Mattespill

Introduksjon

I denne leksjonen vil vi se litt nærmere på hvordan Python jobber mec
hvordan vi kan gjøre ting tilfeldige.

```
$ python3 mattespill.py
Hva er 4 ganger 10?
40
Ja, svaret er 40
Hva er 5 ganger 2?
10
Ja, svaret er 10
Hva er 3 ganger 9?
39
Nei, det riktige svaret er 2
Hva er 2 ganger 11?
22
Ja, svaret er 22
Hva er 9 ganger 4?
35
Nei, det riktige svaret er 3
Du klarte 3 av 5
```

Steg 1: En kalkulator!

Python er, som de fleste programmeringsspråk, veldig glad i å regne.

Sjekkliste

- [ ] Lag et nytt IDLE-vindu ved å velge `File > New File`.

- [ ] Vi begynner med å utforske hvordan vi kan bruke Python til å re
programmet:

```
print(2 + 3)
print(17 - 8)
print(3 * 4)
print(22 / 7)
```

Lagre programmet med navnet `kalkulator.py` og kjør det.

- [ ] Når programmet kjører vil det skrive ut 4 tall. Kjenner du igjen c
betyr? Endre gjerne på programmet og kjør det flere ganger til c
gange og dele.

Steg 2: Vi kaster terning

For å lage et mattespill vil vi bruke tilfeldige tall. Tilfeldige tall blir omt

Sjekkliste

- [ ] For å lage tilfeldige tall skal vi bruke en ny Python-funksjon som
random som betyr tilfeldig, og `int` en forkortelse for *integer* so
grunnpakken til Python, men ligger i stedet i et bibliotek kalt `ra`
biblioteket. Lag et nytt program, `terning.py` som ser slik ut:

```
from random import randint
print(randint(1, 6))
```

Når du kjører programmet vil det skrive ut et tilfeldig tall mellor
tallet seg?

- [ ] Prøv selv å forandre programmet slik at det skriver ut tilfeldige

- [ ] Programmet `terning.py` viser hvordan vi kan late som om vi sl
om vi slår to terninger, og ser summen av dem?

- [ ] Vi kan også utvide programmet slik at det slår terning mange g
for eksempel utvider programmet ditt som følger vil det se ut s

```
from random import randint

for i in range(25):
    print(randint(1, 6) + randint(1, 6))
```

Steg 3: En liten matteprøv

Vi kan nå bruke tilfeldige tall til å lage et enkelt mattespill.

Sjekkliste

- [ ] Lag et nytt program som heter `mattespill.py`. Vi begynner me
sammen:

```
from random import randint

tall1 = randint(2, 12)
tall2 = randint(2, 12)
```

- [ ] Vi vil nå at Python skal gi oss en matteoppgave, kan vi bli spurt

denne linjen nederst i programmet ditt:

```
print('Hva er ' + tall1 + ' ganger ' + tall2 + '?')
```

Hva skjer når du prøver å kjøre programmet?

Du husker kanskje at vi kan bruke `+` for å sette sammen tekst? linjen over prøver vi å bruke `+` på både tekst og tall, og da skjø

For å fortelle Python at her vil vi egentlig sette sammen tekst m tekststrenger. Endre den siste linjen slik at den ser slik ut i stede

```
print('Hva er ' + str(tall1) + ' ganger ' + str(tall2) +
```

Virker programmet ditt bedre nå? Prøv å kjøre programmet flere

Neste steg er at vi vil kunne svare på mattestykket. Til dette bru

```
svar = input()
```

nederst i programmet.

