er at det første elementet er element nummer 0. Det er fort gjort å glemme når man programmerer, og er alltid lurt å sjekke som en del av feilsøkingen. En liste har i utgangspunktet bare elementer med én datatype. Det går ikke an å bruke noen heltall og noen tekstverdier i samme liste. En liste er i utgangspunktet en lang rekke med elementer, og er dermed endimensjonal.

Navnet på lista må være ett sammenhengende ord for at Python skal forstå det. Innen programmering er det vanlig å bruke `_` som mellomrom for navn på variabler, lister og tabeller. Bokstavene æ, ø og å bør vi unngå, siden noen Python-editorer ikke takler navn som bruker andre enn engelske tegn.

Ønskeliste

0. Hund
1. Mobil
2. Sko
3. Bukse
4. Hagenisse



Et element har både et navn og en verdi.

Hva er navnet og hva er verdien til dette elementet?

Oversikt over noen nyttige kommandoer i Python

```
liste = []
```

Lager en tom liste.

```
liste = [1, 3, 5, 7]
```

Lager en liste med verdiene 1, 3, 5, 7.

```
liste[3] = 7
```

Legger inn verdien 7 i element tre, i en liste med minst fire elementer.

```
del liste[3]
```

Sletter element tre, i en liste med minst fire elementer.

```
liste.append(4)
```

Legger til verdien fire som et nytt element i en liste som heter liste. Det har ingenting å si hvor mange elementer som er i lista fra før, for verdien blir alltid lagt i et nytt element etter det siste elementet i den opprinnelige lista.

```
print(liste[2])
```

Henter verdien til element to i lista, og skriver det på skjermen.

```
len(liste)
```

Finner hvor mange elementer det er i en liste.

Diskusjonsoppgaver

1. Hvorfor er det liten vits i å bruke en løkke for å legge sammen mange variabler?
2. Hvorfor er det enkelt å bruke en løkke for å gjøre det samme dersom du har alle verdiene i en liste?

Tabeller i Python

En tabell (array) kan inneholde flere typer data, og den kan være todimensjonal. Det vil si at den har mer enn én kolonne med data. Da trenger vi to tall for å forklare hvilket element i tabellen vi prater om. Det blir på samme måte som et punkt i et koordinatsystem, der vi må bruke både x- og y- koordinaten for å plassere et punkt. Det første elementet begynner på 0 for tabeller også, både for radene og for kolonnene. Tabeller finnes i numpy-biblioteket, og må importeres: `from numpy import *`

Rene tekstfiler (.txt-filer) fra statistisk sentralbyrå gir todimensjonale (2D) tabeller i Python, og er dermed viktige når vi skal analysere data, og lage matematiske modeller. Det er ganske enkelt å gjøre om en 2D- array til en 1D-array som er det `fit_function()` tar inn av data (se kapittel 13).

Tabellen under har navnet budsjett. Akkurat som for lister, begynner vi å telle på null. Dette gjelder for både radene (vannrett) og kolonnene (loddrett). Rad null i budsjett-tabellen inneholder tekstverdiene: Måned, Inntekt, Klær og snacks, Mobilutgifter, Diverse utgifter, SUM. Kolonne null inneholder Måned, Januar, Februar, Mars, April, Mai, Juni.

Måned	Inntekt	Klær og snacks	Mobilutgifter	Diverse utgifter	SUM
Januar	1500	1000	300	100	100
Februar	1500	1000	300	100	100
Mars	3000	1900	300	100	700
April	1500	1000	300	100	100
Mai	1500	1000	300	100	100
Juni	2500	1000	300	300	900

Dette er element [3,5] og har verdien 700.

Diskusjonsoppgaver

1. Hvilken verdi har element [0,0]?
2. Hvilket element har verdien 3000?

Oversikt over noen nyttige kommandoer

```
budsjett[7][2] = 2000
```

Endrer verdien fra 1500 til 2000 i elementet i rad syv og kolonne to i budsjett-tabellen på denne sida.

```
print(budsjett[3][1])
```

Skriver ut verdien til elementet i rad tre og kolonne en, altså 1500 fra budsjettet på denne sida.

```
tabell = zeros([2,3])
```

Lager en tabell med to rader og tre kolonner, der alle verdiene er null.

```
tabell = array([(1, 2, 3),(4, 5, 6)])
```

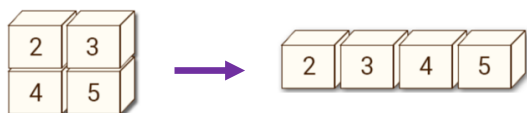
Lager en tabell med to rader og tre kolonner. Tallene innenfor parentesene angir verdiene til elementene i hver rad.

```
tabell = linspace(0, 5, 11)
```

Lager en 1D-tabell med 11 verdier jevnt fordelt fra og med null til og med fem.

```
a = tabell.flatten()
```

Gjør om en 2D-tabell til en 1D-tabell. Alle radene i 2D-tabellen blir lagt etter hverandre til en lang rad.



Funksjoner i Python

Funksjoner i Python tilsvarer «lag en ny kloss» i Scratch, og er egentlig bare småprogrammer. Faktisk trenger en funksjon ikke være veldig liten heller, men det som skiller den fra et vanlig program, er at den må bli definert før den kan brukes. Når funksjonen er definert, så kan vi bruke den så mange ganger vi ønsker uten å skrive all koden på ny, bare ved å skrive én kodelinje.



Disse to programmene gjør det samme, men her tilsvarer ikke hver kloss i Scratch én linje i Python. Funksjonen som blir definert, inneholder en for-løkke som ikke er bygd opp på helt lik måte i Scratch og Python (se for-løkke siden).

I Python trenger vi ikke å trykke på et flagg, men vi må legge inn en linje for å vise variabelen på skjermen. For å kunne se alle verdiene på skjermen, har vi lagt inn en pause på ett sekund i Scratch-programmet.

Både i Scratch og Python kan funksjonene motta innverdier. Det vil si at når vi bruker funksjonen, så må vi oppgi én eller flere verdier som blir plassert i variabler. Disse verdiene blir da brukt til noe i funksjonen, gjerne en utregning. I eksempelet heter variabelen til inn-verdien i Pythonprogrammet inndata.

En viktig forskjell på funksjoner i Scratch og Python, er at i Python kan funksjonene gi oss en ut-verdi. Det vil si at vi kan for eksempel bruke en funksjon til å regne ut et regnestykke for oss, og gi oss et tall tilbake. Dette tallet kan vi for eksempel putte inn i nye regnestykker, i hvis-setninger eller i løkker. I Scratch går ikke dette an.

Snakk om

Hvorfor må vi ha med at vi skal sette variabelen a til 0 i Scratch-programmet?

Går både Scratch-programmet og Pythonprogrammet gjennom løkka like mange ganger?

Oppgave

1. Hva vil vises på skjermen for disse tre programmene?

Skriv ned hva du tror vil vises på skjermen, og skriv inn programmene med Pythonkode etterpå.

Kjør koden, og sjekk om du får det svaret du trodde du skulle få.

2. Dersom du ikke fikk svaret du trodde, hva var grunnen til det, tror du? Diskuter gjerne med andre.
3. Kan du lage noen programmer selv? Få en i klassen til å gjette hva som vil skje i programmene dine, eller kanskje læreren?

```
def Funksjon(inndata):  
    return inndata*2  
  
Funksjon(1)
```

```
def Funksjon(inndata):  
    return inndata*2  
  
print(Funksjon(8.5))
```

```
def Funksjon(inndata):  
    return inndata**2  
  
for i in range(1, 11, 1):  
    print(Funksjon(i))
```

Hvordan definere og bruke funksjoner – en oppskrift

```
def navn(variabler):  
    kode som utføres av funksjonen  
    return returverdien til funksjonen
```

navn(variabler)

Tegning i Python - turtle



Python har et bibliotek som heter turtle, som kan brukes når man ønsker å bruke Python til å tegne figurer eller mønstre. For å importere dette biblioteket, må man skrive: `from turtle import *` øverst i programmet.

