

Programmering i skolen

Et kræsjkurs i Python for
realsfagslærere: Del 2



simula **kode**skolen



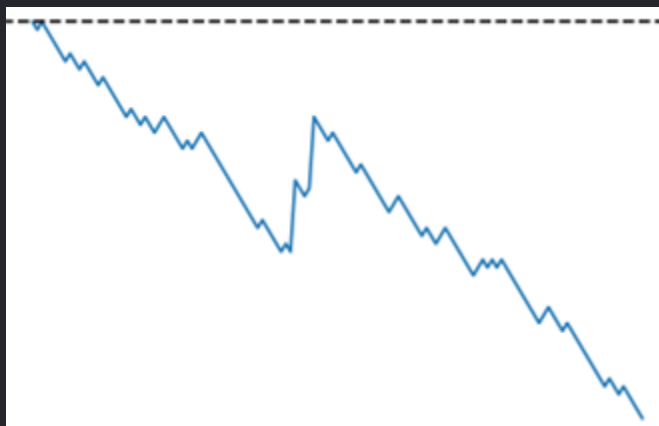
Med programmering kan man enkelt simulere problemer fra sannsynlighetsteori



S1: Bruke digitale verktøy til å simulere og utforske utfall i stokastiske forsøk, og forstå begrepet stokastiske variable

S2: Simulere utfall i, utforske og tolke ulike statistiske fordelinger, og gi eksempler på reelle anvendelser av disse fordelingene

Med programmering kan man enkelt simulere problemer fra sannsynlighetsteori



```
13 def pengespill():
14     total = kast_2d6()
15     if total <= 8:
16         return -10
17     elif total <= 11:
18         return 10
19     else:
20         return 90
21
22 # Telle variabel
23 penger = 500
24
25 # Liste for å huske resultatene over tid
26 pengehistorikk = []
27 pengehistorikk.append(penger)
28
29 # Løkke for å gjenta spillet helt til vi går tom for penger
30 while penger > 0:
31     penger += pengespill()
32     pengehistorikk.append(penger)
33
34 n = len(pengehistorikk)
35 # Plot resultatet
36 plot(pengehistorikk)
37 axhline(500, color='black', linestyle='--')
38 show()
39 print(f'Du spilte {n} ganger før du gikk tom for penger')
40
```



Med programmering kan man enkelt simulere problemer fra sannsynlighetsteori

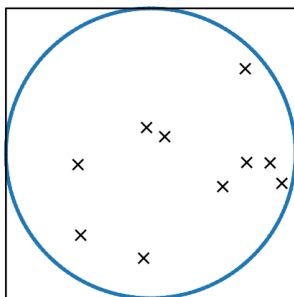
S1: Bruke digitale verktøy til å simulere og utforske utfall i stokastiske forsøk, og forstå begrepet stokastiske variable

R2: Utvikle algoritmer for å beregne integraler numerisk, og bruke programmering til å utføre algoritmene

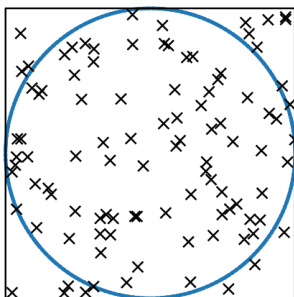
Med programmering kan man enkelt simulere problemer fra sannsynlighetsteori



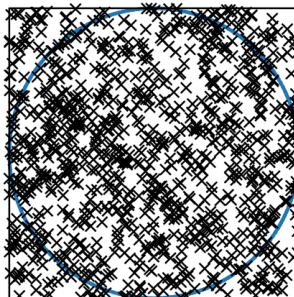
10 kast



100 kast



1000 kast



```
8 from random import uniform
9 from math import sqrt
10
11 antall_kast = 10
12 antall_treff = 0
13
14 for kast in range(antall_kast):
15     # Kast en pil
16     x = uniform(-1, 1)
17     y = uniform(-1, 1)
18
19     # Sjekk om den traff
20     avstand = sqrt(x**2 + y**2)
21     if avstand <= 1:
22         antall_treff += 1
23
24
25 # Estimer pi basert på kastene
26 pi = 4*antall_treff/antall_kast
27
28 # Skriv ut resultater
29 print(f"Antall kast: {antall_kast}")
30 print(f"Antall treff: {antall_treff}")
31 print(f"Estimert pi: {pi}")
32
```

