

MAT1030 – Diskret Matematikk

Plenumsregning 1: Kapittel 1

Mathias Barra

Matematisk institutt, Universitetet i Oslo

16. januar 2009

(Sist oppdatert: 2009-02-02 14:22)



Plenumsregning 1

Velkommen til plenumsregning for MAT1030

Velkommen til plenumsregning for MAT1030

- Fredager 12:15–14:00

Velkommen til plenumsregning for MAT1030

- Fredager 12:15–14:00
- Vi vil gjennomgå utvalgte ukeoppgaver,

Velkommen til plenumsregning for MAT1030

- Fredager 12:15–14:00
- Vi vil gjennomgå utvalgte ukeoppgaver, se på eksempler fra boka og

Velkommen til plenumsregning for MAT1030

- Fredager 12:15–14:00
- Vi vil gjennomgå utvalgte ukeoppgaver, se på eksempler fra boka og repetere stoff fra forelesningene.

Velkommen til plenumsregning for MAT1030

- Fredager 12:15–14:00
- Vi vil gjennomgå utvalgte ukeoppgaver, se på eksempler fra boka og repetere stoff fra forelesningene.
- Spørsmål og svar

Velkommen til plenumsregning for MAT1030

- Fredager 12:15–14:00
- Vi vil gjennomgå utvalgte ukeoppgaver, se på eksempler fra boka og repetere stoff fra forelesningene.
- Spørsmål og svar
- Viktig for at DU skal ha fullt utbytte: **Forsøk på oppgavene selv først!**

Repetisjon: Algoritmer og pseudokode

Repetisjon: Algoritmer og pseudokode

- En **algoritme** er en oppskrift som forteller oss hvordan vi skritt for skritt skal kunne oppnå et resultat eller løse et problem.

Repetisjon: Algoritmer og pseudokode

- En **algoritme** er en oppskrift som forteller oss hvordan vi skritt for skritt skal kunne oppnå et resultat eller løse et problem.
- Det skal ikke kreves intelligens eller forståelse for å følge den. (Hmm. . . dette blir litt filosofisk!)

Repetisjon: Algoritmer og pseudokode

- En **algoritme** er en oppskrift som forteller oss hvordan vi skritt for skritt skal kunne oppnå et resultat eller løse et problem.
- Det skal ikke kreves intelligens eller forståelse for å følge den. (Hmm. . . dette blir litt filosofisk!)
- En **pseudokode** er en måte å beskrive en algoritme på.

Repetisjon: Algoritmer og pseudokode

- En **algoritme** er en oppskrift som forteller oss hvordan vi skritt for skritt skal kunne oppnå et resultat eller løse et problem.
- Det skal ikke kreves intelligens eller forståelse for å følge den. (Hmm. . . dette blir litt filosofisk!)
- En **pseudokode** er en måte å beskrive en algoritme på.
- Hvert steg i algoritmen skal være beskrevet på en **entydig** måte.

Eksempel fra boka

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Løsning

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Løsning

1. Input r [r er radiusen til sirkelen.]

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Løsning

1. Input r [r er radiusen til sirkelen.]
2. $\text{areal} \leftarrow \pi r^2$

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Løsning

1. Input r [r er radiusen til sirkelen.]
2. $\text{areal} \leftarrow \pi r^2$
3. Output areal

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Løsning

1. Input r [r er radiusen til sirkelen.]
2. $\text{areal} \leftarrow \pi r^2$
3. Output areal

I dette eksempelet ser vi litt av [syntaksen](#) vi benytter:

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Løsning

1. Input r [r er radiusen til sirkelen.]
2. $\text{areal} \leftarrow \pi r^2$
3. Output areal

I dette eksempelet ser vi litt av [syntaksen](#) vi benytter:

- Hvert steg i algoritmen er nummerert.

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Løsning

1. Input r [r er radiusen til sirkelen.]
2. $\text{areal} \leftarrow \pi r^2$
3. Output areal

I dette eksempelet ser vi litt av [syntaksen](#) vi benytter:

- Hvert steg i algoritmen er nummerert.
- Kommentarer skrives mellom [] (i boken benytter de {}).

Eksempel fra boka

Eksempel (1.1.1)

Skriv en algoritme som regner ut arealet av en sirkel, gitt radiusen.

Løsning

1. Input r [r er radiusen til sirkelen.]
2. $\text{areal} \leftarrow \pi r^2$
3. Output areal

I dette eksempelet ser vi litt av [syntaksen](#) vi benytter:

- Hvert steg i algoritmen er nummerert.
- Kommentarer skrives mellom [] (i boken benytter de {}).
- Symbolet \leftarrow betegner *tilordning*.