Eksempler på noen grunnleggende funksjoner fra turtle-biblioteket

```
t = Turtle()
```

Denne funksjonen lager et «turtle-objekt» som man trenger når man bruker de andre funksjonene som lager tegningen. Vi kan tenke på det som en blyant eller penn.

```
t.forward(100)
t.backward(100)
```

Disse funksjonene flytter turtle frem eller tilbake 100 steg.

```
t.left(60)
t.right(60)
```

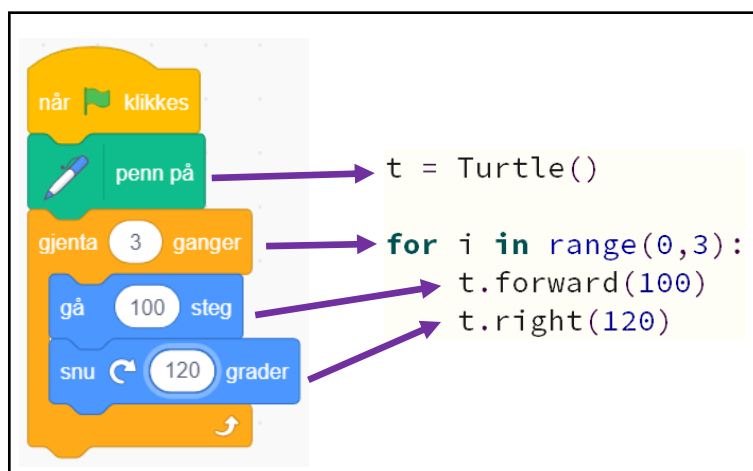
Disse funksjonene snur turtle 120 grader til venstre eller til høyre (det er det samme som at den ytre vinkelen er 60 grader).

```
t.circle(50)
```

Denne funksjonen tegner en sirkel med radius 50 steg.

```
t.goto(100,150)
```

Denne funksjonen får turtle til å bevege seg til punktet med x-koordinat 100 og y-koordinat 150.



Oppgave

1. Hva vil vises på skjermen for disse programmene?

Skriv ned hva du tror vil vises på skjermen, og skriv inn programmene med Pythonkode etterpå. Kjør koden, og sjekk om du får det svaret du trodde du skulle få.

2. Dersom du ikke fikk svaret du trodde, hva var grunnen til det, tror du? Diskuter gjerne med andre.

3. Kan du lage noen programmer selv? Få en i klassen til å gjette hva som blir tegnet med programmene dine, eller kanskje læreren?

```
t = Turtle()
for i in range(0,4):
    t.forward(100)
    t.right(90)
```

```
t = Turtle()
for i in range(0,36):
    t.forward(10)
    t.left(10)
```

```
t = Turtle()
for i in range(0,36):
    t.right(10)
    for i in range(0,36):
        t.forward(10)
        t.left(10)
```

Oversikt over noen kommandoer for micro:bit i Python Mu

Når vi skal bruke Python for å programmere en micro:bit, finnes det en del ferdige kommandoer vi kan bruke. Her viser vi en oversikt over noen av de mest nyttige. Andre eksempler kan du finne på denne nettsiden: <https://microbit-micropython.readthedocs.io/en/v1.0.1/tutorials/introduction.html>

Vi må alltid begynne med å importere det biblioteket som inneholder micro:bit-kommandoene. Det gjør man ved å skrive øverst: `from microbit import *`

Inndata

Knapper

Siden micro:biten har to knapper, A og B, så kan den delen av koden der det står `button_a` i eksemplene byttes ut med `button_b` i stedet.

`button_a.is_pressed()`

`button_a.get_presses()`

knapp

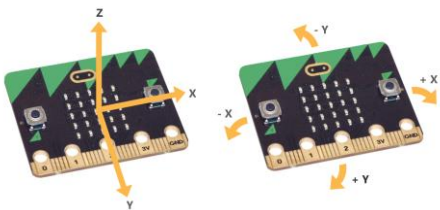
A ▼

trykkes

Denne kommandoen teller antall ganger en knapp har vært trykket siden programmet startet.

Innebygde sensorer

Vi kan bruke micro:bitens innebygde sensorer med Python-programmering også. For akselerasjonen, kan denne måles i alle de tre romlige retningene vi har. Tenk på det som et koordinatsystem med 3 akser. De vanlige, x-aksen og y-aksen, men også en z-akse som står loddrett på begge de to andre aksene.





`accelerometer.get_x()`

akselerasjon (mG) x ▼



`temperature()`

temperatur (°C)



`display.read_light_level()`

lysnivå



`running_time()`

kjøretid (ms)

Påkoblede sensorer – se også siden om sonar

En digital verdi kan bare være enten 1 eller 0, der 1 betyr på og 0 betyr av. Den digitale verdien 1 er egentlig en høy spenning, mens den digitale verdien 0 er en veldig lav spenning. Digitale verdier bruker vi sjelden for sensorer, bare dersom vi skal sjekke om noe er avskrudd eller påskrudd. Sensorer vi kobler på micro:biten gir som regel verdier mellom 0 og 1023, og da må vi lese av en analog verdi i stedet. En analog verdi kan være mye annet enn bare 0 eller 1, og passer dermed best for å lese av hvilken verdi sensoren gir.

`pin0.read_analog()`

`pin0.read_digital()`

les analogverdi fra P0 ▼

les digitalverdi fra P0 ▼

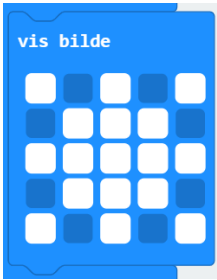
Utdata

Skrive til skjermen

Det går an å skru på de enkelte lysdiodene på skjermen til micro:biten, med verdier mellom 0 og 9 for hvor sterkt de skal lyse.

```
display.show(Image("00909:09990:99999:09990:90909"))
```

display.show()



Påkoblede lysdioder

Når du kobler på en lysdiode til micro:biten, kan du programmere den til å skru seg av eller på. Da må man bruke kommandoen som sender ut digital data, altså verdien 1 eller 0, der 1 tilsvarer på og 0 tilsvarer av.

Hvilken utgang lysdioden er tilkoblet (0, 1 eller 2).

```
pin0.write_digital(1)  
pin0.write_digital(0)
```

På (1) eller av (0).



Sende data over radio-forbindelsen

På samme måte som ved å bruke klosser, kan vi programmere micro:bitene til å sende data til hverandre ved å bruke Python.



```
import radio  
radio.on()
```

Import radio-kommandoen importerer biblioteket som gjør at vi kan bruke radio-forbindelsen mellom micro:bitene. I tillegg må vi bruke radio.on() som skruer på radioforbindelsen.

Til slutt må vi definere hvilken kanal micro:bitene skal kommunisere på. Da må vi passe på å velge samme kanal for alle micro:bitene som skal «snakke sammen».



```
radio.config(channel=1)
```



```
x = str(temperature())  
radio.send_bytes(x)
```

Når man programmerer i Python, må man gjøre om tall til tekst (strenger) før det går an å sende dem over radio-forbindelsen. Det gjøres med str()-kommandoen.



```
x = radio.receive_bytes()
```

Denne kommandoen tar i mot teksten som blir sendt og legger den i en variabel kalt x.

