

# 1 Flervalg

## Hvorfor er det ønskelig å bruke SSD fremfor en vanlig harddisk?

- ☐ En SSD øker minnet på grafikkortet slik at spill og lignende flyter bedre.
- ☐ En SSD er ikke så utsatt for strømtopper og tåler derfor mer enn en magnetisk disk.
- ☐ Det er lettere å lagre data permanent på en SSD.
- ☐ I en SSD lagres data i elektroniske kretser. Det er ingen bevegelige deler, og dermed blir disken raskere og mer pålitelig.

## RAM står for random acces memory, men hva menes egentlig med random i dette tilfellet?

- ☐ At vi henter ut tilfeldig data.
- ☐ At dataen kommer i tilfeldig rekkefølge.
- ☐ At hukommelsescellene kan aksesseres direkte i tilfeldig rekkefølge.
- ☐ At det er tilfeldig hvor dataen kommer fra.

## Hvilke 5 steg er med i "Fetch/Execute Cycle"?

- ☐ Instruction Fetch(IF), Data Fetch(DF), Instruction Decode(ID), Data Decode(DD), Result Return(RR)
- ☐ Instruction Decode(ID), Instruction Fetch(IF), Instruction Execute(EX), Data Fetch(DF), Data Decode(DD)
- ☐ Instruction Fetch(IF), Instruction Execute(EX), Instruction Decode(ID), Data Decode(DD), Result Return(RR)
- ☐ Instruction Fetch(IF), Data Fetch(DF), Instruction Decode(ID), Result Return(RR), Instruction Execute(EX)

**Ca. hvor mange ulike operasjoner kan en datamaskin utføre?**

- ☐ 100
- ☐ 10
- ☐ 10000
- ☐ 1000

**Hvilket av alternativene under er IKKE korrekt?**

- ☐ Det er operativsystemet som tar seg av minnehåndtering og oppstart av maskinen.
- ☐ Python er et høynivå programmeringsspråk.
- ☐ Pipelining er en teknikk der en CPU kan utføre flere instruksjoner parallelt.
- ☐ Assembly er et operativsystem.

Maks poeng: 5

**2 Oppgave 1a)**

```
a = True
b = False
if (a or b) and not (a and b):
    print("Yummy")
else:
    print("Yum")
```

**Hva blir skrevet ut til skjerm når kodesnutten over kjører?**

- ☐ Yummy
- ☐ Yum
- ☐ Ingen av de over

Maks poeng: 1

### 3 Oppgave 1b)

```
vari = 4
for i in range(Vari):
    print('#'*i)
```

**Påstand: Kodesnutten over vil kjøre uten problemer**

- ☐ Sant
- ☐ Usant

-----  
-----

```
vari = 3
while vari:
    print('#')
    vari -= 1
```

**Påstand: Kodesnutten over vil kjøre evig**

- ☐ Usant
- ☐ Sant

-----  
-----

```
vari = 3
while vari:
    print('#'*vari)
    vari -= 2
```

**Påstand: Kodesnutten over vil kjøre evig**

- ☐ Usant
- ☐ Sant

Maks poeng: 3

## 4 Oppgave 1c)

```
a = True
b = False
x = 10
while a and not b and x:
    x -= 1
print(x)
```

**Hva blir skrevet ut til skjerm når kodesnutten over kjører?**

- ☐ 10
- ☐ Løkken kjører evig. 10 9 8 7 osv. skrives ut til skjerm.
- ☐ 0
- ☐ Løkken kjører evig. Ingenting blir skrevet ut til skjerm.

Maks poeng: 1

## 5 Oppgave 1d)

Du ønsker å lagre informasjon tilknyttet en hekkeløp-konkurranse. For hver idrettsutøver skal du lagre deres navn, alder, hvilken plass de kom på i løpet, hvor lang tid de brukte, og om de har vært med i VM. (Du kan gå ut ifra at alle fullfører løpet, dvs. du har en verdi for tid for alle utøverne.)

Hvilken variabeltype(integer/float/string/boolean) ville du brukt i tilordning av denne informasjon om en idrettsutøver:

### Finn de som passer sammen

	Integer	Boolean	String	Float
Navn	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Alder	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Plassering	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Tid (i sekunder)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Har deltatt i VM	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Maks poeng: 5

## 6 Oppgave 1e)

```
if a > b:
    print("Hello")
elif a == b:
    print("Nice to meet you")
else:
    print("Bye")
```

Hva vil skrives til skjerm når koden over kjører med følgende verdier:

	Hello	Nice to meet you	Bye
a = 35, b = 98	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
a = "One", b = "Four"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
a = 42, b = 42.00	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
a = "Hmm", b = "Si det"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Maks poeng: 4

## 7 Oppgave 2a)

For hvert av feltene under, skal du fylle inn rett regneoperasjon. Variablene  $a$  og  $b$  er heltall, f.eks. 7 og 3.

