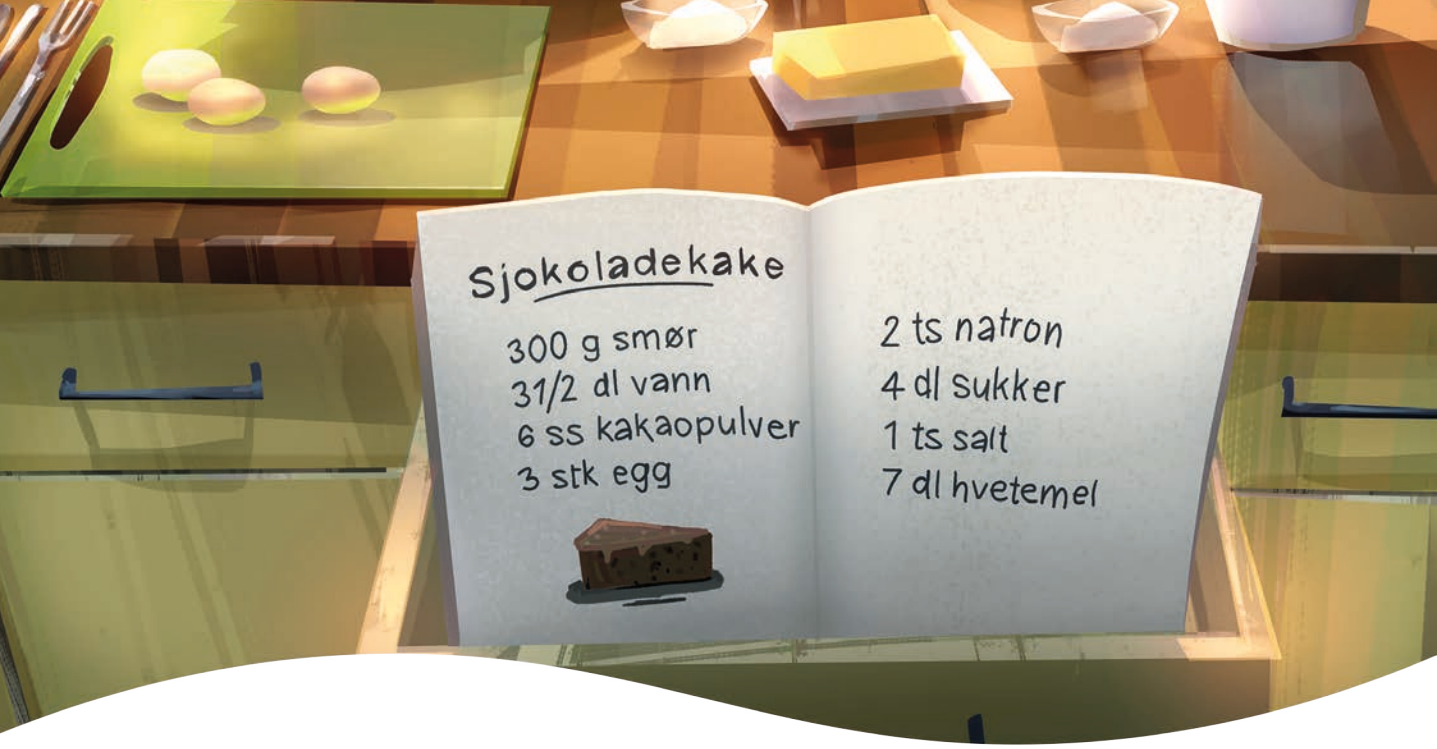
PROGRAMMERING 8–10
fra CAPPELEN DAMM

Espen Hjardar

Bokmål

CAPPELEN DAMM



Hva vil det si å følge en oppskrift?

Programmering

Koding og programmering betyr det samme. I programmering bruker vi en algoritme for å beskrive en fremgangsmåte som vi kan bruke for å løse en eller flere oppgaver. Innen matematikken kan det for eksempel være en multiplikasjonsalgoritme.