Musikk

- med Python Mu



En micro:bit kan enkelt kobles til en summer (også kalt buzzer) eller høyttaler for å spille av lyd. Siden en summer bare kan spille av en enkelt tone, vil det bli mye finere lyd fra en høyttaler, da den kan spille de overtonene vi er vant med å høre som en del av de rene tonene i musikk.

For å kunne bruke en høyttaler eller en summer i Python Mu, må vi importere et ferdig bibliotek som inneholder metoder for å lage og kontrollere lyd.

```
from music import *
```

Når vi skal spille av en tone i Python Mu, må vi oppgi tre ting som tekst (strenger). Tonens navn, hvilken oktav den tilhører og hvor lang tid den skal spilles. Et piano dekker om lag syv oktaver, MakeCode for micro:bit dekker tre oktaver og Python Mu dekker ni oktaver. Oktav 0 er den laveste oktaven, som gir de mørkeste tonene. Oktav nummer fire er den som inneholder kammertonen A med frekvens 440 Hz. I MakeCode tilsvarer lav oktav (liten) oktav nummer tre, midtre oktav (enstrøken) oktav nummer fire, og høy oktav (tostrøken) oktav nummer fem i Python Mu. Lengden på tonen oppgis som et tall som tilsvarer antall slag multiplisert med 4. Der lengden på hvert slag er definert av Mu. Funksjonen under spiller av fire toner.

```
music.play("C3:4", "03:4", "E3:4", "C#3:8")
```

Navn på tonen Hvilken oktav tonen tilhører Hvor mange slag tonen skal spilles

Dersom vi ønsker å skrive litt mindre, kan play()-funksjonen huske hvilken oktav vi bruker og hvor mange slag tonen varer. Dette er spesielt nyttig å bruke når vi skriver hele melodier, da slipper det å bli fullt så mye å skrive. Som regel vil en melodi i hovedsak være del av bare én oktav. Når vi benytter oss av Mus hukommelse, ser kommandoen over slik ut:

```
music.play("C3:4", "D", "E", "C#:8")
```

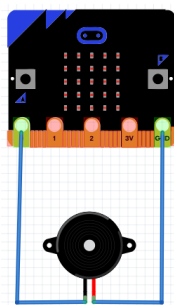
Dersom du vil bestemme selv akkurat hvor lenge hver tone skal spilles, kan du bruke en funksjon som kalles pitch(). Da må du oppgi frekvensen til tonen, og ikke navnet på den. Noen frekvenser er oppgitt i tabellen i opplegg 11. Spilletiden oppgis i millisekunder, og dermed kan du bestemme tempoet (bpm) med stor nøyaktighet.

```
music.pitch(440, 1000, pin1)
```

Frekvensen Varighet i millisekunder Hvilken utgang høyttaleren (summeren) er koblet til. Denne trenger du ikke skrive noe på, dersom du bruker pin0, da den ligger inne som standardverdi.

Det finnes også en rekke ferdige melodier. Du kan finne dem på denne siden: <https://microbit-micropython.readthedocs.io/en/v1.0.1/tutorials/music.html>

Det har ikke noe å si hvilken utgang på høyttaleren eller summeren du kobler til hvilken utgang på micro:biten.



Oppgave

Lag et program som spiller av «Lisa gikk til skolen».

Lag en egen melodi som du spiller av på micro:biten.

Logge til PC/MAC med Python Mu

Noe det kan være veldig fint å bruke micro:bit til, er som en datalogger. Det vil si at vi bruker den til å samle inn data, slik vi har gjort mye av, også får vi datamaskinen til å skrive opp målingene på skjermen og lage en graf av dem mens vi måler.

For å vise noe på skjermen bruker vi `print()`-kommandoen. Det som står inni parenteser er det som kommandoen skriver til skjermen, altså viser på skjermen. Dersom vi bare er interessert i å få ut en liste med målinger etter hvert som de blir målt, kan vi bare bruke en kodesnutt som ser slik ut:

```
x = temperature()
print(x)
```

Måler temperaturen og legger verdien i variabel x.

Skriver ut verdien av variabel x til skjermen.

Etter at vi har overført hele programmet til micro:biten, må vi trykke på to knapper for å få fram tabellen med målingene. Trykk først på «Plotter», deretter på «REPL». Det er viktig å gjøre dette i denne rekkefølgen, selv om vi ikke skal vise noen graf (plotteren viser grafen), ellers krasjer programmet.



The screenshot shows the BBC micro:bit REPL interface. At the top, there is a toolbar with icons for Mode, New, Load, Save, Flash, Files, REPL, Plotter, Zoom-in, Zoom-out, Theme, Check, Help, and Quit. The REPL window displays a list of temperature readings: 27, 28, 28, 28, 29, 29, 28, 28, 28, 28, 28, 28, 28, 28.

Slik ser da listen med målingene ut.

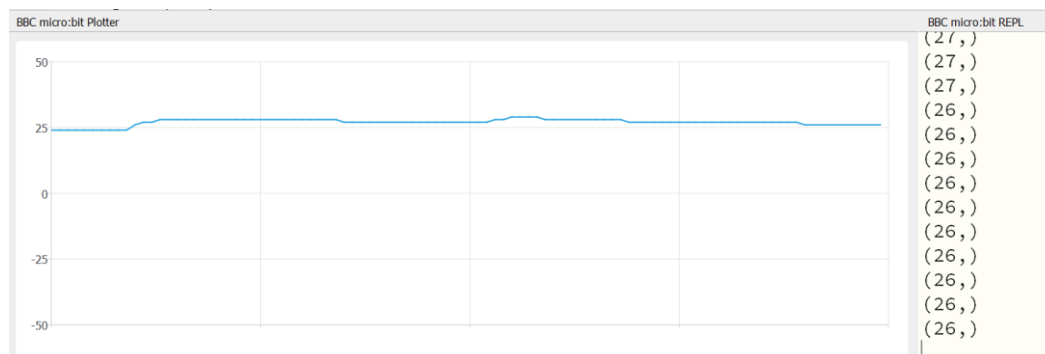
Ofte vil vi heller se grafen for målingene, og da må vi forandre litt på koden vår. I Python Mu vil vi få vist grafer, dersom vi har riktig format på det vi skriver ut til skjermen. Dette formatet blir vist under.

```
x = temperature()
print((x,))
```

Diskuter

Hva er forskjellen på de to kodesnuttene på denne siden?

Når det nye programmet er overført, og du åpner plotter, deretter REPL, så vil vi se både grafen og en liste over verdiene. Denne lista er litt mer tungvint å lese ut fra, men så lenge man klarer å se bort fra ekstra parenteser og komma, så står verdiene fra temperaturmålingene som på figuren under.



Oppgave

Lag hele programmet som måler temperaturen og plotter den som en graf på skjermen.

Hva skjer om du skriver inn en annen verdi i stedet for et tomt felt i `print()`-kommandoen?

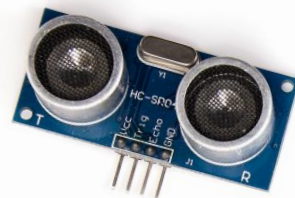
Diskuter

Å logge data kan vi gjøre med alle de innebyggede sensorene til micro:bit, men også med alle sensorer vi kobler på selv.

Hva må du endre i koden for å få til dette?

