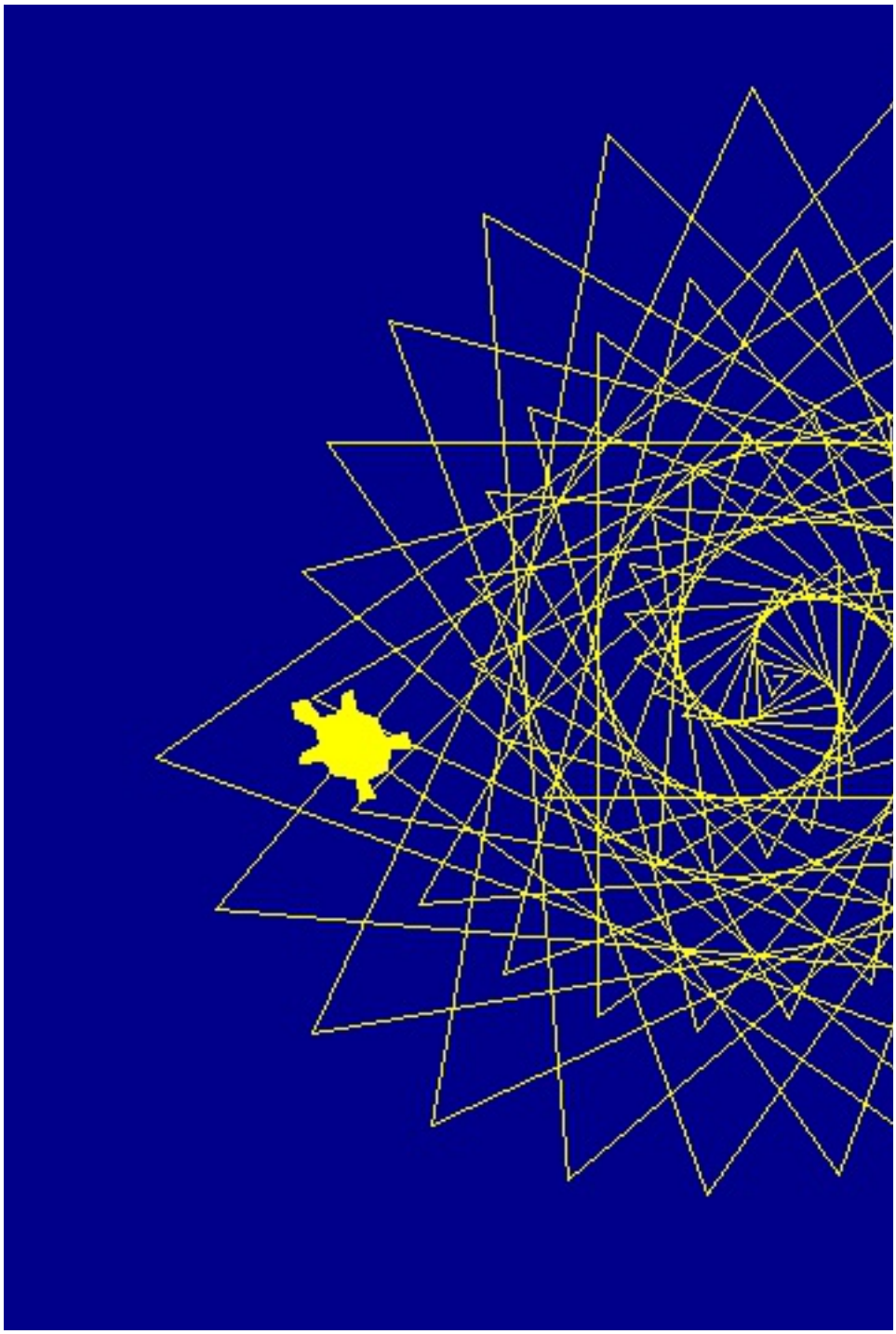
Skilpaddekunst

Introduksjon

Skilpadder (turtles på engelsk) er en form for roboter som har vært i b
skilpadde-biblioteket i Python til å utforske flere programmeringskons