I programmering er en algoritme et sett med instruksjoner som forteller et program hva det skal gjøre, og enda viktigere, i hvilken rekkefølge instruksjonene skal gjennomføres.

Det finnes mange forskjellige programmeringsspråk, men algoritmene er ofte det samme på tvers av programmeringsspråk.

Begrep:

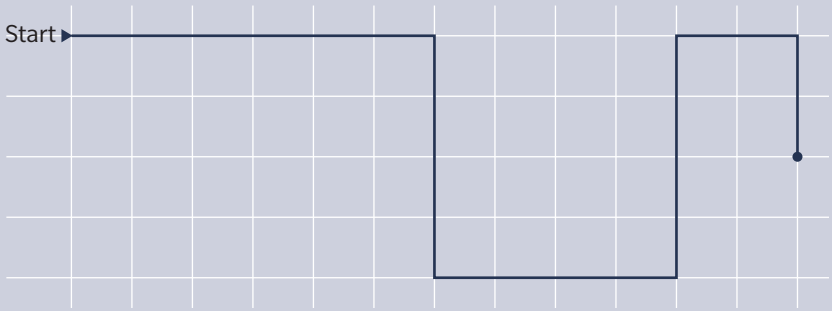
Pseudokode	En testkode hvor du gjør en analyse av det som skal skje, eller det som koden skal gjøre.
Løkke	En kode som gjør at en handling blir gjentatt flere ganger. Kalles en loop på engelsk.
Hvis-setning	En kode som har en betingelse. Her kan koden velge mellom ulike alternativer. Kalles en if-setning på engelsk.
Hvis-ellers-setning	En kode som har en eller flere betingelser, men som gjør noen annet hvis ikke noen av betingelsene er til stede. Kalles en If-else-setning på engelsk.
Inndata	Tall, variabler eller tekst som skrives inn. Kalles input på engelsk.
Utdata	Resultatet av en vurderings eller regneoperasjon. Kalles output på engelsk.

Lage og teste ulike algoritmer

Du skal nå øve deg på å lage og teste ut algoritmer. I arbeidet med å lage algoritmer bør du ofte lage en skisse eller en pseudokode over oppgaven som du vil at algoritmen skal løse eller gjøre før du lager selve algoritmen.

EKSEMPEL 1.1

Lag en algoritme som gjør at en person kan gå ruten som er vist på skissen.



Løsning

1. Start med å plassere deg på startfeltet i pilens retning
2. Gå 6 skritt fram
3. Snu 90 grader mot høyre
4. Gå 4 skritt fram
5. Snu 90 grader mot venstre
6. Gå 4 skritt fram
7. Snu 90 grader mot venstre
8. Gå 4 skritt fram
9. Snu 90 grader mot høyre
10. Gå 2 skritt fram
11. Snu 90 grader mot høyre
12. Gå 2 skritt fram
13. Stopp

OPPGAVER

1.1 Du skal nå lage en algoritme/kode gjør at du kan gå fra et startpunkt til et stoppunkt. Du velger selv hvor de to punktene skal være.

a) Lag en algoritme og test den på deg selv. La så en medelev teste algoritmen og se om eleven stopper på samme sted som deg.

b) Hvis medeleven ikke stoppet på samme sted som deg juster du algoritmen og prøver den på en annen medelev.

c) I hvilke situasjoner trenger du å legge inn en hvis-setning i algoritmen din?

d) Legg inn en hvis-setning i algoritmen din og test den på en medelev.