Avstandsmåler (sonar)

- med Python Mu

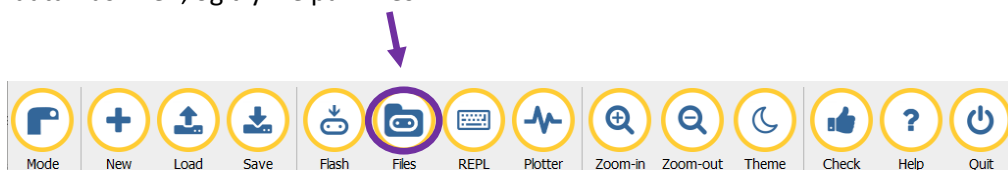


En sonar sender ut et lydsignal, og måler hvor lang tid det tar før det signalet blir reflektert tilbake til sonaren. Dersom man vet hva lydfarten er, så kan vi enkelt regne ut avstanden til det som reflekterer lydsignalet. Sonaren vet hva lydfarten er, og vi kan dermed bruke den til å måle avstander med. For å kunne bruke en sonar i Python Mu, så må vi importere et bibliotek på selve micro:biten. Dette biblioteket er egentlig et program som gjør det enklere for oss å programmere servoen, uten å måtte tenke på hvordan alle detaljene i målingene.

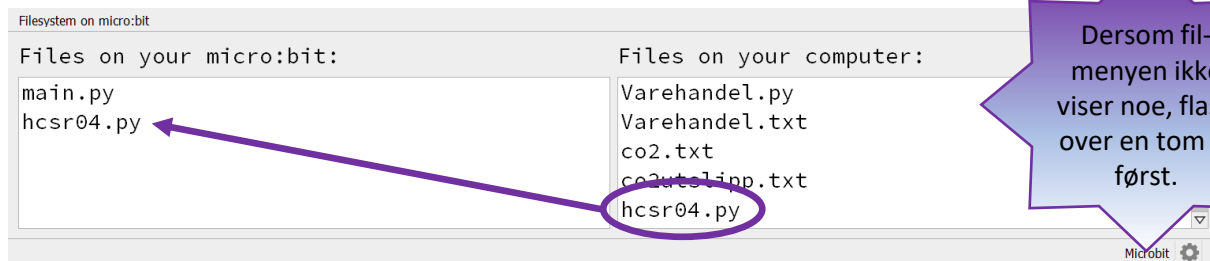
Det første vi må gjøre er å få tak i biblioteket til sonaren, og det kan finnes på www.kunnskapsfilm.no.....

Kopier koden som står på siden i en ny fane (fil) i Python Mu og lagre den som «hcsr04.py» ved å trykke på «Save» og lagre den i katalogen som dukker direkte opp.

Så må denne fila legges over på selve micro:biten, og det gjøres ved å koble micro:biten til datamaskinen, og trykke på «Files».



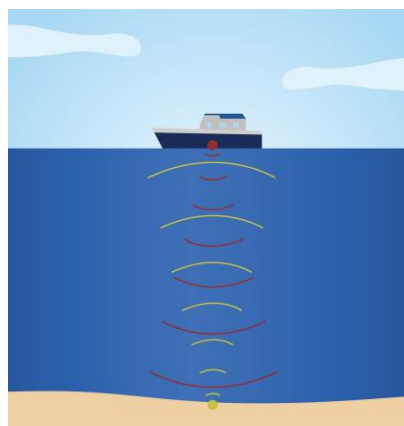
Da kommer denne menyen opp i Mu, og du må lete i feltet på høyre side til du finner fila som du nettopp lagret: «hcsr04.py». Når du finner den, drar du den over i det venstre feltet. Da vil lyset på baksiden av micro:biten blinke til programmet er overført.



Da gjenstår det å lage selve programmet som bruker sonaren til å måle avstander med. Under er det en puslespill-oppgave der kodelinjene må ryddes i riktig rekkefølge for å lage dette programmet.

Puslespill-oppgave

```
print((HCSR04(),))  
from hcsr04 import HCSR04  
sleep(1000)  
while True:  
    from microbit import *
```



Servo

- med Python Mu



En servo mottar et signal fra micro:biten slik at den beveger seg slik vi ønsker. Det finnes to typer servoer, 180° og kontinuerlige (360°), og de blir styrt på litt forskjellig måte. 180°-servoen kan bevege seg fra en -90°-vinkel til en 90°-vinkel, altså 180° til sammen. For en kontinuerlig servo oppgir vi hvor raskt den skal snurre rundt i prosent. Selve hastigheten kan variere litt med hvor mye strøm som er igjen i batteriene, for servoer trekker ganske mye strøm. Så dersom du har en servo på 100 % hastighet på begynnelsen og slutten av skoletimen, vil den snurre en del saktere på slutten av timen.

For å kunne bruke en servo i Python Mu, må vi importere et bibliotek på selve micro:biten. Dette biblioteket er egentlig et program som gjør det enklere for oss å programmere servoen, uten å måtte tenke på alle detaljene i signalene som styrer den, tilsvarende som for sonaren.

Det første vi må gjøre er å få tak i biblioteket til servoen, og det og det kan finnes på nettressursen til Kobling på www.kunnskapsfilm.no.

Kopier koden som står på siden i en ny fane (fil) i Python Mu og lagre den som «servo.py» ved å trykke på «Save» og lagre den i katalogen som dukker direkte opp.

Så må denne fila legges over på selve micro:biten, og det gjøres slik som for sonaren på forrige side.

180°-servo

Sett vinkel på servo P0 ▼ til 90 °

```
servovinkel(pin0, 90)
```

Ved å bruke denne kommandoen, kan vi stille in hvilken vinkel servoen skal bevege seg til. Kommandoen tar inn hvilken utgang vi har koblet servoen til, og hvilken vinkel servoen skal bevege seg til. Vinkelen kan variere mellom -90° og 90°.

Kontinuerlig servo (360°)

servo P0 ▼ kjører kontinuerlig på 50 %

```
servohastighet(pin0, 50)
```

Ved å bruke denne kommandoen, kan vi stille in hvilken hastighet servoen skal snurre med. Denne kommandoen tar også inn hvilken utgang vi har koblet servoen til, i tillegg til hastigheten i prosent. Hastigheten kan variere mellom -100 % og 100 %, der fortegnet bestemmer i hvilken retning servoen skal snurre.

stans servo P0 ▼

```
servostopp(pin0)
```

Denne kommandoen får en kontinuerlig servo til å slutte å snurre rundt. Det går også an å bruke kommandoen `servohastighet()` til dette.

Skrive til fil på micro:bit

- med Python Mu



I slutten av kapittel 9 brukte vi datasett i form av tekstfiler for å lage matematiske modeller. Hadde det ikke vært kult om vi kunne fått micro:biten til å lage slike filer av sine målinger? Da kan vi bruke micro:biten til å samle inn data over en viss periode, uten å være koblet til en datamaskin. Datasettet blir lagret i micro:biten som en .txt-fil, og vi kan legge den over i datamaskinen, før vi plotter resultatene eller lager en matematisk modell med Python.

Det aller første vi må gjøre, er å opprette en tekstfil, det vil si en fil som slutter på .txt, og det er enklest å gjøre i Mu. Lagre den samme i samme mappe som resten av dine Python Mu-filer.

Deretter må den fila du har laget, kopieres over på micro:biten. Dette gjør du på samme måten som for sonaren.

```
with open ('filnavn.txt', 'w') as fil:
```

Åpner fila som heter filnavn.txt og tildeler den til et objekt som vi kaller fil. Siden vi har 'w' (write) i kommandoen gjøres det klar til å skrive inn i fila.

```
with open ('filnavn.txt', 'r') as fil:
```

Tilsvarende som linja over, men siden vi har 'r' (read) i kommandoen gjøres det klar til å lese fila. Det er valgfritt å ta med 'r'.

```
fil.write(y+'\n')
```

Skriver variabelen y til fila vår. '+\n' gjør at det blir et linjeskift etter hvert tall, og det må vi ha for at python skal kunne lese filene. Må stå med innrykk.

```
y = str(x)
```

Gjør om variabelen x til en streng (tekstverdi). Må stå med innrykk.

```
z = fil.read()
```

```
print(z)
```

Skriver variabelen z til skjermen på PC. Må IKKE stå med innrykk.