Repetisjon: Kontrollstrukturer

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
-	-	-

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$ ◀
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$

Kjøring

Steg	sum	i
1	-	-

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$ ◀
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$

Kjøring

Steg	sum	i
1	0	-
	▲	

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	0	-

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	0	1
		▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	0	1

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring


Steg	sum	i
2.1	1	1

▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** 
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	1	1

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	1	2
		▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$ ◀

Kjøring

Steg	sum	i
2.1	1	2

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	3	2

▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	3	2

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	3	3
		▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	3	3

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	6	3

▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	6	3

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	6	4
		▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	6	4

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	10	4
	▲	

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	10	4

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	10	5
		▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	10	5

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	15	5
	▲	

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	15	5

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For $i = 1$ to 6 do** ◀
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
2	15	6
		▲

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	15	6

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $sum \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $sum \leftarrow sum + i$ ◀

Kjøring

Steg	sum	i
2.1	21	6
	▲	

Repetisjon: Kontrollstrukturer

- En **kontrollstruktur** brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

For-do

1. $\text{sum} \leftarrow 0$
2. **For** $i = 1$ **to** 6 **do**
 - 2.1. $\text{sum} \leftarrow \text{sum} + i$

Kjøring

Steg	sum	i
-	21	-

Repetisjon: Kontrollstrukturer

Repetisjon: Kontrollstrukturer

If-then



Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**

1.1. $x \leftarrow -x$

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**

else

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$
- else**

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

Repetisjon: Kontrollstrukturer

If-then

1. **If $x < 0$ then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If $x \geq 0$ then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

While-do

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

While-do

1. **While** $svar \neq 1$ **do**

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

While-do

1. **While** $svar \neq 1$ **do**
 - 1.1. Input $svar$

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

While-do

1. **While** $svar \neq 1$ **do**
 - 1.1. Input $svar$

Repeat-until

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

While-do

1. **While** $svar \neq 1$ **do**
 - 1.1. Input $svar$

Repeat-until

1. $i \leftarrow 0$

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

While-do

1. **While** $svar \neq 1$ **do**
 - 1.1. Input $svar$

Repeat-until

1. $i \leftarrow 0$
2. **Repeat**

until $x_i = 0$

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

While-do

1. **While** $svar \neq 1$ **do**
 - 1.1. Input $svar$

Repeat-until

1. $i \leftarrow 0$
2. **Repeat**
 - 2.1. $i \leftarrow i + 1$
until $x_i = 0$

Repetisjon: Kontrollstrukturer

If-then

1. **If** $x < 0$ **then**
 - 1.1. $x \leftarrow -x$

If-then-else

1. **If** $x \geq 0$ **then**
 - 1.1. $y \leftarrow \sqrt{x}$**else**
 - 1.2. Output ' \sqrt{x} fins ikke' [er kompleks]

While-do

1. **While** $svar \neq 1$ **do**
 - 1.1. Input $svar$

Repeat-until

1. $i \leftarrow 0$
2. **Repeat**
 - 2.1. $i \leftarrow i + 1$
 - 2.2. Input x_i**until** $x_i = 0$

Eksempler fra boka

Eksempler fra boka

Eksempel (1.2.1/1.2.2)

Finn det minste tallet i en liste av tall.

Eksempler fra boka

Løsning

Eksempler fra boka

Løsning

1. Input the number of values n

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\min \leftarrow x_1$

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\min \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\min \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \min$ **then**

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\min \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \min$ **then**
 - 4.1.1. $\min \leftarrow x_i$

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
-	-	-	-	-	-	-

Eksempler fra boka

Løsning

1. **Input the number of values n** ◀
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For $i = 2$ to n do**
 - 4.1. **If $x_i < \text{min}$ then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
1	-	-	-	-	-	-

Eksempler fra boka

Løsning

1. **Input the number of values n** ◀
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For $i = 2$ to n do**
 - 4.1. **If $x_i < \text{min}$ then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
1	-	-	3	-	-	-

▲

Eksempler fra boka

Løsning

1. Input the number of values n
2. **Input the list of numbers x_1, \dots, x_n** ◀
3. $\text{min} \leftarrow x_1$
4. **For $i = 2$ to n do**
 - 4.1. **If $x_i < \text{min}$ then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
2	-	-	3	-	-	-

Eksempler fra boka

Løsning

1. Input the number of values n
2. **Input the list of numbers x_1, \dots, x_n** ◀
3. $\text{min} \leftarrow x_1$
4. **For $i = 2$ to n do**
 - 4.1. **If $x_i < \text{min}$ then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
2	-	-	3	5	4	8
				▲	▲	▲

Eksempler fra boka

Løsning


1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$ ◀
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
3	-	-	3	5	4	8

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$ 
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
3	5	-	3	5	4	8
	▲					

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For $i = 2$ to n do** ◀
 - 4.1. **If $x_i < \text{min}$ then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
4	5	-	3	5	4	8

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For $i = 2$ to n do** ◀
 - 4.1. **If $x_i < \text{min}$ then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min


Kjøring

Steg	min	i	n	x_1	x_2	x_3
4	5	2	3	5	4	8

▲

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then** 
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
4.1	5	2	3	5	4	8