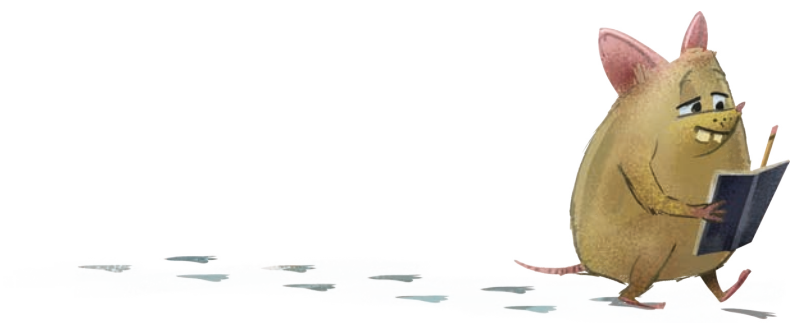
1.2 Du skal nå lage og teste en algoritme som inneholder en løkke. En løkke er en kode som gjentar en annen kode et ønsket antall ganger, eller uendelig antall ganger. Her kommer et forslag på hvordan en slik pseudokode kan se ut:

1. Start med å plassere deg på en åpen plass
2. Gjenta 4 ganger:
   - Gå 2 skritt rett fram
   - Snu 90 grader mot høyre
3. Stopp

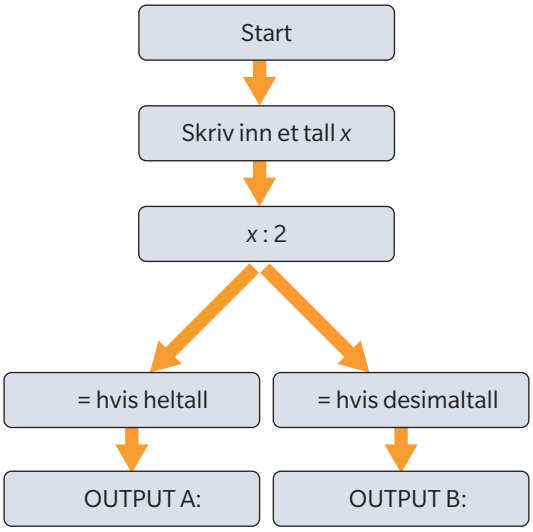
a) Lag en egen algoritme som inneholder en løkke, og test den på to eller flere medelever.

b) Hvis personene ikke stoppet på samme sted juster du algoritmen og prøver på nytt.

c) Legg inn en hvis-ellers-setning i algoritmen din og test den på en medelev.

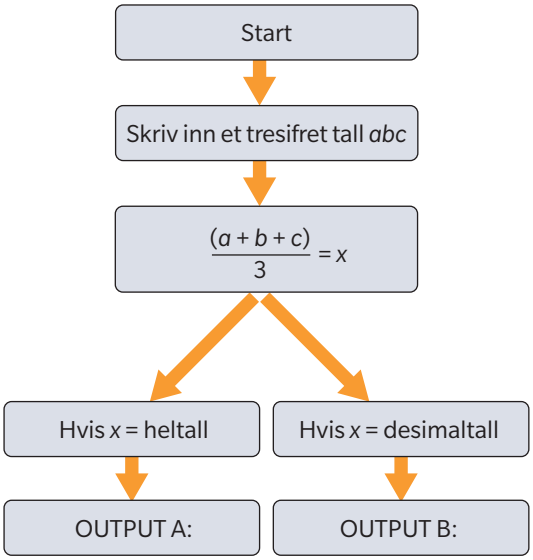


1.3 Se på algoritmen når du svarer på oppgaven.



a) Undersøk og forklar hva algoritmen gjør eller undersøker.
b) Gi et eksempel på hva OUTPUT A og hva OUTPUT B kan være.

1.4 Se på algoritmen når du svarer på oppgaven.



a) Undersøk og forklar hva algoritmen gjør eller undersøker.
b) Gi et eksempel på hva output A og hva output B kan være.