Leser det som står i fila vår, og legger det inn i variabel z. Må stå med innrykk.

Da gjenstår det å lage selve programmet. Under er et eksempel der micro:biten måler temperaturen og legger målingen i en .txt-fil ved navn datasett.txt. Det kan være greit å kunne sjekke hvordan datasett.txt-fila ser ut uten å overføre til datamaskinen og åpne den, derfor er dette også med i puslespill-oppgaven.

Puslespill-oppgave

```
print(innhold)          t = temperature()
while not button_a.is_pressed():
    sleep(1000)          fil.write(str(t)+'\n')
with open('datasett.txt') as fil:
with open('datasett.txt', 'w') as fil:
    innhold = fil.read()  from microbit import *
```

Ekstraoppgave

Lag et program som bruker en sensor du kobler på micro:biten for å samle inn data som du lagrer i en fil på micro:biten. Hva må du forandre på?

Hva må du forandre i programmet for å lage en fil med to kolonner der den første kolonnen er hvor lang tid det har gått før målingen er gjort?

Kobling B

Programmering og skaperverksted for 8. - 10. klasse

Kapittel 13

Innhold

Kapittel 13 – Modellering med Python (for de spesielt interesserte)

- Modellering og maskinl ring.....3
- Plotte graf.....4
- Line r regresjon.....5
- Ikke-line r regresjon.....6
- Korrelasjon.....9
- Valg av regresjonsmodell.....10

Modellering og maskinlæring med Python



Maskinlæring – hva er det?

Maskinlæring er en del av fagområdet kunstig intelligens, men det er mindre avansert enn man skulle tro. Den mest grunnleggende formen for maskinlæring er å lage matematiske modeller ved hjelp av enkle regresjoner slik som dere tidligere har gjort i Geogebra.

Etter hvert kan man bruke mer og mer avanserte modeller, og gjerne systemer som består av mange modeller som henger sammen med hverandre. Et eksempel på et slikt gigantisk system er værmodellering. Der hentes det inn allslags data om temperaturer, vindstyrke og lufttrykk fra mange steder i verden. Disse dataene brukes for å lage matematiske modeller, altså forskjellige matematiske funksjoner, slik dere har gjort tidligere. Videre kan meteorologene sette inn nye verdier i funksjonene sine, for å regne ut hvilket vær det er størst sannsynlighet for å bli i forskjellige områder på et senere tidspunkt. Hvordan henger dette sammen med gyldighetsområde og usikkerhet i målinger?

Mye modellering foregår uten at mennesker aktivt programmerer alle delene, programmene har «fått lov til» å lage sine egne matematiske modeller basert på forskjellige datasett. Dette kan gjøre det vanskeligere å vite hva som ligger bak avgjørelser som disse dataprogrammene gjør, da en del av algoritmene kan være vanskelig å få tilgang til. På en måte er det fint at maskiner kan gjøre dette for oss, på den andre siden kan det for eksempel kunne føre til problemer med diskriminering.

Diskriminering på grunn av maskinlæring har man sett flere eksempler på. Hvis et maskinlæringsprogram for ansiktsgjenkjenning får tilgang til datasett som nesten bare består av hvite menn, vil dette programmet bli mye dårligere på å kjenne igjen ansiktene til kvinner og ikke-hvite mennesker. Dette har man kunnet merke på enkelte mobilmerkens ansiktsgjenkjenning for å låse opp telefonen. Det er også funnet eksempler på diskriminering i ansettelser, for eksempel der et maskinlæringsprogram foreslo Amazon å heller ansette menn enn kvinner, basert på inndata om at det var veldig mange flere menn enn kvinner ansatt der i utgangspunktet. Les gjerne mer her:

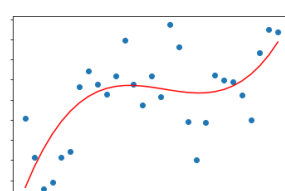
<https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scrapes-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G>



Fra data



Via regresjon



Til modeller

Lage graf med Python

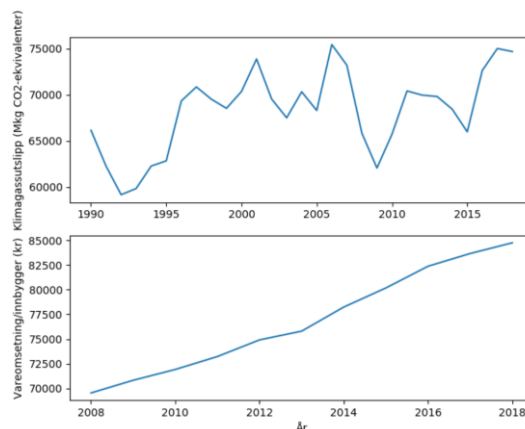


Hvorfor Python?

Dere kan allerede tegne grafer med Geogebra, så hva er poenget med å bruke Python i stedet?

I Geogebra kan vi plote en funksjon, eller legge inn punkter og få plottet en graf fra punktene. Siden vi må skrive inn alle punktene for hånd, kan det være tungvint dersom vi har veldig mange målinger. Dette gjelder spesielt om vi har ferdige filer med mange målinger. I Python kan man lage sub-plot, det vil si bilder med flere grafer som viser forskjellige målinger i samme bilde. Men hovedgrunnen til å bruke Python til plotting av grafer, er at det er begynnelsen på å bruke statistikk for å analysere data. Det er slik forskere jobber når de har samlet inn data!

I Python kan du bruke data som ligger i ferdige filer. Du slipper å skrive dem inn! Det som er viktig er at filene er riktig type. Filene kan bare inneholde tall, og lagres i tekstformat (.txt) for de eksemplene vi ser på her.



Plotting av grafer i Python består av seks deler:

1. Importere pylab-biblioteket i programmet
2. Importere data fra en .txt-fil
3. Legge dataene i en tabell (array) for x-verdiene og en for y-verdiene
4. Skrive hvilke data du skal plote
5. Skrive inn tittel og merkelapper på aksene
6. Få plottet til å vises på skjermen

I tillegg kan du skrive inn hvordan grafen skal se ut med for eksempel farge, linjetykkelse og linjestil.

`from pylab import *` → Importerer det nødvendige biblioteket

`plot(x, y)` → Lager et plot av alle verdiene i tabellene x og y

`show()` → Viser figuren av plottet på skjermen

`tabell = loadtxt("data.txt")` → Importerer filen data.txt og legger den inn i en tabell ved navn tabell

`x = tabell[:,0]`
`y = tabell[:,1]` → Plukker alle x-verdiene fra tabellen og legger dem i en tabell som heter x. Tilsvarende for y-verdiene.

`title("Tittel")`
`xlabel("x-verdier")`
`ylabel("y-verdier")` → Lager tittel og merkelapper til aksene.

Oppgaver

1. Bruk kodelinjene på venstre side for å lage et program som plottet en graf fra en fil som heter data.txt.
2. Skriv en beskrivelse for hver del av programmet ditt uten å se på denne siden. Legg dem gjerne inn som kommentarer i koden din. Sammenlign så beskrivelsene dine med de på denne siden. Hvilke forskjeller fant du?
3. Finn et valgfritt datasett fra statistisk sentralbyrå, og bruk Python til å lage en graf av dataene. Hva må du endre i eksempelet på denne siden?