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$ ◀
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
4.1.1	5	2	3	5	4	8

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$ ◀
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
4.1.1	4	2	3	5	4	8

▲

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For $i = 2$ to n do** ◀
 - 4.1. **If $x_i < \text{min}$ then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
4	4	2	3	5	4	8

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For $i = 2$ to n do** ◀
 - 4.1. **If $x_i < \text{min}$ then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min


Kjøring

Steg	min	i	n	x_1	x_2	x_3
4	4	3	3	5	4	8

▲

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then** 
 - 4.1.1. $\text{min} \leftarrow x_i$
5. Output min

Kjøring

Steg	min	i	n	x_1	x_2	x_3
4.1	4	3	3	5	4	8

Eksempler fra boka

Løsning

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$
5. **Output** min ◀

Kjøring

Steg	min	i	n	x_1	x_2	x_3
5	4	-	3	5	4	8

Reserverte ord og uttrykk i pseudokode

Reserverte ord og uttrykk i pseudokode

- **If**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**
- **or**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**
- **or**
- **not**

Reserverte ord og uttrykk i pseudokode

- **If**
 - **then**
 - **else**
 - **For**
 - **to**
 - **do**
 - **While**
 - **Repeat**
 - **until**
 - **and**
 - **or**
 - **not**
- **Input**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**
- **or**
- **not**
- **Input**
- **Output**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**
- **or**
- **not**
- **Input**
- **Output**
- **←**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**
- **or**
- **not**
- **Input**
- **Output**
- **←**
- **true**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**
- **or**
- **not**
- **Input**
- **Output**
- **←**
- **true**
- **false**

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**
- **or**
- **not**
- **Input**
- **Output**
- \leftarrow
- **true**
- **false**
- **[]** (eller **{}** i boka)

Reserverte ord og uttrykk i pseudokode

- **If**
- **then**
- **else**
- **For**
- **to**
- **do**
- **While**
- **Repeat**
- **until**
- **and**
- **or**
- **not**
- **Input**
- **Output**
- \leftarrow
- **true**
- **false**
- **[]** (eller **{}** i boka)
- vanlig matematiske uttrykk (f.eks. $x < 0$ og \sqrt{x})

Eksempler fra boka

Eksempler fra boka

Eksempel (1.2.3)

Lag en algoritme som sjekker om en streng $c_1c_2 \dots c_n$ av n tegn består av kun sifre, eller om ikke-sifre forekommer, og skriv ut en passende melding.

Eksempler fra boka

Eksempel (1.2.3)

Lag en algoritme som sjekker om en streng $c_1c_2 \dots c_n$ av n tegn består av kun sifre, eller om ikke-sifre forekommer, og skriv ut en passende melding.

- Hvis input er 12345 skal algoritmen returnere meldingen:
Strengen inneholder kun sifre.

Eksempler fra boka

Eksempel (1.2.3)

Lag en algoritme som sjekker om en streng $c_1c_2 \dots c_n$ av n tegn består av kun sifre, eller om ikke-sifre forekommer, og skriv ut en passende melding.

- Hvis input er 12345 skal algoritmen returnere meldingen:
Strengen inneholder kun sifre.
- Hvis input er 123@45 skal algoritmen returnere meldingen:
Strengen inneholder tegn som ikke er sifre.

Eksempler fra boka

Utkast 1

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false

2. **Repeat**

until *ikkesiffer_oppdaget* = true

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**

until *ikkesiffer_oppdaget* = true

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; $ikkesiffer_oppdaget \leftarrow \text{false}$
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. $ikkesiffer_oppdaget \leftarrow \text{true}$
- until** $ikkesiffer_oppdaget = \text{true}$

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 2.2. $i \leftarrow i + 1$
- until** *ikkesiffer_oppdaget* = true

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 2.2. $i \leftarrow i + 1$
- until** *ikkesiffer_oppdaget* = true

- Identifikatorer/variable skrives i *kursiv* uten mellomrom.

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 2.2. $i \leftarrow i + 1$
- until** *ikkesiffer_oppdaget* = true

- Identifikatorer/variable skrives i *kursiv* uten mellomrom.
- *ikkesiffer_oppdaget* kalles gjerne for en *logisk* (eller Boolsk) variabel, siden den kun vil ta verdiene true eller false.

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 2.2. $i \leftarrow i + 1$
- until** *ikkesiffer_oppdaget* = true

- Identifikatorer/variable skrives i *kursiv* uten mellomrom.
- *ikkesiffer_oppdaget* kalles gjerne for en *logisk* (eller Boolsk) variabel, siden den kun vil ta verdiene true eller false.
- Kunne ha skrevet '**until** *ikkesiffer_oppdaget*' i siste linje.