Vi skal nå få programmet til å sjekke at vi har svart riktig. For å Disse kan sjekke om noe er sant, og vi vil bruke dem for å sjekk svaret. Legg til en **if-test** nederst i programmet ditt slik som de

```
from random import randint

tall1 = randint(2, 12)
tall2 = randint(2, 12)

print('Hva er ' + str(tall1) + ' ganger ' + str(tall2) +
svar = input()

if svar == tall1 * tall2:
    print('Ja, svaret er ' + svar)
else:
```

```
    print('Nei, det riktige svaret er ' + str(tall1 * ta
```

Pass på at som i **for-løkker** må du skyve koden i **if-testen** inn

- [ ] Kjør programmet ditt. Virker det? Hva skjer om du svarer feil på

  Hmm ... det er et problem med programmet vårt. Programmet s

Bugs

Dette er et eksempel på noe vi kaller en bug i et program. Program
ment og forventet. Selve ordet *bug* betyr insekt, og grunnen til at
faktisk et problem at insekter fløy inn i datamaskiner og ødela prog

Sjekkliste

- [ ] Problemet med programmet vårt er ikke helt lett å finne, men s
  bety at `svar` aldri er helt lik `tall1 * tall2`. Igjen er problemet

  Når vi bruker `input` til å lese inn `svar` vil dette alltid være teks
  vi kan bruke `str` for å gjøre om tall til tekst kan vi bruke `int` fo
  forkortelse for *integer* som betyr heltall).

  Endre `if`-linjen i programmet ditt til

  ```
  if int(svar) == tall1 * tall2:
  ```

  Virker programmet bedre nå?

Steg 4: Telle riktige svar

Det er litt kjedelig å alltid starte programmet på nytt. La oss prøve å s

Sjekkliste

- [ ] Hvordan kan vi lage en løkke slik at programmet stiller oss for e
litt selv før du går videre.

- [ ] Til slutt vil vi at programmet også skal telle hvor mange riktige
variabel som vi for eksempel kan kalle `ant_rett`. Når vi begynn
svart noenting enda. Hver gang vi svarer riktig kan vi så øke ve
melding til slutt om hvor mange riktige svar spilleren klarte vil p

```
from random import randint

ant_stykker = 5
ant_rett = 0

for i in range(ant_stykker):
    tall1 = randint(2, 12)
    tall2 = randint(2, 12)

    print('Hva er ' + str(tall1) + ' ganger ' + str(tall
    svar = input()

    if int(svar) == tall1 * tall2:
        print('Ja, svaret er ' + svar)
        ant_rett = ant_rett + 1
    else:
        print('Nei, det riktige svaret er ' + str(tall1

print('Du fikk ' + str(ant_rett) + ' av ' + str(ant_styk
```

Steg 5: Rekursjon, hva er

Vi avslutter med å se på noe som heter rekursjon. Dette er en veldig s

Vi vil skrive et program som kan regne ut fakultetet av et tall. Fakultet
mindre enn seg. For eksempel er fakultetet av 4

```
fakultet(4) = 4 * 3 * 2 * 1 = 24
```

Trikset med rekursjon er at vi kan redusere oppgaven til noe som er v
at hvis vi hadde visst fakultetet av 3 kunne vi funnet fakultetet av 4 b

```
fakultet(4) = 4 * 3 * 2 * 1 = 4 * fakultet(3)
```

Videre kan vi finne fakultetet av 3 hvis vi vet fakultetet av 2 og så vid

```
fakultet(3) = 3 * 2 * 1 = 3 * fakultet(2)
fakultet(2) = 2 * 1 = 2 * fakultet(1)
fakultet(1) = 1
```

Sjekkliste

- [ ] La oss se om vi kan skrive et program som jobber på denne må

  ```
  def fakultet(tall):
      if tall == 1:
          return 1

  print(fakultet(1))
  ```

  Her bruker vi flere ting du har sett tidligere. Husk at `def` brukes
  `fakultet` som vi etterpå kaller inne i `print`-funksjonen.

- [ ] Foreløbig har vi bare sagt at vi vet at fakultetet av 1 er 1. Men r
  alle andre tall ved å legge på en enkelt kodelinje:

  ```
  def fakultet(tall):
      if tall == 1:
  ```

```
        return 1
    return tall * fakultet(tall-1)

print(fakultet(4))
```

Denne linjen sier at fakultetet til et tall er tallet selv ganget mec

Prøv å regne ut fakultetet av andre tall. Skjønner du hvordan de
du må kanskje venne deg til det?