Programmering kan ta i bruk numeriske tilnærminger av den deriverte til å simulere fysiske prosesser



R1: Analysere og tolke ulike funksjoner ved å bruke derivasjon

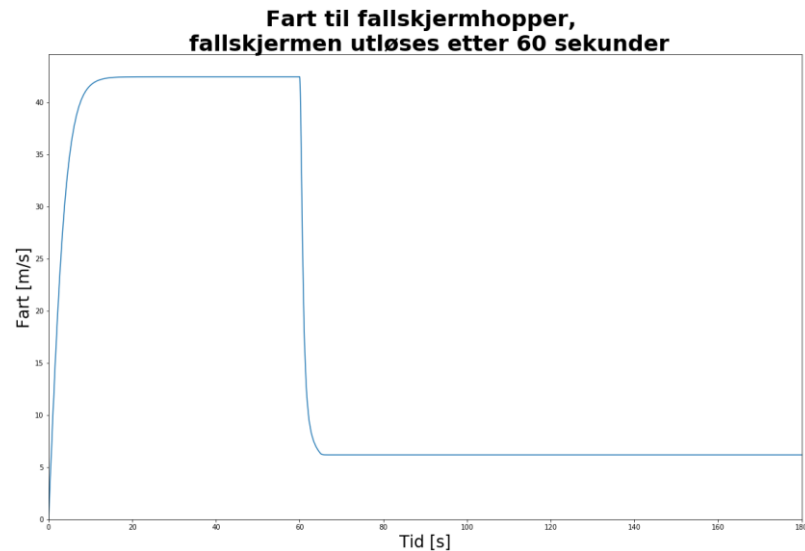
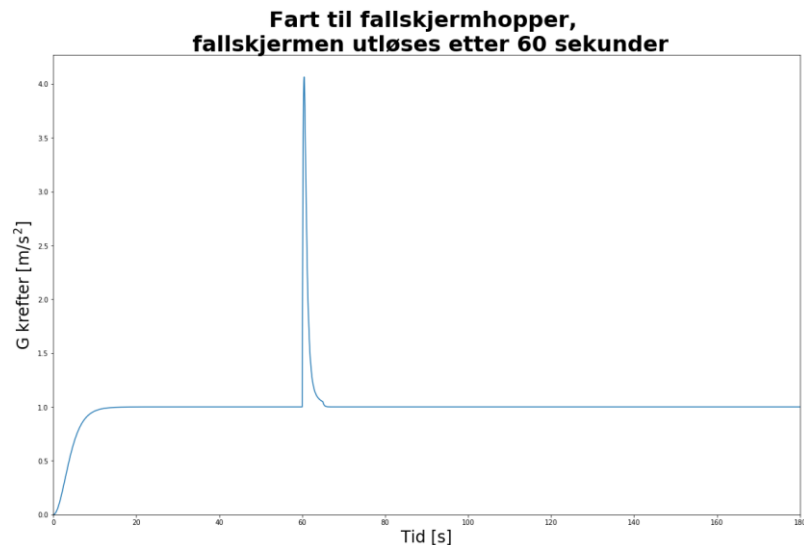
S1, R2: Utforske rekursive sammenhenger ved å bruke programmering og presentere egne framgangsmåter

Programmering kan ta i bruk numeriske tilnærminger av den deriverte til å simulere fysiske prosesser

```
30 # Simuler de første 60 sekundene
31 for i in range(0, 60/dt):
32     t[i+1] = t[i] + dt
33     v[i+1] = v[i] + a(v[i])*dt
34     gforces[i] = 1 - a(v[i])/g
35
36 # Simulerer de neste 5 sekundene
37 for i in range(60/dt, 65/dt):
38     C += (C_p-C)/(5/dt)
39     A += (A_p-A)/(5/dt)
40
41     t[i+1] = t[i] + dt
42     v[i+1] = v[i] + a(v[i])*dt
43     gforces[i] = 1 - a(v[i])/g
44
45 # Simuler de siste 115 sekundene
46 for i in range(65/dt, 180/dt):
47     t[i+1] = t[i] + dt
48     v[i+1] = v[i] + a(v[i])*dt
49     gforces[i] = 1 - a(v[i])/g
50
```