Steg 1: Møt skilpadden

For å bruke skilpadder i Python må vi importere et bibliotek som heter
brukes mye for å organisere og gjenbruke kode som andre har skreve
Python. Vi vil her bruke den enkleste, hvor vi starter alle skilpaddepro

```
from turtle import *
```

Her betyr `*` alt, slik at linjen sier `Importer all kode fra turtle-bib`

Turtles

Navnet **Turtle** betyr *skilpadde* på norsk. Bakgrunnen til dette navn
William Grey Walter et par roboter som kunne bevege seg rundt. D
skallformet. De fikk derfor etterhvert kallenavnet skilpadder.

Senere ble måten disse skilpaddene beveget seg på (vi skal se hvo
programmeringsspråk, spesielt som en måte å tegne på. Språket *L*
skilpaddegrafikk, men nesten alle programmeringsspråk støtter de
Python.

Sjekkliste

- [ ] Det er på tide å lage vår første skilpadde. Start IDLE og åpne et
skrive inn følgende kode:

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')
```

Lagre programmet med navnet `skilpadde.py` og kjør det. Du s
blå bakgrunn. Hvis dette ikke skjer kan du se i det opprinnelige

- [ ] La oss se litt nøyere på hva programmet gjør så langt. Det er er
effekten av endringene og bedre forstå hvordan ting virker.

- [ ] Linjen `shape('turtle')` sier at vi vil bruke en skilpaddefigur. I s
`square`, `triangle` eller `classic`.

- [ ] Med `shapesize(2)` forteller vi programmet hvor stor vi vil at sk
- [ ] Kommandoene `bgcolor` og `color` bestemmer fargene på henh kjenner til veldig mange farger (men bare på engelsk), så prøv
- [ ] I de senere programmene vil vi bruke disse linjene på toppen. D du liker bedre i stedet.

Steg 2: En kunstnerisk skil

Skilpadden er ikke bare fin å se på. Den kan også tegne! I dette stege skilpadden om til en kunstner.

Sjekkliste

- [ ] Legg til en linje nederst i programmet ditt, slik at det ser slik ut:

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

forward(200)
```

- [ ] Når du kjører programmet vil du se at skilpadden har beveget s der den beveget seg.
- [ ] I tillegg til `forward` kan vi også bruke kommandoene `backward`

`right` for å svinge mot høyre. Prøv for eksempel å endre progr

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

forward(200)
left(60)
forward(50)
backward(200)
right(90)
forward(100)
```

Ser du at skilpadden utfører alle kommandoene du gir den?

Hvis vi setter sammen kommandoene litt systematisk kan vi teg
eksempel, om vi vil tegne en firkant kan vi først gå fremover, de
deretter fremover igjen, så svinge, så fremover, så svinge og til
som

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)
```

Tegner skilpadden en firkant når du kjører dette programmet?

- [ ] Hva med en trekant? Hvordan må du forandre koden din for at s
Prøv selv å endre koden og kjør den, ble resultatet som du trodd

Steg 3: Gjenta deg selv

Hvis du ser på koden vi har brukt for å tegne trekanter og firkanter ha
samme kode om og om igjen kan vi be Python gjenta deler av koden.

Sjekkliste

- [ ] Det følgende programmet tegner også en firkant, akkurat som d

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

for i in range(4):
    forward(100)
    right(90)
```

Endre koden din som over, og kjør programmet.

- [ ] Legg merke til at linjene som kommer etter `for` er skjøvet inn t
forteller hvor mye kode som skal gjentas i løkken. For å skyve k
IDLE. For å trekke koden tilbake til venstre kan du trykke `Shift`

- [ ] Prøv å trekk linjen `right(90)` til venstre, slik at for-løkken ser sl

```
for i in range(4):
    forward(100)
right(90)
```

Hva tror du programmet ditt vil gjøre nå? Forsøk å kjøre prograr

Skilpadden vil bare gå fremover fire ganger før den svinger til h
linje i stedet for en firkant.

- [ ] Nå som vi bruker en for-løkke har det også blitt mye enklere å e
endre 4 til 3 i for-løkken. I tillegg må vi endre vinkelen skilpadde
runde rundt trekanten må den snu totalt 360 grader. Siden den
i hvert hjørne. Programmet for å tegne en trekant blir dermed s

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

for i in range(3):
    forward(100)
    right(120)
```

- [ ] Prøv å endre programmet slik at det tegner andre mangekanter
åttekant eller kanskje en femtenkant.