Eksempler fra boka

Utkast 1

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 2.2. $i \leftarrow i + 1$
- until** *ikkesiffer_oppdaget* = true

- Identifikatorer/variable skrives i *kursiv* uten mellomrom.
- *ikkesiffer_oppdaget* kalles gjerne for en *logisk* (eller Boolsk) variabel, siden den kun vil ta verdiene true eller false.
- Kunne ha skrevet '**until** *ikkesiffer_oppdaget*' i siste linje.
- Denne vil ikke **terminere** hvis strengen består av kun sifre.

Eksempler fra boka

Utkast 2

Eksempler fra boka

Utkast 2

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
2. **Repeat**

until *ikkesiffer_oppdaget* = true **or** $i = n + 1$

Eksempler fra boka

Utkast 2

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 2.2. $i \leftarrow i + 1$
- until** *ikkesiffer_oppdaget* = true **or** $i = n + 1$

Utkast 2

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 2.2. $i \leftarrow i + 1$
- until** *ikkesiffer_oppdaget* = true **or** $i = n + 1$
- Hvis alle tegn er sifre vil algoritmen terminere når verdien til i er $n + 1$.

Utkast 2

1. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
 2. **Repeat**
 - 2.1. **If** c_i ikke er et siffer **then**
 - 2.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 2.2. $i \leftarrow i + 1$
- until** *ikkesiffer_oppdaget* = true **or** $i = n + 1$

- Hvis alle tegn er sifre vil algoritmen terminere når verdien til i er $n + 1$.
- Nå gjenstår Input og Output.

Eksempler fra boka

Utkast 3

Eksempler fra boka

Utkast 3

1. Input n
2. Input $c_1 c_2 \dots c_n$

Eksempler fra boka

Utkast 3

1. Input n
2. Input $c_1 c_2 \dots c_n$

5. **If** *ikkessiffer_oppdaget* = true **then**
 - 5.1. Output 'Strengen inneholder tegn som ikke er sifre.'**else**
 - 5.2. Output 'Strengen inneholder kun sifre.'

Eksempler fra boka

Utkast 3

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 1$; *ikkessiffer_oppdaget* \leftarrow false
4. **Repeat**
 - 4.1. **If** c_i ikke er et siffer **then**
 - 4.1.1. *ikkessiffer_oppdaget* \leftarrow true
 - 4.2. $i \leftarrow i + 1$**until** *ikkessiffer_oppdaget* = true **or** $i = n + 1$
5. **If** *ikkessiffer_oppdaget* = true **then**
 - 5.1. Output 'Strengen inneholder tegn som ikke er sifre.'**else**
 - 5.2. Output 'Strengen inneholder kun sifre.'

Eksempler fra boka

Utkast 3

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 1$; *ikkesiffer_oppdaget* \leftarrow false
4. **Repeat**
 - 4.1. **If** c_i ikke er et siffer **then**
 - 4.1.1. *ikkesiffer_oppdaget* \leftarrow true
 - 4.2. $i \leftarrow i + 1$**until** *ikkesiffer_oppdaget* = true **or** $i = n + 1$
5. **If** *ikkesiffer_oppdaget* = true **then**
 - 5.1. Output 'Strengen inneholder tegn som ikke er sifre.'**else**
 - 5.2. Output 'Strengen inneholder kun sifre.'

- Men, hva hvis $n = 0$? Da vil steg 4.1 ikke kunne utføres.

Kommentarer

- **Repeat-until**-løkken blir alltid utført minst én gang.

Kommentarer

- **Repeat-until**-løkken blir alltid utført minst én gang.
- Et alternativ er å bruke en **While**-løkke.

Kommentarer

- **Repeat-until**-løkken blir alltid utført minst én gang.
- Et alternativ er å bruke en **While**-løkke.
- Da utføres testen i begynnelsen i stedet for på slutten, som med **Repeat-until**.

Eksempler fra boka

Eksempler fra boka

Løsning

Eksempler fra boka

Løsning

1. Input n

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 0$; *ikkesiffer_oppdaget* \leftarrow false

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 0$; *ikkesiffer_oppdaget* \leftarrow false
4. **While** *ikkesiffer_oppdaget* = false **and** $i < n$ **do**

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 0$; *ikkesiffer_oppdaget* \leftarrow false
4. **While** *ikkesiffer_oppdaget* = false **and** $i < n$ **do**
 - 4.1. $i \leftarrow i + 1$

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 0$; *ikkesiffer_oppdaget* \leftarrow false
4. **While** *ikkesiffer_oppdaget* = false **and** $i < n$ **do**
 - 4.1. $i \leftarrow i + 1$
 - 4.2. **If** c_i ikke er et siffer **then**

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 0$; *ikkesiffer_oppdaget* \leftarrow false
4. **While** *ikkesiffer_oppdaget* = false **and** $i < n$ **do**
 - 4.1. $i \leftarrow i + 1$
 - 4.2. **If** c_i ikke er et siffer **then**
 - 4.2.1. *ikkesiffer_oppdaget* \leftarrow true