Programmering kan ta i bruk numeriske tilnærminger av den deriverte til å simulere fysiske prosesser



Med programmering kan vi simulere forskjellige typer populasjonsvekst



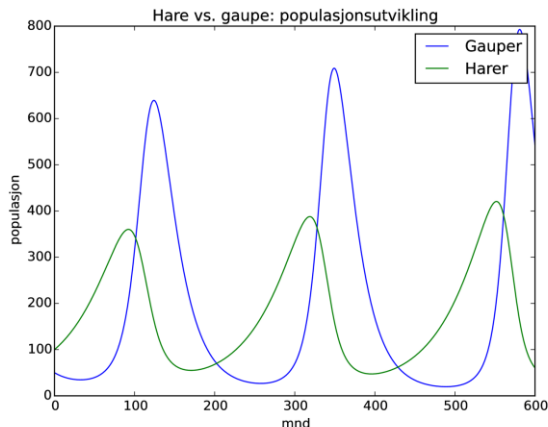
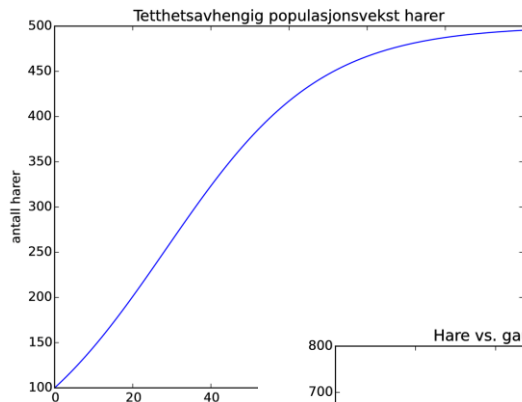
Kjerneelement:

Elevene skal forstå, skape og bruke teknologi, inkludert programmering og modellering, i arbeid med naturfag.

Naturfag vg1: Vurdere og lage programmer som modellerer naturfaglige fenomener

S2, R2: Utforske rekursive sammenhenger ved å bruke programmering og presentere egne framgangsmåter

Med programmering kan vi simulere forskjellige typer populasjonsvekst

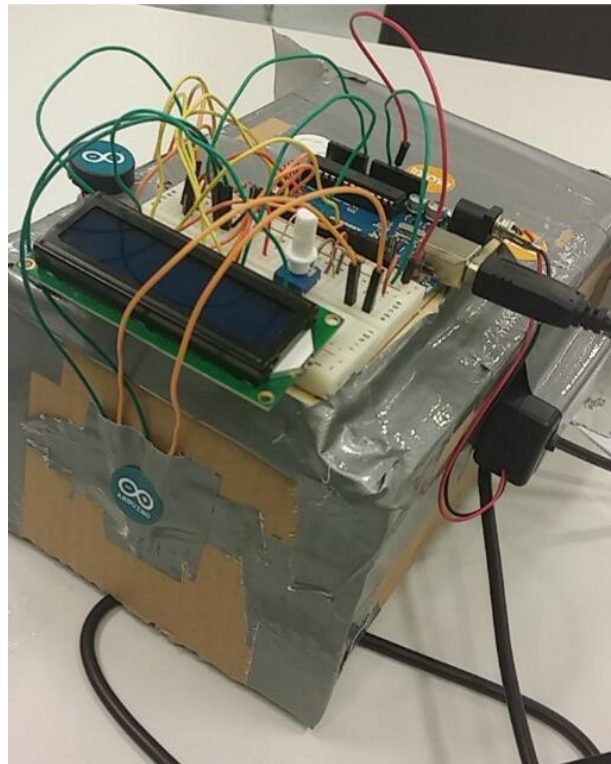


```
1 import numpy as np
2 n = 12*50      #antall tidsintervaller
3 y0 = 100       #antall byttedyr når vi starter
4 x0 = 50        #antall rovdyr når vi starter
5 index_set = range(n+1)
6
7 x = np.zeros(len(index_set))
8 y = np.zeros(len(index_set))
9
10
11 a = 0.05       # dødsrate gauper
12 b = 0.0003     # reproduksjonsrate gauper
13
14 c = 0.02       # vekstrate harer
15 d = 0.0001     # dødsrate harer
16
17
18 y[0] = y0
19 x[0] = x0
20 for k in index_set[:-1]:
21     #print y[k]
22     y[k+1] = y[k] + c*y[k] - d*y[k]*x[k]
23     x[k+1] = x[k] - a*x[k] + b*x[k]*y[k]
```

Arduino og micro:bit kan la elever lage programmer som regner på data hentet fra sensorer

Elevene skal forstå, skape og bruke teknologi, inkludert programmering og modellering, i arbeid med naturfag.