Steg 4: Alle ting fortjener e

Vi skal fortsette med å gjøre koden vår enda mer fleksibel ved å gi tin
koden gjør.

Sjekkliste

- [ ] Vi innfører først variabler som sier hvor mange sider vi vil tegne
vi skal snu ved hvert hjørne. Endre programmet ditt slik at det s

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

sides = 4
length = 100
angle = 90

for i in range(sides):
    forward(length)
    right(angle)
```

Tegner programmet fortsatt en firkant?

- [ ] Nå kan du få programmet til å tegne en trekant bare ved å endr

- [ ] Vi kan gjøre programmet enda smartere. I stedet for at du selv
linjen `angle = 90` med

```
angle = 360 / sides
```

Nå kan du prøve å bare endre verdien av `sides` og kjøre om igj

Steg 5: Egne kommandoer

I Python kan vi også lage våre egne kommandoer ved å definere funk
oss selv på.

Sjekkliste

- [ ] Vi skal nå lage en funksjon som tegner en mangekant. Dette gjø
forkortelse for *define* som betyr definer). Endre programmet dit

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

def polygon(sides, length):
    angle = 360 / sides

    for i in range(sides):
        forward(length)
        right(angle)

polygon(4, 100)
```

Kjør programmet. Kjenner du igjen firkanten?

- [ ] Nå som vi har laget `polygon`-funksjonen er det kjempelett å teg
følgende linjene nederst i programmet ditt:

```
polygon(3, 100)
polygon(4, 100)
polygon(5, 100)
forward(125)
right(180)
polygon(3, 150)
polygon(5, 150)
```

```
polygon(7, 150)
```

Steg 6: Skilpaddekunst

Vi vil til slutt generalisere funksjonen vår litt slik at den ikke bare tegn

Sjekkliste

- [ ] Vi lager nå en ny funksjon `polylines` som ligner veldig mye på at de ikke alltid summerer seg til 360. Dette gjør underverker fo slik ut:

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

def polylines(sides, length, angle):
    for i in range(sides):
        forward(length)
        right(angle)

polylines(5, 100, 144)
```

Kjør programmet. Hva tegner skilpadden nå?

- [ ] En annen variant kan være hvor vi tegner en litt skjev mangeka firkanter hvor vinklene er 91 grader i stedet for 90 grader. Dette kommandoen med

```
polylines(91, 200, 91)
```

Vi kan også endre lengden av strekene etterhvert som vi tegner
mye funksjonen `spiral` ligner på `polylines`:

```
from turtle import *

shape('turtle')
shapesize(2)
bgcolor('darkblue')
color('yellow')

def polylines(sides, length, angle):
    for i in range(sides):
        forward(length)
        right(angle)

def spiral(sides, length, angle):
    for i in range(sides):
        forward(length)
        right(angle)
        length = length + 5

spiral(100, 5, 125)
```

Prøv forskjellige verdier i stedet for 100, 5 og 125 når du kaller
bilder synes du?

Prøv selv

Kombiner de forskjellige funksjonene vi har laget, `polygon`, `polyli`
kommandoene du har lært som for eksempel `forward` og `left`. Kl
Eller kanskje du kan tegne en by? Et hus kan for eksempel lages ve
toppen for taket.

Et tips helt på slutten er at funksjonene `penup()` og `pendown()` sty

Disse er veldig nyttige når man vil tegne flere figurer som ikke hen

Et annet tips er funksjonen `speed()`. Denne justerer hastigheten s
`speed(1)` tegne veldig sakte, mens `speed(11)` tegner kjempefort.