1. `print(a, "added to", b, "is", a  b)`

2. `print(a, "subtracted from", b, "is", a  b)`

3. `print(a, "divided by", b, "is", a  b)`

4. `print("Integer division of", a, "and", b, "gives", a  b)    # Integer  
division = heltallsdivisjon`

5. `print(a, "modulo", b, "is", a  b)`

6. `print("The square of", b, "is", b  2)`

Maks poeng: 6

## 8 Oppgave 2b)

Du har fått følgende kode, men dessverre mangler den noen deler:

```
num_1 = int(...("A number please: "))  
num_2 = 42  
print(..., "multiplied with", num_2, "is", ...)
```

Din oppgave er å fikse opp i dette.

**Plasser alternativene på korrekt sted i koden.**

 [Hjelp](#)

input

num\_2

num\_1 x num\_2

num\_1\*\*num\_2

print

num\_1\*num\_2

str

num\_1

num\_1 = int\_(  ("A number please: "))

num\_2 = 34

print(  , "multiplied with", num\_2, "is",  )

Maks poeng: 3



## 9 Oppgave 2c)

Du har fått enda en kodesnutt, men nok en gang er det noe kode som mangler.

Kodesnutten er:

```
bottom_limit = int(input("Bottom limit: "))
upper_limit = int(input("Upper limit: "))
num = int(input("Number to check: "))
result = 0

for i in range(...):
    if ...:
        result += 1

if result == 1:
    print(result, ..., num, "in the interval [", bottom_limit, ",", upper_limit,
"]".)
else:
    print(result, ..., num, "in the interval [", bottom_limit, ",", upper_limit,
"]".)
```

Også denne gangen er din oppgave å fylle inn de delene som mangler.

Om *bottom\_limit* = 2, *upper\_limit* = 9, og *num* = 3, skal koden finne hvor mange tall i intervallet [2, 9] som er delelig på 3 og lagre dette i variabelen *result*. Videre skal koden skrive til skjerm dette resultatet. Altså vil det i dette tilfellet bli skrevet til skjerm:

*3 numbers are divisible by 3 in the interval [ 2 , 9 ].*

Koden skal også ta hensyn til grammatikken i svaret. Altså skal den om *result* = 1 skrive til skjerm:

*1 number is divisible by ...*

**Dra de elementene som trengs dit de hører til for at koden blir komplett og oppfyller kravene over.**



bottom\_limit, upper\_limit+1

"number is divisible by"

i / 3 == i % 3

"numbers are divisible by"

bottom\_limit, upper\_limit

i % num == 0

i / 3 == int

```
bottom_limit = int(input("Bottom limit: "))
upper_limit = int(input("Upper limit: "))
num = int(input("Number to check: "))
result = 0
```

```
for i in range( ):
    if :
        result += 1
```

```
if result == 1:
    print(result, , num, "in the interval [", bottom_limit, ",",
upper_limit, "].")
else:
    print(result, , num, "in the interval [", bottom_limit, ",",
upper_limit, "].")
```

Maks poeng: 4

## 10 Oppgave 2d)

I matematikk er en geometrisk rekke en rekke der forholdet mellom alle leddene er konstant.

For eksempel er

$$1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \dots$$

en geometrisk rekke fordi hvert ledd er lik det forrige ganget med  $\frac{1}{2}$

Det kan vises at dersom antall ledd i rekken går mot uendelig vil rekken konvergere (summen går mot en endelig verdi) hvis forholdet mellom hvert ledd ligger mellom -1 og 1:

$$1 + r + r^2 + r^3 + r^4 + \dots = \sum_{n=0}^{\infty} r^n = \frac{1}{1-r} \quad |r| < 1$$

Fullfør funksjonen Geom(r) som regner ut verdien til en uendelig geometrisk rekke for et tilfeldig flyttall r mellom -1 og 1:



Hjelp







def



=

.

return result

Maks poeng: 3

## 11 Oppgave 3a)

I denne oppgaven skal du lage et program for å regne ut hvor mye vekslепenger en kunde skal få. Programmet skal spørre brukeren om prisen på varene og hvor mye kunden betalte. Deretter skal programmet regne ut hvor mye vekslепenger brukeren skal få.

Om kunden har betalt nok, skal programmet skrive følgende til skjerm:

*Prisen er ... kr. Du betalte ... kr. Tilbake ... kr.*

Om kunden prøver å lure seg unna, og ikke har betalt nok, skal programmet skrive til skjerm:

*Prisen er ... kr. Du betalte bare ... kr. Beløp som mangler er ... kr.*

Eksempel 1 på kjøring:

Pris på varene: 29

Betalt beløp: 200

Prisen er 29 kr. Du betalte 200 kr. Tilbake 171 kr.

Eksempel 2 på kjøring:

Pris på varene: 67

Betalt beløp: 42

Prisen er 67 kr. Du betalte bare 42 kr. Beløp som mangler er 25 kr.

**Skriv ditt svar her...**

1	
---	--

Maks poeng: 10

## 12 Oppgave 3b)

I denne oppgaven skal du lage en annen versjon av vekslepenge-programmet. Her spørres fortsatt brukeren om prisen på varene og hvor mye kunden betalte.