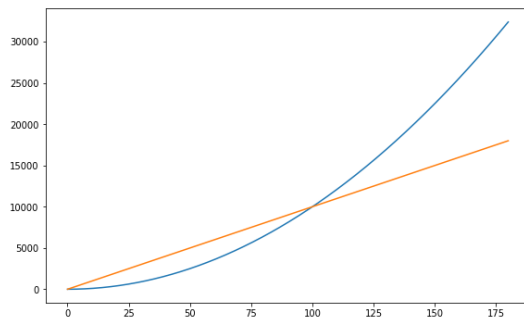
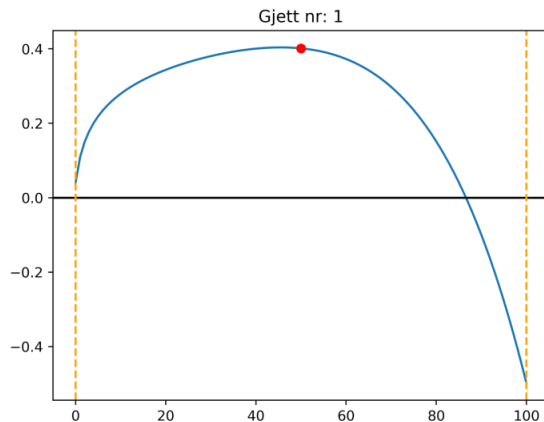
Naturfag VG1: utforske en selvvalgt naturfaglig problemstilling, presentere funn og argumentere for valg av metoder



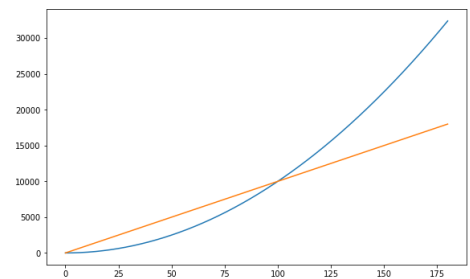
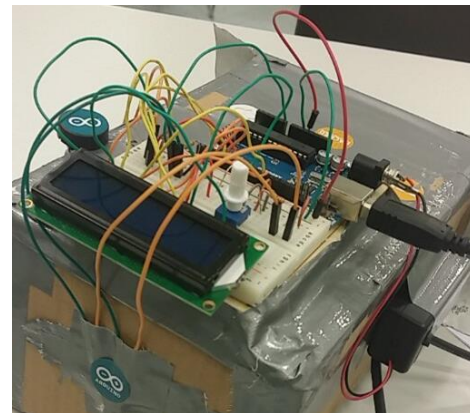
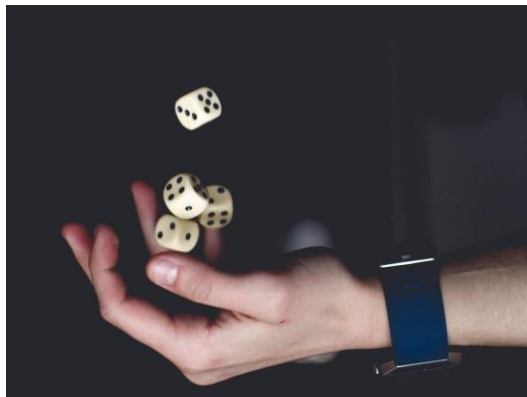
Med programmering kan vi finne løsninger på likninger vi ikke kan løse for hånd

1T: Formulere og løse problem ved hjelp av algoritmisk tenking, ulike problemløsningsstrategier, digitale verktøy og programmering

1T: Utforske strategier for å løse likningar, likningssystem og ulikskapar og argumentere for tenkjemåtene sine



Som vi har sett har programmering mange anvendelser i realfag



I dag: Videre Python og et opplegg til klasserommet

- Funksjoner
- Plotting
- Prosjekt: Programmere en likningsløser som bruker *halveringsmetoden*