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 0$; *ikkessiffer_oppdaget* \leftarrow false
4. **While** *ikkessiffer_oppdaget* = false **and** $i < n$ **do**
 - 4.1. $i \leftarrow i + 1$
 - 4.2. **If** c_i ikke er et siffer **then**
 - 4.2.1. *ikkessiffer_oppdaget* \leftarrow true
5. **If** *ikkessiffer_oppdaget* = true **then**

else

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 0$; *ikkessiffer_oppdaget* \leftarrow false
4. **While** *ikkessiffer_oppdaget* = false **and** $i < n$ **do**
 - 4.1. $i \leftarrow i + 1$
 - 4.2. **If** c_i ikke er et siffer **then**
 - 4.2.1. *ikkessiffer_oppdaget* \leftarrow true
5. **If** *ikkessiffer_oppdaget* = true **then**
 - 5.1. Output 'Strengen inneholder tegn som ikke er sifre.'**else**

Eksempler fra boka

Løsning

1. Input n
2. Input $c_1 c_2 \dots c_n$
3. $i \leftarrow 0$; *ikkesiffer_oppdaget* \leftarrow false
4. **While** *ikkesiffer_oppdaget* = false **and** $i < n$ **do**
 - 4.1. $i \leftarrow i + 1$
 - 4.2. **If** c_i ikke er et siffer **then**
 - 4.2.1. *ikkesiffer_oppdaget* \leftarrow true
5. **If** *ikkesiffer_oppdaget* = true **then**
 - 5.1. Output 'Strengen inneholder tegn som ikke er sifre.'**else**
 - 5.2. Output 'Strengen inneholder kun sifre.'

Eksempler fra boka

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reellt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reellt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reellt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reellt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reellt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. svar \leftarrow svar $\times x$

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. svar \leftarrow svar $\times x$
4. Output svar

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. $\text{svar} \leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. $\text{svar} \leftarrow \text{svar} \times x$
4. Output svar

Kjøring


x	n	i	svar
<hr/>			

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. **Input** x, n 
2. $\text{svar} \leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. $\text{svar} \leftarrow \text{svar} \times x$
4. **Output** svar

Kjøring


x	n	i	svar
<hr/>			

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. **Input** x, n 
2. svar $\leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. svar \leftarrow svar $\times x$
4. **Output** svar

Kjøring


x	n	i	svar
2	3		
▲	▲		

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$ 
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. svar \leftarrow svar $\times x$
4. Output svar

Kjøring


x	n	i	svar
2	3		

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$ 
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. svar \leftarrow svar $\times x$
4. Output svar

Kjøring


x	n	i	svar
2	3		2
			▲

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$
3. **For $i = 1$ to $n - 1$ do** 
 - 3.1. svar \leftarrow svar $\times x$
4. Output svar

Kjøring

x	n	i	svar
2	3		2

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do** ◀
 - 3.1. svar \leftarrow svar $\times x$
4. Output svar

Kjøring

x	n	i	svar
2	3	1	2
		▲	

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. $\text{svar} \leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. $\text{svar} \leftarrow \text{svar} \times x$ ◀
4. Output svar

Kjøring

x	n	i	svar
2	3	1	2

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. $\text{svar} \leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. $\text{svar} \leftarrow \text{svar} \times x$ ◀
4. Output svar

Kjøring


x	n	i	svar
2	3	1	4
			▲

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do** 
 - 3.1. svar \leftarrow svar $\times x$
4. Output svar

Kjøring

x	n	i	svar
2	3	1	4

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. svar $\leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do** ◀
 - 3.1. svar \leftarrow svar $\times x$
4. Output svar

Kjøring

x	n	i	svar
2	3	2	4
		▲	

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. $\text{svar} \leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. $\text{svar} \leftarrow \text{svar} \times x$ ◀
4. Output svar

Kjøring

x	n	i	svar
2	3	2	4

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. $\text{svar} \leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. $\text{svar} \leftarrow \text{svar} \times x$ ◀
4. Output svar

Kjøring

x	n	i	svar
2	3	2	8
			▲

Eksempler fra boka

Eksempel (1.3.1)

Lag en algoritme som regner ut x^n , hvor x er et reelt tall og n er et positivt heltall. (Vi antar at vi har multiplikasjon, men ikke eksponensiering.)