Lineær regresjon med Python



Hvorfor Python?

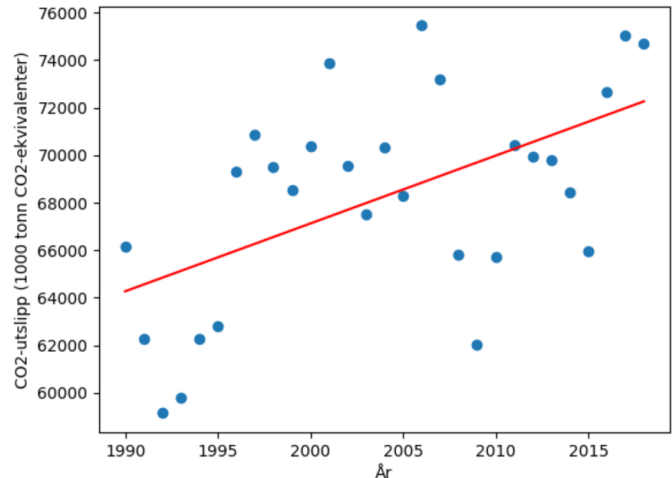
Dere har allerede utført mange lineære regresjoner med Geogebra, så hva er poenget med å bruke Python i stedet?

I Geogebra må vi skrive inn alle målingene våre for hånd, eller kopiere fra f.eks. Excel, og da kan det være vanskelig å gjøre en regresjon dersom vi har mye data. Tenk deg at du har 100 målinger, da tar det veldig lang tid å skrive inn alle målingene i regnearket i Geogebra. Det kan fort bli en feil eller to underveis også. Tenk om du har enda flere målinger, kanskje flere enn 1000! Da blir det ganske kjedelig å skrive inn alle tallene.

I Python kan du bruke data som ligger i ferdige filer. Du slipper å skrive dem inn! Det som er viktig er at filene er riktig type. Slik som med plotting av grafer i Python.

En annen fordel med å bruke Python, er at det er et programmeringsspråk som blir brukt av forskere som jobber med modellering. Andre forskere kan bruke andre programmeringsspråk som er spesialtilpasset akkurat den modelleringen de jobber med, men de følger de samme prinsippene som Python. Så dersom du kan modellere med Python, er det enklere å modellere i et annet programmeringsspråk enn om du bare har brukt Geogebra tidligere.

Samlet klimagassutslipp for Norge 1990 – 2018



Regresjon i Python består av seks deler:

1. Importere de bibliotekene som trengs
2. Importere data fra en .txt-fil
3. Legge dataene i en tabell for x-verdiene og en for y-verdiene
4. Lage en programmeringsfunksjon som sier at vi skal bruke en lineær modell i regresjonen
5. Utføre regresjonen, og vise på skjermen hva den matematiske modellen blir, stigningstallet (a) og skjæringspunktet med y-aksen (b)
6. Tegne grafen med resultatet – tilpass koden for plotting av graf
 - Plotte punktene med et scatterplot
 - Plotte regresjonsgrafen – hvordan få programmet til å regne ut y-verdiene for modellen?

Puslespilloppgave

Sorter kodelinjene etter numrene i tekstboksen over, om regresjon i Python. Da vil du få den riktige rekkefølgen i programmet ditt.

- | | |
|---|---|
| 1. <code>co2data = loadtxt("co2utslipp.txt")</code> | 5. <code>x_verdier = co2data[:,0]</code>
<code>y_verdier = co2data[:,1]</code> |
| 2. <code>from pylab import *</code>
<code>from scipy.optimize import curve_fit</code> | 6. <code>def funksjon(x, a, b):</code>
<code> return (a * x) + b</code> |
| 3. <code>scatter(x_verdier, y_verdier)</code>
<code>plot(x_verdier, funksjon(x_verdier, a, b), color="red")</code>
<code>title("Samlet klimagassutslipp for Norge 1990 – 2018")</code>
<code>xlabel("År")</code>
<code>ylabel("CO2-utslipp (1000 tonn CO2-ekvivalenter)")</code>
<code>show()</code> | |
| 4. <code>koeffisienter, kovarianser = curve_fit(funksjon, x_verdier, y_verdier)</code>
<code>a, b = koeffisienter</code>
<code>print(a,b)</code> | |

Ikke-lineær regresjon med Python



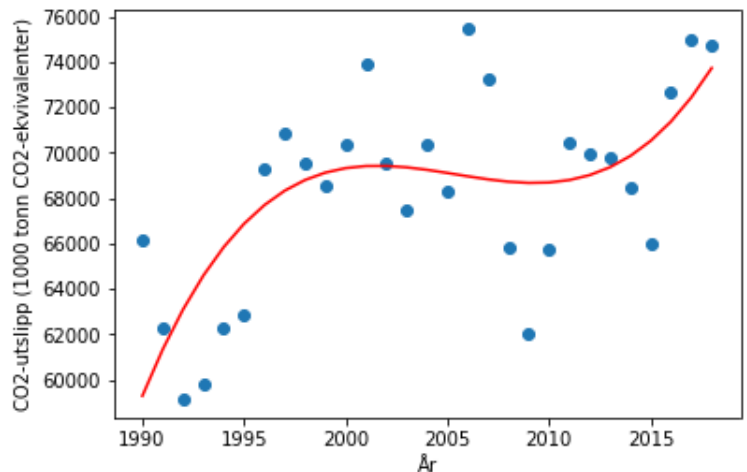
Forskjell på lineær og ikke-lineær regresjon i Python

Det er liten forskjell på hvordan programmene for lineær og ikke-lineær regresjon ser ut. Den viktigste forskjellen er å forandre på hvilken funksjonstype datasettet skal tilpasses. Det står en oversikt over noen funksjonstyper, og hvordan man skriver dem i python på de to neste sidene.

Men hvordan skal vi vite hvilken funksjonstype som passer til datasettet vårt? Går det an å finne det ut på forhånd? Svaret på det spørsmålet avhenger av datasettet ditt. Du kan lage et scatterplot av datasettet ditt, og se om du kjenner igjen en av funksjonstypene. Som regel er det likevel vanskelig å gjette seg til en funksjonstype på forhånd, og vi kan prøve oss fram med forskjellige funksjonstyper.

Programmeringen er veldig lik for lineær og ikke-lineær regresjon. Vi må bare endre den lineære funksjonen til en ikke-lineær funksjon. Da vil modellen vår tilpasse seg datasettet vårt best mulig, og vi får vite koeffisientene som bestemmer funksjonen for modellen vår. I tillegg må vi endre til riktig antall koeffisienter i programmet vårt. Se eksempelet under på hva som må endres.

Samlet klimagassutslipp for Norge 1990 – 2018



```
def funksjon(x, a, b): → def funksjon(x, a, b, c, d):  
    return (a * x) + b      return (a*x**3)+(b*x**2)+(c*x)+d
```

```
koeffisienter, kovarianser = curve_fit(funksjon, x_verdier, y_verdier)  
a, b = koeffisienter  
print(a,b)
```

```
koeffisienter, kovarianser = curve_fit(funksjon, x_verdier, y_verdier)  
a, b, c, d = koeffisienter  
print(a,b,c,d)
```

```
scatter(x_verdier, y_verdier)  
plot(x_verdier, funksjon(x_verdier, a, b), color="red")  
title("Samlet klimagassutslipp for Norge 1990 - 2018")  
xlabel("År")  
ylabel("CO2-utslipp (1000 tonn CO2-ekvivalenter)")  
show()
```

```
scatter(x_verdier, y_verdier)  
plot(x_verdier, funksjon(x_verdier, a, b, c, d), color="red")  
title("Samlet klimagassutslipp for Norge 1990 - 2018")  
xlabel("År")  
ylabel("CO2-utslipp (1000 tonn CO2-ekvivalenter)")  
show()
```

Vi bruker data fra statistisk sentralbyrå over klimagassutslipp i Norge for årene 1990 – 2018, og ønsker å lage en matematisk modell over utviklingen over tid. Vi prøver ut forskjellige funksjonstyper i regresjonsprogrammet, og ser hvilke modeller vi får.