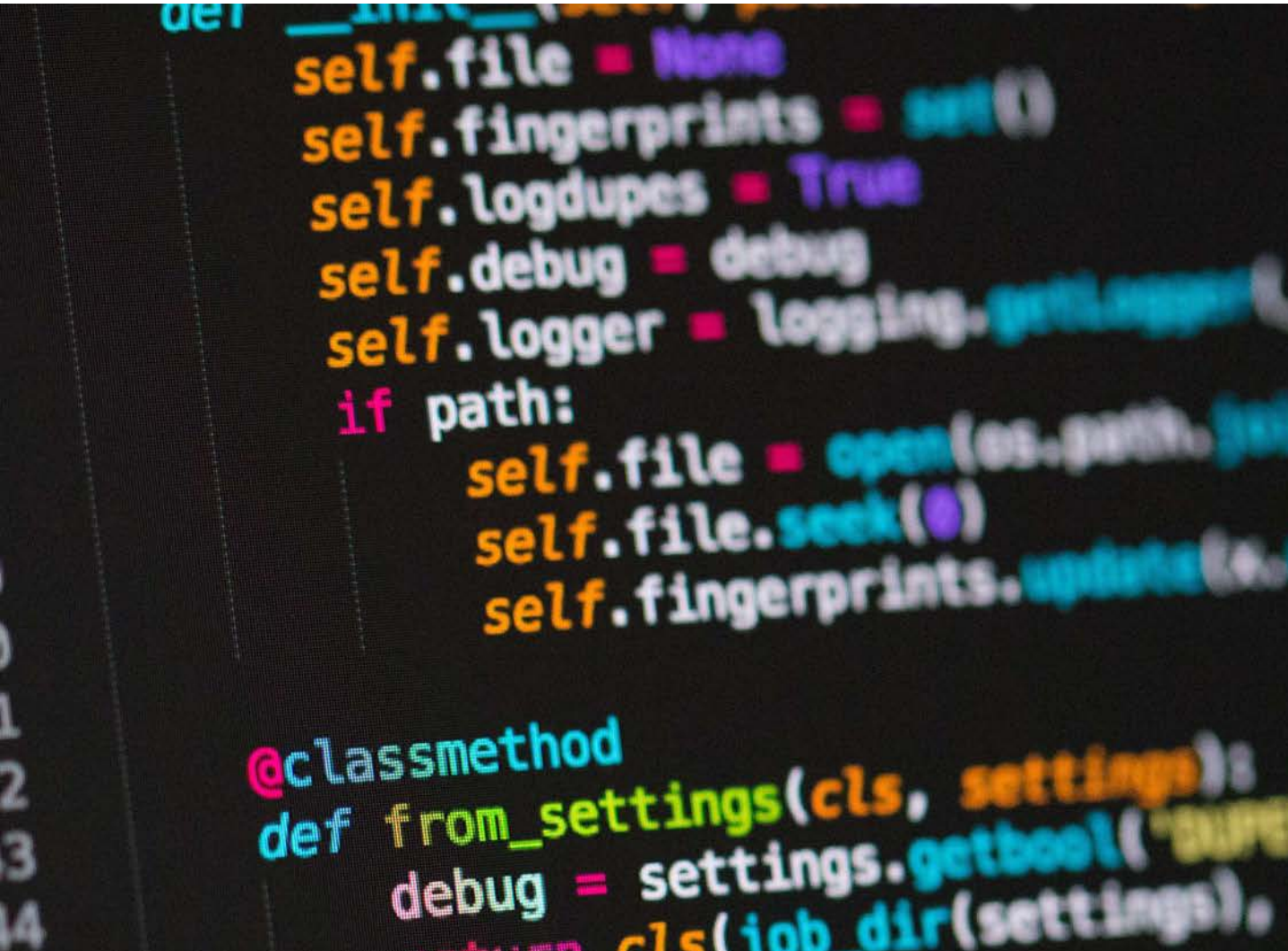
1.5 Lag en egen tekst-algoritme som

- undersøker om et tilfeldig heltall er delelig med 2.
- undersøker om et tilfeldig heltall er delelig med 2 eller 3.
- undersøker om et tilfeldig heltall er delelig med 5 eller 10.

Et oddetall er et tall som har rest 1 når du deler tallet på 2. Det betyr at et oddetall kan skrives som et partall + 1, siden et partall alltid kan deles på 2. Vi kan da skrive et oddetall slik: $2p + 1$

Bruk uttrykket for ovenfor til å bevise at summen av to oddetall alltid blir et partall.