Løsning

1. Input x, n
2. $\text{svar} \leftarrow x$
3. **For** $i = 1$ **to** $n - 1$ **do**
 - 3.1. $\text{svar} \leftarrow \text{svar} \times x$
4. **Output svar** ◀

Kjøring

x	n	i	svar
2	3	-	8

Eksempler fra boka

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Kjøring

$$\begin{array}{cc} x & y \\ \hline 2 & 3 \end{array}$$

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Kjøring

$$\begin{array}{cc|c} x & y & \\ \hline 2 & 3 & \rightsquigarrow \\ \hline x & y & \\ \hline 3 & 3 & \end{array}$$

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

Kjøring

$$\begin{array}{cc|c} x & y & \\ \hline 2 & 3 & \rightsquigarrow \\ \hline x & y & \\ \hline 3 & 3 & \end{array}$$

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$

Kjøring

$$\begin{array}{cc|c} x & y & \\ \hline 2 & 3 & \rightsquigarrow \\ \hline x & y & \\ \hline 3 & 3 & \end{array}$$

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$
2. $x \leftarrow y$

Kjøring

$$\begin{array}{cc|c} x & y & \\ \hline 2 & 3 & \rightsquigarrow \\ \hline x & y & \\ \hline 3 & 3 & \end{array}$$

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$
2. $x \leftarrow y$
3. $y \leftarrow \text{temp}$

Kjøring

$$\begin{array}{cc|c} x & y & \\ \hline 2 & 3 & \rightsquigarrow \\ \hline x & y & \\ \hline 3 & 3 & \end{array}$$

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$
2. $x \leftarrow y$
3. $y \leftarrow \text{temp}$

Kjøring

$$\begin{array}{cc|cc} x & y & & \\ \hline 2 & 3 & \rightsquigarrow & \\ \hline x & y & & \\ \hline 3 & 3 & & \end{array}$$

Kjøring

$$\begin{array}{ccc|c} x & y & & \text{temp} \\ \hline 2 & 3 & & \\ \hline \end{array}$$

Eksempler fra boka


Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$ 
2. $x \leftarrow y$
3. $y \leftarrow \text{temp}$

Kjøring

$$\begin{array}{cc|c} x & y & \\ \hline 2 & 3 & \rightsquigarrow \\ \hline x & y & \\ \hline 3 & 3 & \end{array}$$

Kjøring

$$\begin{array}{ccc|c} x & y & & \text{temp} \\ \hline 2 & 3 & & \\ \hline \end{array}$$

Eksempler fra boka


Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$ 
2. $x \leftarrow y$
3. $y \leftarrow \text{temp}$

Kjøring

$$\begin{array}{cc|cc} x & y & \rightsquigarrow & x & y \\ \hline 2 & 3 & & 3 & 3 \end{array}$$

Kjøring

$$\begin{array}{ccc|c} x & y & \text{temp} & \\ \hline 2 & 3 & 2 & \end{array}$$

Eksempler fra boka


Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$
2. $x \leftarrow y$ 
3. $y \leftarrow \text{temp}$

Kjøring

$$\begin{array}{cc|c} x & y & \\ \hline 2 & 3 & \rightsquigarrow \\ \hline x & y & \\ \hline 3 & 3 & \end{array}$$

Kjøring

$$\begin{array}{ccc|c} x & y & & \text{temp} \\ \hline 2 & 3 & & \\ \hline 2 & 3 & & 2 \end{array}$$

Eksempler fra boka

Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$
2. $x \leftarrow y$ ◀
3. $y \leftarrow \text{temp}$

Kjøring

$$\begin{array}{cc|c} x & y & \\ \hline 2 & 3 & \rightsquigarrow \\ \hline x & y & \\ \hline 3 & 3 & \end{array}$$

Kjøring

$$\begin{array}{ccc|c} x & y & \text{temp} & \\ \hline 3 & 3 & 2 & \end{array}$$

Eksempler fra boka


Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$
2. $x \leftarrow y$
3. $y \leftarrow \text{temp}$ 

Kjøring

$$\begin{array}{cc|cc} x & y & \rightsquigarrow & x & y \\ \hline 2 & 3 & & 3 & 3 \end{array}$$

Kjøring

$$\begin{array}{ccc|c} x & y & \text{temp} & \\ \hline 3 & 3 & 2 & \end{array}$$

Eksempler fra boka


Eksempel (1.3.2)

Lag en algoritme som bytter verdien til to variable.

Feil

1. $x \leftarrow y$
2. $y \leftarrow x$

Riktig

1. $\text{temp} \leftarrow x$
2. $x \leftarrow y$
3. $y \leftarrow \text{temp}$ 

Kjøring

$$\begin{array}{cc|cc} x & y & \rightsquigarrow & x & y \\ \hline 2 & 3 & & 3 & 3 \end{array}$$

Kjøring

$$\begin{array}{ccc|c} x & y & \text{temp} & \\ \hline 3 & 2 & 2 & \end{array}$$

Eksempler fra boka

Eksempler fra boka

Eksempel (1.3.3)

Søkere til en jobb tar en test med 20 spørsmål. Lag en algoritme som returnerer en liste av jobbsøkere (identifisert med tall), deres poengsum og en beskjed om hvorvidt de vurderes for stillingen (de som har mer enn 16 poeng) eller settes på venteliste (de med poengsum fra 12 til 15).

