hvorfor_programmere

November 8, 2019

1 Programmering

1.1 Hvorfor skal elevene lære programmering?

1.1.1 Generelle grunner

- Grunnleggende forståelse for IKT (almenndannelse)
- Viktig nasjonal interesse med økt rekruttering til IKT

1.1.2 Pedagogiske og didaktiske grunner

- Lettere å arbeide med realistiske problemstillinger
- Utfordrer elevene til å tenke systematisk
- Utvikler logisk forståelse og tankegang

1.2 Vanlig faglig problemstilling i grunnskolen:

- Vei fart tid
- Lineære funksjoner

Litt kjedelig

[figur 1][./bilder/vei_fart_tid_linear.png]

- Ikke veldig spennende
- Hva skjer når farta ikke er konstant?
- Svar med klassisk matematikk krever gjerne integrasjon / differensiallikninger
- Programmering kan gjøre problemet tilgjengelig for (flinke) elever tidlig.
- Veldig plagsomt at klassisk matematikk ikke gjør problemet mer tilgjengelig

Vi skal se på en løsning med konstant akselerasjon senere - fra dette kan man også bygge modeller der akselerasjonen varierer!

1.3 Andre aktuelle anvendelser

- Programmere og kalibrere måleinstrumenter i naturfag og kunst og håndtverk (arduino, raspberry pi)
- Programmere og kalibrere aktuatorer i naturfag og kunst og håndtverk (arduino, raspberry pi)

- Jobbe med store datasett (eksempler på datasett: https://www.kaggle.com/datasets)
- Muligheter for å kombinere naturfag, matematikk, kunst og håndtverk i større grad
- Mer tverrfaglig arbeid!
- Programmering som verktøy i alle fag

2 Et enkelt program

Å se på dette programmet har to hovedmål * Vite hvordan du kan kjøre python-programmer * Vite hvordan du skriver ut data (tall, tekst etc) fra python ut til skjermen * Bli kjent med datatypen for tekst: *strenger*

[1]:
```
print('Hello, world!')
```

```
Hello, world!
```

- Funksjonen `print` skriver teksten `'Hello, world!'` ut til skjermen.
- Hermetegnene forteller Python at innholdet er tekst (en streng).
- I programmering kaller vi data som skal være tekst for **strenger**.

3 Et program som behandler tall

[4]:
```
# Regn ut arealet av et pararellogram med lengde l og bredde h
l = 8
h = 3.5

A = l*h
print(A)
```

```
28.0
```

Programmet regner ut arealet av et pararellogram - vi kan tenke på programmet som en oppskrift for dette.

- Den øverste linjen er en kommentar. Den blir ignorert av datamaskinen og skal gjøre programmet lettere å lese for mennesker
- `l` og `h` er *variabler* av typen `int` (heltall) og `float` (et "desimaltall"). En variabel kan du tenke på som en boks som inneholder data, og navnet på variabelen gir oss en måte å få tak i dataene
- Til slutt regner vi ut produktet `l*h`, lagrer det i variabelen A og skriver det ut til skjermen

Gode variabelnavn og gode kommentarer er viktig for lesbar kode. Les mer om gode variabelnavn og kommentarer her: http://lstor.me/kommentarer/.

4 Forbedring av programmet

Programmet har en svakhet: hver gang du vil endre dimensjonene på pararellogrammet, må koden endres.

Bedre løsning: les inn tallene fra brukeren

[2]:
```
# Regn ut arealet av et pararellogram med lengde l og bredde h
l = input('Skriv inn lengden på pararellogrammet: ')
h = input('Skriv inn høyden til pararellogrammet: ')

# Gjør strengene `l` og `h` om til desimaltall
l = float(l)
h = float(h)

A = l*h
print(A)
```

```
Skriv inn lengden på pararellogrammet: 4.0
Skriv inn høyden til pararellogrammet: 5,0
```

```
---------------------------------------------------------------------------

ValueError                                Traceback (most recent call last)

<ipython-input-2-56b59ba95142> in <module>
  5 # Gjør strengene `l` og `h` om til desimaltall
  6 l = float(l)
----> 7 h = float(h)
  8
  9 A = l*h


ValueError: could not convert string to float: '5,0'
```

5 Oppgaver for å komme i gang med python

5.1 Oppgave 1

Skriv et program `hei_verden.py` som skriver teksten `Hei, Verden!` ut til skjermen.

5.2 Oppgave 2

Skriv et program som regner ut arealet til en trekant med lengde h og grunnlinje g. Du kan velge verdier for h og g selv.

5.3 Oppgave 3

Gjør om programmet over slik at det leser inn lengden h og grunnlinja g, regner ut arealet til trekanten og skriver det ut til skjermen.

5.4 Oppgave 4: Lag et didaktisk verktøy

Du skal introdusere elevene dine for funksjoner for første gang, og ønsker å leke den (for lærere) kjente leken "gjett regelen" der elevene sier noen tall, du beregner en tilhørende verdi med en lineær funksjon, og elevene skal prøve å formulere hva funksjonen gjør med tallene.

Bruk en lineær funksjon $y = ax + b$ der du selv bestemmer tallene a og b som holdes hemmelig for elevene. Programmet skal lese inn en verdi for x og gi ut verdien for y.