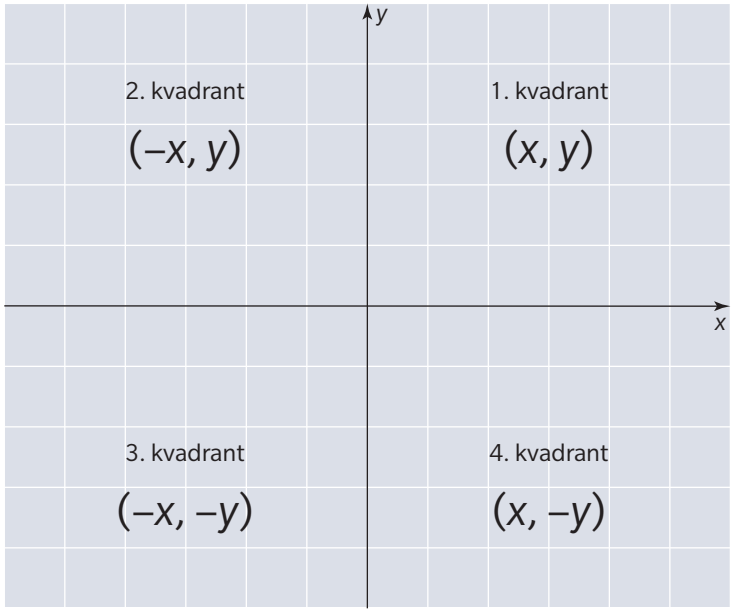
Bruk enten en algoritme eller algebra når du utfører beviset.



Lage en tegne-algoritme

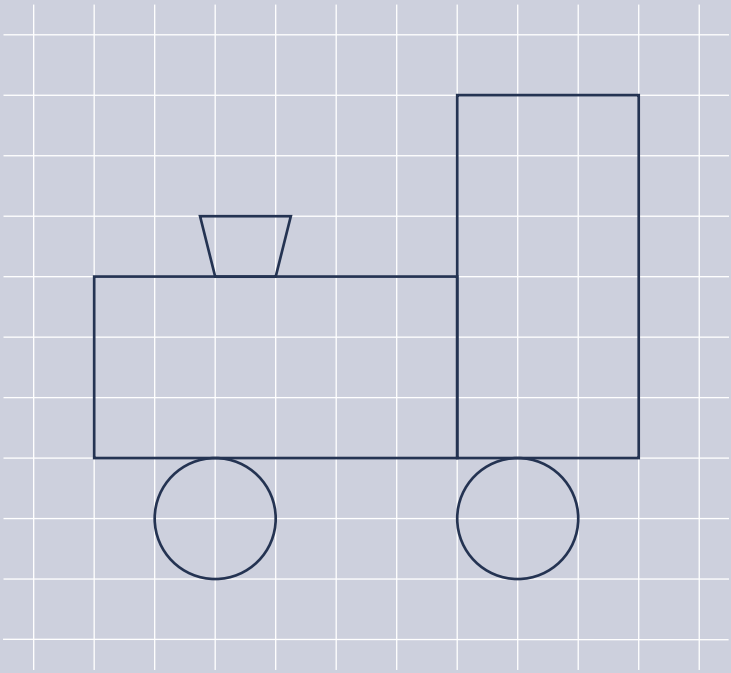
I programmering lager vi i noen ganger algoritmer som styrer en penn. Da programmerer vi pennen til å bevege seg i ulike lengderetninger eller langs akser i et koordinatsystem.

Felles for mange programmeringsspråk er at de bruker et koordinatsystem for å orientere seg på skjermen. Sentrum har da koordinatene (0, 0) og koordinatsystemet deles inn i fire kvadranter hvor koordinatene har ulike fortegn.



EKSEMPEL 1.2

Lage en tegne-algoritme som gjør at en medelev kan tegne figuren som er vist her.



Løsning

Tegn etter algoritmen under, du trenger et ruteark, blyant og linjal når du tegner algoritmen.