Det som er nytt, er at programmet maser på kunden om å betale mer, og tar inn dette beløpet. Når beløpet kunden har betalt er større eller lik prisen på varene, skriver programmet ut hvor mye kunden skal ha tilbake i vekslepenger.

Eksempel på kjøring:

Pris på varene: 342

Betalt beløp: 250

Prisen er 342 kr. Du har bare betalt 250 kr. Beløp som mangler er 92 kr.

Hvor mye mer gir du? 42

Prisen er 342 kr. Du har bare betalt 292 kr. Beløp som mangler er 50 kr.

Hvor mye mer gir du? 28

Prisen er 342 kr. Du har bare betalt 320 kr. Beløp som mangler er 22 kr.

Hvor mye mer gir du? 76

Prisen er 342 kr. Du betalte 396 kr. Tilbake 54 kr.

**Skriv ditt svar her...**

1	
---	--

Maks poeng: 10



### 13 Oppgave 3c) - Litt vanskelig

I denne oppgaven skal du skrive et nytt veksle-program. Også dette programmet skal spørre etter prisen kunden må betale og beløpet kunden har betalt.

Det som er nytt er at programmet deretter skal gi beskjed om hvor mange hundre-lapper, femti-lapper, tjue-kroninger, ti-kroninger, fem-kroninger og en-kroninger kunden skal ha tilbake.

(Du kan i denne oppgaven gå ut ifra at vekslepengene befinner seg i intervallet  $[0, 199]$  kr. Dvs. at du ikke trenger å tenke på to-hundrelapper og over, i tillegg vil kunden alltid betale summen eksakt eller mer.)

Eksempel på kjøring:

Pris på varene: 29

Betalt beløp: 200

Prisen er 29kr. Du betalte 200kr. Tilbake: 171kr.

Her er vekslepengene dine: 1 hundrelapp(er), 1 femtilapp(er), 1 tjuekrone(r),

0 tikrone(r), 0 femkrone(r), 1 enkrone(r).

**Skriv ditt svar her...**

1



## 14 Oppgave 4a)

Et primtall er et tall som kun er delelig på seg selv og 1. De første ti primtallene er 2, 3, 5, 7, 11, 13, 17, 19 og 23.

I denne oppgaven skal du bruke nedtrekksmenyene til å fullføre funksjonen *isPrimeNumber(n)* som sjekker om et tall er primtall. Funksjonen skal ta inn et positivt heltall *n* og returnere enten *True* eller *False* avhengig om *n* er primtall eller ikke. Merk at ingen heltall mindre eller lik 1 er primtall!

### Fullfør isPrimeNumber(n):

```
def isPrimeNumber(n):  
    if n <= 1:  
        return  (False, n, isPrime, True)  
    isPrime = True  
    for i in range(2, n//2 + 1):    #Holder å sjekke opp til og med n/2  
        if  (n%i, i//n, i%n, n//i) == 0:  
            isPrime = False  
            break  
    return  (isPrime, False, i, True)
```

Maks poeng: 3

## 15 Oppgave 4b)

I denne oppgaven skal du lage et program som bruker en valgfri løkke og funksjonen `isPrimeNumber(n)` fra forrige oppgave til å skrive ut alle primtall fra 2 til og med 100.

**Du trenger ikke ha fått til forrige oppgave for å løse denne!**

Tips:

`isPrimeNumber(n)` tar inn et heltall  $n$  og returnerer True hvis  $n$  er primtall og False hvis ikke. Det betyr at du kan bruke funksjonen i en betingelse som dette:

```
if isPrimeNumber(23):  
    print("Fant et primtall :D")
```

**Skriv ditt svar her...**

1	
---	--

Maks poeng: 10

## 16 Bonus 1: Konvertering

Fullfør koden under, slik at funksjonen `converting_stuff(stuff, conv_to)` endrer variabelen `stuff` til et heltall om `conv_to='i'`, endrer den til et flyttall om `conv_to='f'`, endrer den til et tegn om `stuff` er et heltall og `conv_to='c'`, og for alle andre verdier av `conv_to` endrer `stuff` til en streng.

```
def converting_stuff(stuff, conv_to):
    if conv_to == 'i':
         (stuff = roof(stuff), stuff = round(stuff), stuff =
int(stuff), stuff = float(stuff))
    elif conv_to == 'f':
         (stuff = str(stuff), stuff = int(stuff), stuff =
round(stuff), stuff = float(stuff))

     (elif
instance(conv_to, int) and stuff == 'c':, elif isinstance(stuff, int) and
conv_to == 'c':, elif not isinstance(stuff, int) and conv_to == 'c':, elif
instance(stuff, int) or conv_to == 'c':)
        stuff = chr(stuff)
    else:
        stuff = str(stuff)
    return stuff
```

Maks poeng: 3

## 17 Bonus 2: Gangetabellen

Lag et program som skriver ut gangetabellen for opptil  $n$  tall på et pent format!

*Hint: Nøstede løkker.*

Eksempel,  $n=4$ :

	1	2	3	4
1	1	2	3	4
2	2	4	6	8
3	3	6	9	12
4	4	8	12	16

**Skriv ditt svar her...**

1

Maks poeng: 10