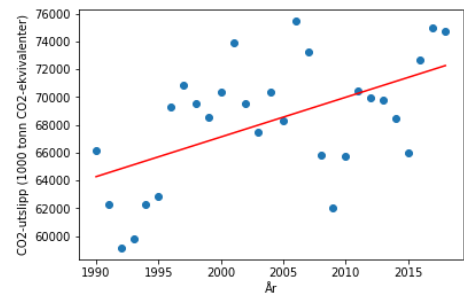
Vi bruker en lineær modell i programmet vårt

```
def funksjon(x, a, b):
    return (a * x) + b
```

Programmet gir oss den lineære funksjonen

$$f(x) = 286x - 504059$$

som den beste tilpasningen til datasettet vårt.



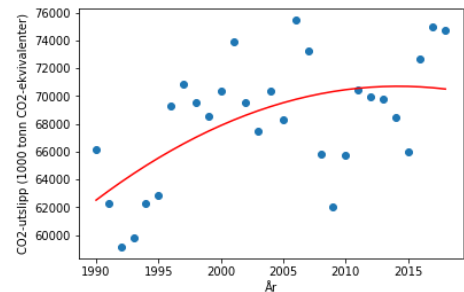
Vi bruker en andregradsmodell i programmet vårt

```
return (a*x**2)+(b*x)+c
```

Programmet gir oss andregradsfunksjonen

$$f(x) = -14x^2 + 56298 - 56627608$$

som den beste tilpasningen til datasettet vårt.



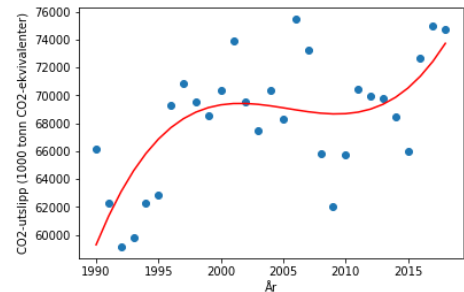
Vi bruker en tredjegradsmodell i programmet vårt

```
return (a*x**3)+(b*x**2)+(c*x)+d
```

Programmet gir oss tredjegradsfunksjonen

$$f(x) = 3x^3 - 19753x^2 + 39612328x - 26479502215$$

som den beste tilpasningen til datasettet vårt.



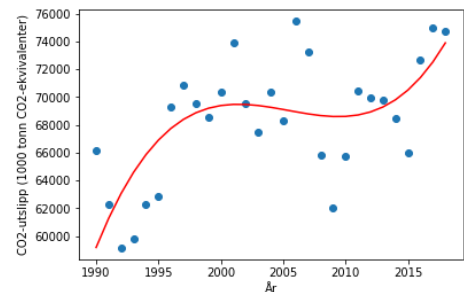
Vi bruker en fjerdegradsmodell i programmet vårt

```
return (a*x**4)+(b*x**3)+(c*x**2)+(d*x)+e
```

Programmet gir oss fjerdegradsfunksjonen

$$f(x) = 0,004x^4 - 30x^3 + 79206x^2 - 92029574x + 39188643912$$

som den beste tilpasningen til datasettet vårt.



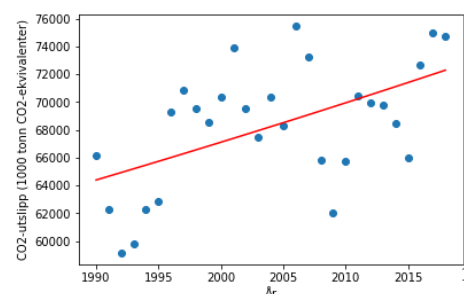
Vi bruker en eksponentiell modell i programmet vårt

```
return a * (b**x)
```

Programmet gir oss eksponentialfunksjonen

$$f(x) = 17 \cdot 1,004^x$$

som den beste tilpasningen til datasettet vårt.



Vi bruker en potensmodell i programmet vårt

```
return a*(x**b)
```

RuntimeError: Optimal parameters not found: Number of calls to function has reached maxfev = 600.

Når vi prøver å lage en potens-modell for CO₂-dataene våre, får vi det ikke til. For noen datasett vil noen funksjonstyper passe veldig dårlig. Da kan det hende at vi får en feilmelding i programmet, om at vi har brutt en tidsbegrensing. Da har det tatt programmet for lang tid å tilpasse modellen for oss, og vi får ikke noe resultat. Det kan være et tegn på at modellen (funksjonstypen) vi har valgt passer dårlig med datasettet vårt.

Vi ser på noen andre eksempler med et datasett for kumulativt antall Covid 19-smittede i Norge for uke 8 til 32 i 2020.

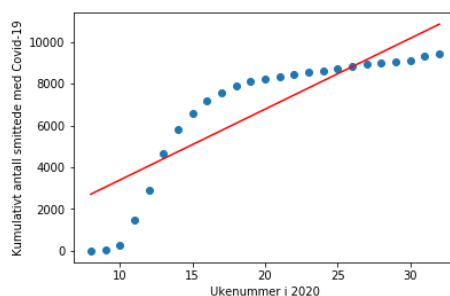
Vi bruker en potensmodell i programmet vårt

```
return a*(x**b)
```

Programmet gir oss potensfunksjonen

$$f(x) = 0,800 \cdot x^{1,00}$$

som den beste tilpasningen til datasettet vårt.



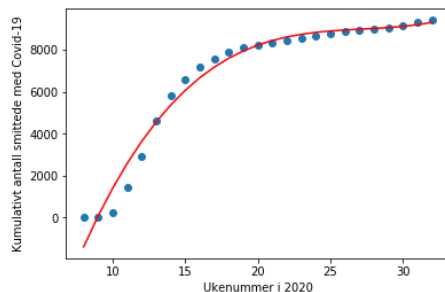
Vi bruker en tredjegradsmodell i programmet vårt

```
return (a * x**3) + (b * x**2) + (c * x) + d
```

Programmet gir oss tredjegradsfunksjonen

$$f(x) = 1,37x^3 - 112x^2 + 3080x - 19578$$

som den beste tilpasningen til datasettet vårt.



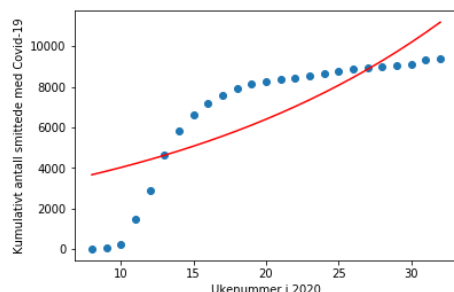
Vi bruker en eksponentialmodell i programmet vårt

```
return a * (b**x)
```

Programmet gir oss eksponentialfunksjonen

$$f(x) = 2521 \cdot 1,048^x$$

som den beste tilpasningen til datasettet vårt.