Eksempler fra boka

Eksempel (1.3.3)

Søkere til en jobb tar en test med 20 spørsmål. Lag en algoritme som returnerer en liste av jobbsøkere (identifisert med tall), deres poengsum og en beskjed om hvorvidt de vurderes for stillingen (de som har mer enn 16 poeng) eller settes på venteliste (de med poengsum fra 12 til 15).

- Vi bruker *antall_søkere* for å betegne antall søkere.

Eksempler fra boka

Eksempel (1.3.3)

Søkere til en jobb tar en test med 20 spørsmål. Lag en algoritme som returnerer en liste av jobbsøkere (identifisert med tall), deres poengsum og en beskjed om hvorvidt de vurderes for stillingen (de som har mer enn 16 poeng) eller settes på venteliste (de med poengsum fra 12 til 15).

- Vi bruker *antall_søkere* for å betegne antall søkere.
- Vi bruker en **For-do**-løkke, fra 1 til *antall_søkere*, og gjør det samme for hver søker.

Eksempler fra boka

Eksempel (1.3.3)

Søkere til en jobb tar en test med 20 spørsmål. Lag en algoritme som returnerer en liste av jobbsøkere (identifisert med tall), deres poengsum og en beskjed om hvorvidt de vurderes for stillingen (de som har mer enn 16 poeng) eller settes på venteliste (de med poengsum fra 12 til 15).

- Vi bruker *antall_søkere* for å betegne antall søkere.
- Vi bruker en **For-do**-løkke, fra 1 til *antall_søkere*, og gjør det samme for hver søker.
- De 20 svarene som søker *i* har avgitt betegner vi med $a_{i,1}, a_{i,2}, \dots, a_{i,20}$.

Eksempler fra boka

Eksempel (1.3.3)

Søkere til en jobb tar en test med 20 spørsmål. Lag en algoritme som returnerer en liste av jobbsøkere (identifisert med tall), deres poengsum og en beskjed om hvorvidt de vurderes for stillingen (de som har mer enn 16 poeng) eller settes på venteliste (de med poengsum fra 12 til 15).

- Vi bruker *antall_søkere* for å betegne antall søkere.
- Vi bruker en **For-do**-løkke, fra 1 til *antall_søkere*, og gjør det samme for hver søker.
- De 20 svarene som søker *i* har avgitt betegner vi med $a_{i,1}, a_{i,2}, \dots, a_{i,20}$.
- Det riktige svaret på spørsmål *q* betegner vi med c_q .

Eksempler fra boka

Eksempel (1.3.3)

Søkere til en jobb tar en test med 20 spørsmål. Lag en algoritme som returnerer en liste av jobbsøkere (identifisert med tall), deres poengsum og en beskjed om hvorvidt de vurderes for stillingen (de som har mer enn 16 poeng) eller settes på venteliste (de med poengsum fra 12 til 15).

- Vi bruker *antall_søkere* for å betegne antall søkere.
- Vi bruker en **For-do**-løkke, fra 1 til *antall_søkere*, og gjør det samme for hver søker.
- De 20 svarene som søker *i* har avgitt betegner vi med $a_{i,1}, a_{i,2}, \dots, a_{i,20}$.
- Det riktige svaret på spørsmål *q* betegner vi med c_q .
- Vi bruker enda en **For-do**-løkke for å sjekke svarene.

Eksempler fra boka

Eksempler fra boka

Løsning

Eksempler fra boka

Løsning

1. Input *antall_søkere*

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**
 - 2.1. $\text{score} \leftarrow 0$

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**
 - 2.1. $\text{score} \leftarrow 0$
 - 2.2. **For** $q = 1$ **to** 20 **do**

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**
 - 2.1. $\text{score} \leftarrow 0$
 - 2.2. **For** $q = 1$ **to** 20 **do**
 - 2.3. Output i, score
 - 2.4. **If** $\text{score} \geq 16$ **then**

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**
 - 2.1. $\text{score} \leftarrow 0$
 - 2.2. **For** $q = 1$ **to** 20 **do**
 - 2.3. Output i, score
 - 2.4. **If** $\text{score} \geq 16$ **then**
 - 2.4.1. Output 'Anbefalt'
 - else if** $\text{score} \geq 12$ **then**

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**
 - 2.1. $\text{score} \leftarrow 0$
 - 2.2. **For** $q = 1$ **to** 20 **do**
 - 2.3. Output i, score
 - 2.4. **If** $\text{score} \geq 16$ **then**
 - 2.4.1. Output 'Anbefalt'
 - else if** $\text{score} \geq 12$ **then**
 - 2.4.2. Output 'Venteliste'

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**
 - 2.1. $score \leftarrow 0$
 - 2.2. **For** $q = 1$ **to** 20 **do**
 - 2.2.1. Input $a_{i,q}$ [$a_{i,q}$ er søker is svar på spørsmål q .]
 - 2.3. Output $i, score$
 - 2.4. **If** $score \geq 16$ **then**
 - 2.4.1. Output 'Anbefalt'
 - else if** $score \geq 12$ **then**
 - 2.4.2. Output 'Venteliste'