5.5 Oppgave 5

En båt seiler med konstant hastighet $v = 43.2\text{km/t}$. Fra havn A til havn B. Ved tiden $t = 0$ var båten 180 km unna havn A.

Skriv et program som regner ut strekningen $s_A(t)$ fra havn A etter tiden t.

5.6 Oppgave 6

Modifiser programmet ditt i oppgave 3 slik at det leser inn farten v, strekningen s_0 ved tiden $t = 0$ og tiden t_0. Programmet skal regne ut strekningen s etter tiden t og skrive det ut til skjermen.

5.7 Oppgave 7: Bruk en komplisert formel

Programmer kan gjøre det lettere å beregne verdien til kompliserte utrykk. Et objekt som beveger seg gjennom en væske eller en gass med tetthet ρ målt i kg/m^3 og dynamisk viskositet μ målt i kg/ms, opplever en drag-kraft

$$F_d = \frac{1}{2} C_d A \rho v^2,$$

der C_d er *drag-koeffisienten* som avhenger av objektets form, overflate osv., og A er arealet av objektets tverrsnitt målt i kvadratmeter. Du skal beregne F_d når du har fått oppgitt følgende data:

$$\begin{aligned} \mu &= 1.802 \cdot 10^{-5} \\ \rho &= 1.225 \\ C_d &= 0.8 \\ A &= 0.8 \end{aligned}$$

a) Lag et program `drag_koeffisient.py` som beregner drag-koeffisienten F_d.

b) Gjør samme beregningen i et ipython-shell.

6 Noen konkrete anvendelser

Vi kan begynne med å se på hvordan man kan gjøre beregninger knyttet til bevegelse under konstant akselerasjon a. Det finnes en formel programmet kan testes mot:

$$s(t) = s_0 + v_0 t + \frac{1}{2} a t^2$$

6.0.1 Eksempel på algoritmisk tenkning

Får å beregne strekningen s, må vi bryte problemet opp i mange delproblemer der vi beregner hastigheten v fra akselerasjonen a i mange små tidssteg, og i hvert tidssteg oppdaterer vi s med formlene

$$v(t_n) = 10 - at_n$$

$$s_{n+1} = s_n + \Delta t \cdot v(t_n)$$

[7]:
```
def modell_strekning(T, s_0, v):
    """
    modell-problem med kjent fart

    idé: Hvis man kjenner s(t) og v(t), kan s(t + dt) tilnærmes slik:
    s(t + dt) = s(t) + dt*v(t)
    Jo mindre tidssteg dt, jo mer nøyaktig blir tilnærmingen.
    """

    dt = 0.1    # tidssteg
    t = [0]     # første tidspunkt
    s = [s_0]   # start-strekning
    k = 0       # telle-variabel

    while t[k] < T:
        s_next = s[k] + dt*v( t[k] )
        t_next = t[k] + dt

        s.append(s_next)
        t.append(t_next)
        k = k + 1

    return t, s
```

6.0.2 Tidsperspektiv på slike prosjekter

Min antakelse: én til to uker hvis elevene er kjent med bruk av løkker.

6.0.3 Læringsverdi

- Algoritmisk tenkning
- Vurdering av modellen, hvor stor må Δt være, etc
- Går det an å utvide modellen? F.eks i tilknytning til Newtons lover i fysikk.
- Hva skjer dersom vi utvider modellen og bruker den på en fjær som svinger?

6.0.4 Behandling av store datasett

Vi skal lese inn **digre** datasett. Regneark sliter med å behandle så store mengder data, og det er også ganske knotete å holde på med. Vi bruker pakken pandas. Dere kan bruke conda til å installere den. Skriv dette i terminalen for å installere nødvendige pakker

```
conda install pandas
conda install xlrd
```

I mappen `programmering_uskole/programmer/eksempler_datalogging` ligger det en exel-fil `temperatur_data.xlsx` med dummy-temperaturer.

Vi kan bruke pandas til å både skrive data til excel-filer, og for å laste data fra excel-filer inn i python. Du kan senere ta en titt på hvordan dette er gjort i filene `programmering_uskole/programmer/eksempler_datalogging/makedata.py` og `programmering_uskole/programmer/eksempler_datalogging/read_data.py`

6.1 Andre anvendelser

Biologi: Simulere bestander eller økosystemer. Start med eksponentiell vekst, introduser bæreevne, introduser sesonger, introduser rovdyr, introduser tilfeldige sykdommmer. Store muligheter!

Fysikk: Vei-fart-tid og newtons lover.

Tverrfaglige prosjekter: Naturfag, kunst og håndtverk, matematikk. Utforske bruk av arduino og raspberry pi med måleinstrumenter og aktuatorer. Sammenføyning som plastsveising og lodding aktuelt.

Bruk av pakken pandas kan være aktuelt i andre fag? (databehandling ut over excel)

Personlig mening: synes det er lettere å bruke pandas enn excel ...

Viktig å lære en del grunnleggende emner først. Men aller viktigst å ha det gøy!

7 Viktigst: Ha det gøy. La elevene ha det gøy med koding

Bruk programmering som et didaktisk verktøy. La elevene utvikle ferdighetene gradvis. Tør å la elevene formulere egne prosjekter, og tilby både utfordrende og enklere prosjekter elevene kan arbeide med.