1. Tegn et rektangel hvor lengden er 6 cm og høyden er 3 cm midt på arket.
2. På høyre side av rektanglet tegner du et nytt rektangel med lengde 3 cm og høyde 6 cm.
3. Tegn så to sirkler på undersiden av de to rektanglene, 2 cm inn fra hver side, med en diameter på 2 cm. La sentrum i sirklene ligge 1 cm under rektanglene.
4. På toppen av det første rektanglet tegner du et trapes. Start 2 cm inn fra venstre og la grunnlinjen i trapeset være 1 cm, høyden og topplinjen være 1,5 cm.

OPPGAVER

1.6 Lag en tegning satt sammen av geometriske figurer som kvadrat, rektangel, sirkel, likesidet trekant og halvsirkel.

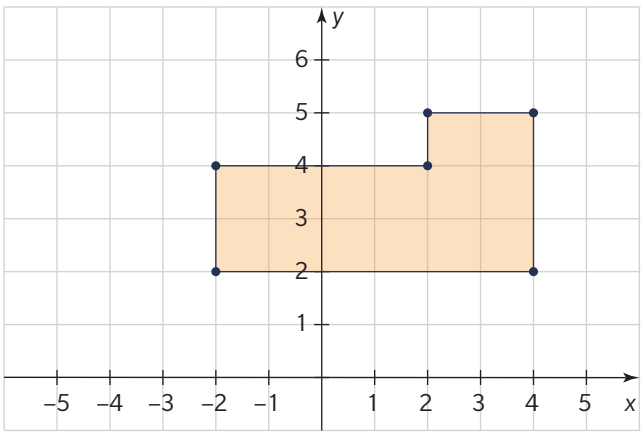
a) Lag en tekst algoritme som passer til tegningen din.

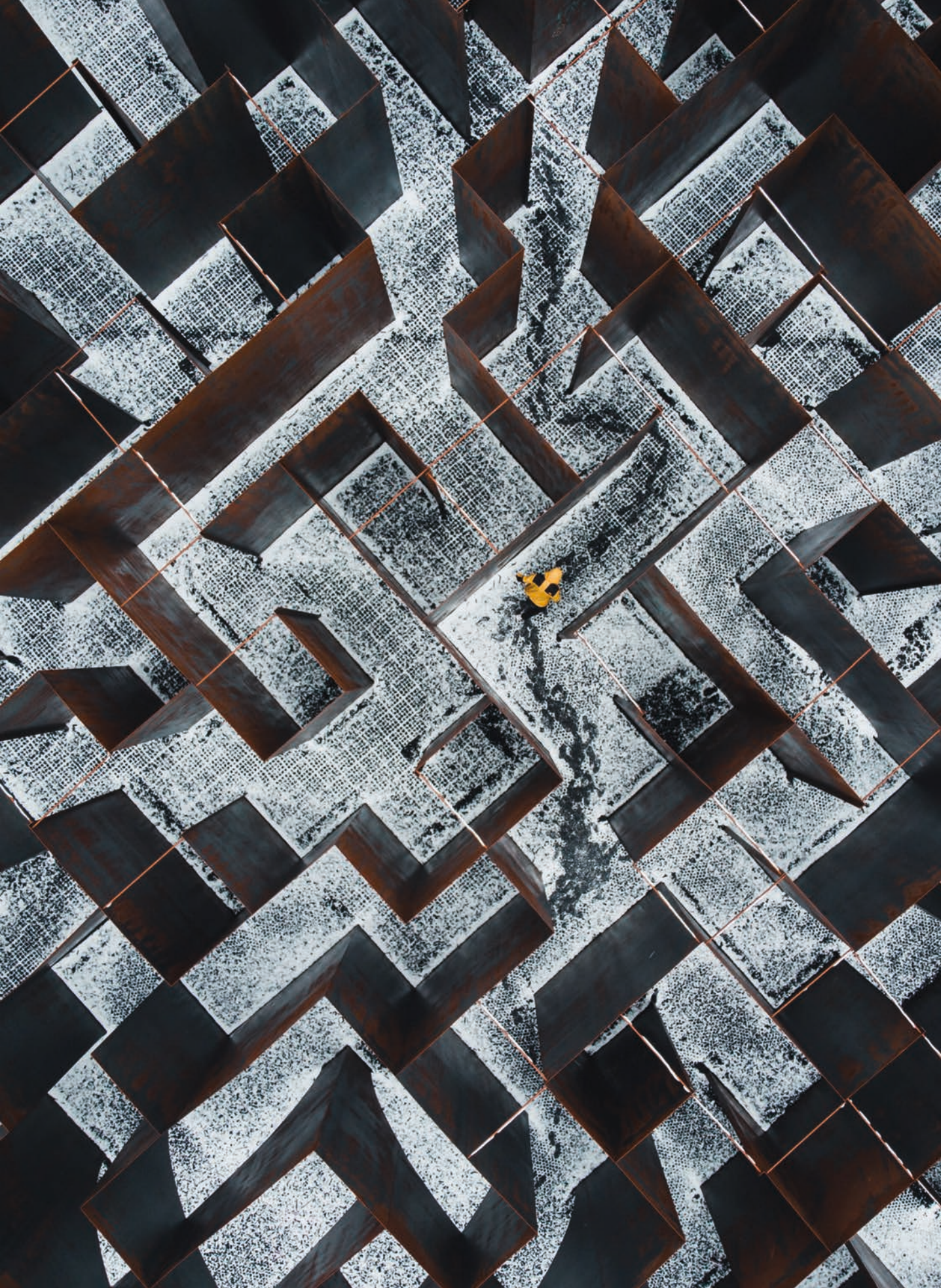
b) Test algoritmen din på en eller flere medelever, blir tegningen deres lik som din?