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**
 - 2.1. $score \leftarrow 0$
 - 2.2. **For** $q = 1$ **to** 20 **do**
 - 2.2.1. Input $a_{i,q}$ [$a_{i,q}$ er søker is svar på spørsmål q .]
 - 2.2.2. **If** $a_{i,q} = c_q$ **then**
 - 2.3. Output $i, score$
 - 2.4. **If** $score \geq 16$ **then**
 - 2.4.1. Output 'Anbefalt'
 - else if** $score \geq 12$ **then**
 - 2.4.2. Output 'Venteliste'

Eksempler fra boka

Løsning

1. Input *antall_søkere*
2. **For** $i = 1$ **to** *antall_søkere* **do**
 - 2.1. $score \leftarrow 0$
 - 2.2. **For** $q = 1$ **to** 20 **do**
 - 2.2.1. Input $a_{i,q}$ [$a_{i,q}$ er søker is svar på spørsmål q .]
 - 2.2.2. **If** $a_{i,q} = c_q$ **then**
 - 2.2.2.1. $score \leftarrow score + 1$
 - 2.3. Output $i, score$
 - 2.4. **If** $score \geq 16$ **then**
 - 2.4.1. Output 'Anbefalt'
 - else if** $score \geq 12$ **then**
 - 2.4.2. Output 'Venteliste'

En praktisk forkortelse

En praktisk forkortelse

Pseudokode

En praktisk forkortelse

Pseudokode

1. **If** $x > 0$ **then**

else

En praktisk forkortelse

Pseudokode

1. **If** $x > 0$ **then**
 - 1.1. Output 'Større'
- else**

En praktisk forkortelse

Pseudokode

1. **If** $x > 0$ **then**
 - 1.1. Output 'Større'
- else**
 - 1.2. **If** $x < 0$ **then**

En praktisk forkortelse

Pseudokode

1. **If $x > 0$ then**
 - 1.1. Output 'Større'
- else**
 - 1.2. **If $x < 0$ then**
 - 1.2.1. Output 'Mindre'

En praktisk forkortelse

Pseudokode

1. **If $x > 0$ then**
 - 1.1. Output 'Større'
- else**
 - 1.2. **If $x < 0$ then**
 - 1.2.1. Output 'Mindre'

Forkortet pseudokode

En praktisk forkortelse

Pseudokode

1. **If $x > 0$ then**
 - 1.1. Output 'Større'**else**
 - 1.2. **If $x < 0$ then**
 - 1.2.1. Output 'Mindre'

Forkortet pseudokode

1. **If $x > 0$ then**

else if $x < 0$ then

En praktisk forkortelse

Pseudokode

1. **If $x > 0$ then**
 - 1.1. Output 'Større'**else**
 - 1.2. **If $x < 0$ then**
 - 1.2.1. Output 'Mindre'

Forkortet pseudokode

1. **If $x > 0$ then**
 - 1.1. Output 'Større'**else if $x < 0$ then**

En praktisk forkortelse

Pseudokode

1. **If $x > 0$ then**
 - 1.1. Output 'Større'**else**
 - 1.2. **If $x < 0$ then**
 - 1.2.1. Output 'Mindre'

Forkortet pseudokode

1. **If $x > 0$ then**
 - 1.1. Output 'Større'**else if $x < 0$ then**
 - 1.2. Output 'Mindre'

En oppgave

En oppgave

Oppgave (1.1)

Finn det minste tallet i en liste av tall, samt indeksen til det minste tallet.

En oppgave

Løsning

[Ny kode]

En oppgave

Løsning

[Ny kode]

1. Input the number of values n

En oppgave

Løsning

[Ny kode]

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n

En oppgave

Løsning

[Ny kode]

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$; **indeks** $\leftarrow 1$ ◀

En oppgave

Løsning

[Ny kode]

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$; **indeks** $\leftarrow 1$ ◀
4. **For** $i = 2$ **to** n **do**

En oppgave

Løsning

[Ny kode]

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$; **indeks** $\leftarrow 1$ ◀
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**

En oppgave

Løsning

[Ny kode]

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$; **indeks** $\leftarrow 1$ ◀
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$

En oppgave

Løsning

[Ny kode]

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$; **indeks** $\leftarrow 1$ ◀
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$
 - 4.1.2. **indeks** $\leftarrow i$ ◀

En oppgave

Løsning

[Ny kode]

1. Input the number of values n
2. Input the list of numbers x_1, \dots, x_n
3. $\text{min} \leftarrow x_1$; **indeks** $\leftarrow 1$ ◀
4. **For** $i = 2$ **to** n **do**
 - 4.1. **If** $x_i < \text{min}$ **then**
 - 4.1.1. $\text{min} \leftarrow x_i$
 - 4.1.2. **indeks** $\leftarrow i$ ◀
5. Output min **og** **indeks** ◀