Nå vet vi hvordan vi kan bruke data for å tilpasse mange forskjellige funksjonstyper. Men hvordan skal vi vite hvilken modell som passer best med datasettet vårt? Må vi bare se hvilken funksjonstype som ser ut til å passe best med datasettet vårt, eller kan vi gjøre det på en mer objektiv måte? Bla om til neste side for å finne ut mer.



Korrelasjon i Python

Når vi skal beregne korrelasjonen, er det som regel Pearsons korrelasjonskoeffisient vi mener, og den kalles R . Det er også den vi skal bruke her. Vi er mest interessert i denne korrelasjonskoeffisienten opphøyd i andre, altså R^2 , fordi den verdien sier hvor stor andel av variasjonen i målingene/datasettet vårt som henger direkte sammen. Verdien kan være mellom null og en, og desto høyere verdi, desto større del av variasjonen i y kan forklares av variasjonen i x . R^2 oppgis som et desimaltall, men multipliserer vi det med 100 %, får vi svaret i prosent.

Den enkleste formen for korrelasjon, er mellom x og y -verdiene i et datasett. Da finner vi hvor stor del av variasjonen i y som kan forklares av variasjonen i x . Dette gjøres ved at Python foretar en lineær regresjon, slik at R^2 viser hvor god den lineære sammenhengen mellom x og y -verdiene er. Da er R^2 et mål på hvor godt den lineære funksjonen passer med datasettet vårt.

Det går også an å gjøre tilsvarende for ikke-lineære regresjoner. Da får vi et objektivt mål på hvilken modell som passer best med vårt datasett. Dersom man vil finne R^2 for en regresjon der $f(x)$ er en eksponentialfunksjon, begynner vi med å utføre selve regresjonen. Når den er ferdig, vet vi hva koeffisientene i eksponentialfunksjonen er, altså vet vi hva $f(x)$ er. Da bruker vi den estimerte funksjonen til å putte inn x -verdiene fra datasettet vårt, for så å regne ut de tilsvarende $f(x)$ verdiene. For å finne R^2 bruker vi y -verdiene fra datasettet vårt og $f(x)$ -verdiene som vi regnet ut ved å bruke den eksponentielle modellen vi fant. Da gjelder fortsatt at den modellen som gir en verdi nærmest $R^2 = 1,00$ er den beste modellen.

Korrelasjon i Python består av fem deler:

1. Importere bibliotekene som trengs
2. Importere data fra en .txt-fil
3. Legge dataene i en tabell (array) for x -verdiene og en for y -verdiene
4. Beregne korrelasjonskoeffisienten
5. Vise korrelasjonskoeffisienten på skjermen

Datasettene vi bruker kan vi for eksempel hente fra statistisk sentralbyrå. Hvordan man gjør dette står i opplegg 26. Eller kan man bruke en micro:bit for å samle inn data. Dersom vi har mange målinger, er det fint å lagre det som en fil på micro:biten. Hvordan dette gjøres står i kapittel 12.

```
from pylab import *  
from scipy.stats import pearsonr
```

→ Importerer de nødvendige bibliotekene.

```
data = loadtxt("filnavn.txt")
```

→ Legger inn data fra fil i en tabell.

```
x = data[:,0]  
y = data[:,1]
```

→ Legger x -verdiene fra første kolonne inn i en tabell og y -verdiene fra den andre kolonnen i en tabell.

```
R = pearsonr(x, y)[0]
```

→ Regner ut korrelasjonskoeffisienten.

```
print(R)  
print(R**2)
```

→ Skriver korrelasjonskoeffisienten og R^2 til skjermen. R^2 tilsvarer R^2 -verdien i Geogebra.

Oppgaver

1. Finn et valgfritt datasett fra statistisk sentralbyrå, og bruk Python til å finne korrelasjonen for datasettet. Hva må du endre i eksempelet på denne siden?
2. Hvordan kan man beregne korrelasjonen mellom to forskjellige datasett? Hvilke kriterier må være oppfylt for at det skal kunne gå an?

Valg av regresjonsmodell i Python



Vi kan ikke *bare* se på R^2 når vi skal velge modell (velge mellom de ulike funksjonstypene). Vi bør nemlig ha en idé utfra teori, om hvilken type modell som burde fungere best. Det er for eksempel slik at en polynomfunksjon vil passe bedre og bedre, desto høyere grad den er. Det vil si at en fjerdegradsfunksjon vil alltid passe like godt eller bedre enn en tredjegradsfunksjon. Eller at en andregradsfunksjon vil alltid passe like godt eller bedre enn en førstegradsfunksjon, som er det samme som en lineær funksjon.

Regresjon med korrelasjon i Python består av sju deler:

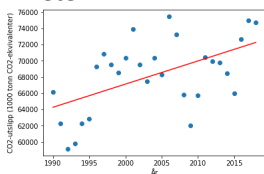
1. Importere bibliotekene som trengs
2. Importere data fra en .txt-fil
3. Legge dataene i en tabell for x-verdiene og en for y-verdiene
4. Lage en funksjon som sier hvilken modell som skal brukes i regresjonen
5. Utføre regresjonen, og vise på skjermen hva den matematiske modellen blir
6. Tegne grafen med resultatet
7. Regne ut R og R^2 og skrive dem ut til skjermen

```
from pylab import *  
from scipy.optimize import curve_fit  
from scipy.stats import pearsonr
```

```
R = pearsonr(y_verdier, funksjon(x_verdier, a, b, c))[0]  
print(R)  
print(R**2)
```

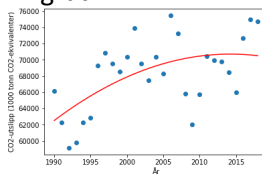
Nå kan vi sjekke hvilken matematisk modell som beskriver datasettene våre best. Vi sjekker hvilken modell som passer best for både CO₂-utslippene og kumulativt antall covid-19 smittede. Sjekk om du får samme verdier for de ulike modellene.

Lineær



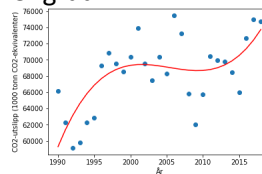
R = 0,5428
 $R^2 = 0,2947$

2. grad



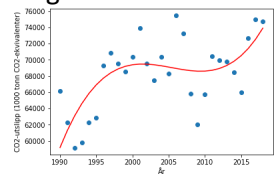
R = 0,5780
 $R^2 = 0,3340$

3. grad



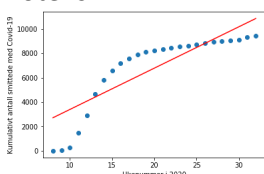
R = 0,6695
 $R^2 = 0,4483$

4. grad



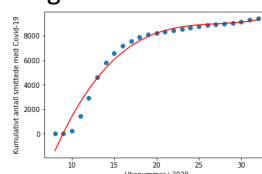
R = 0,6710
 $R^2 = 0,4502$

Potens



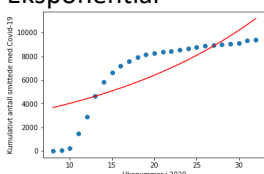
R = 0,8730
 $R^2 = 0,7620$

3. grad



R = 0,9869
 $R^2 = 0,9740$

Ekspontial



R = 0,8004
 $R^2 = 0,6407$

Diskusjonsoppgaver

1. Hvilken modell beskriver CO₂-utslippene best? Begrunn svaret.
2. Hvilken modell beskriver kumulativt antall covid-19 smittede best? Begrunn svaret.
3. Kan det potensielt finnes andre modeller som beskriver datasettene våre bedre?
4. Dersom du har laget et program som allerede utfører en regresjon, hva må du legge inn for at det også skal finne og skrive ut R og R^2 ?