c) Undersøk hva som fungerte og ikke fungerte. Juster koden og prøv på nytt. Kom du nærmere ønsket resultat denne gangen?

? La en medelev lage en algoritme på grunnlag av din tegning i oppgaven ovenfor.
Sammenlikn de to algoritmene og finn ut hva som var ulikt og hva som var likt.

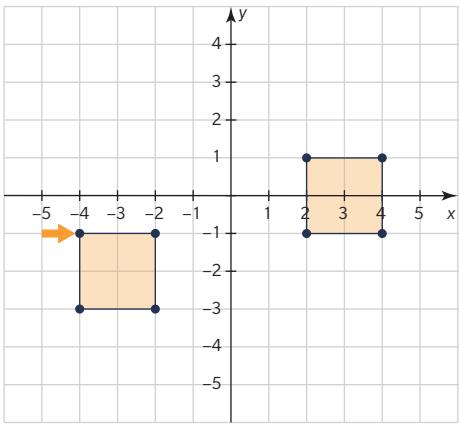
1.7 Lag en algoritme som styrer en penn som tegner figuren under. Du angir selv startpunkt, startretning og stoppunkt.

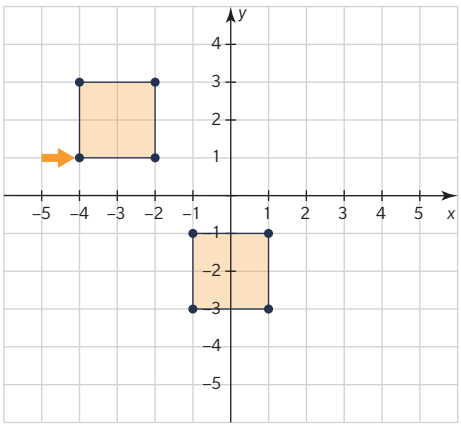




1.8 Algoritmen styrer en penn i et koordinatsystem. Pennen starter i pilens retning.

1. Start i (–4, –1)
2. Gjenta 4 ganger
   - Gå 2 frem
   - Snu 90 grader til høyre
3. Flytt til 6 frem
4. Gjenta 4 ganger
   - Gå 2 frem
   - Snu 90 grader til venstre
5. Stopp





a) Hvilket av koordinatsystemene passer til algoritmen ovenfor?

b) Skriv algoritmen som passer til tegningene i det andre koordinatsystemet.