Oppgaver: Løkker, betingelser og tekstbehandling

Oppgave 1 *For-løkker*

Bruk en `for`-løkke til å gjennomføre disse oppgavene.

a) Print meldingen `'Hei, verden'` 5 ganger.

b) Print alle tallene fra 1 til 101.

Løsning oppgave 1 *For-løkker*

a)

```
for verdi in range(5):
    print("Hei, verden")
```

b)

```
for tall in range(1,101):
    print(tall)
```

Oppgave 2 *To for-løkke-eksempler*

Under ser du to kodesnutter som tar i bruk `for`-løkker.

Første kodesnutt:

```
for tall in range(3):
  print(f"Tallet er {tall}")
  print("Ha en fin dag!")
```

Andre kodesnutt:

```python
for tall in range(3):
  print(f"Tallet er {tall}")
print("Ha en fin dag!")
```

a) Hvordan ser utskriften ut for hver av kodesnuttene?

b) Hva er forskjellen mellom kodesnuttene? Og hvorfor er denne forskjellen viktig?

Løsning oppgave 2 *To for-løkke-eksempler*

a) Kodesnutt 1:

```
Tallet er 0
Ha en fin dag!
Tallet er 1
Ha en fin dag!
Tallet er 2
Ha en fin dag!
```

Kodesnutt 2:

```
Tallet er 0
Tallet er 1
Tallet er 2
Ha en fin dag!
```

b) Forskjellen er innrykket på linje 3. Første kodesnutt har et innrykk slik at linja hører inn under den kodeblokken som gjentas av for løkka. Det betyr at "Ha en fin dag!" vil bli skrevet ut for hver runde i løkka. Andre kodesnutt har ikke et innrykk og linja hører dermed ikke til løkke-blokken. "Ha en fin dag!" vil derfor kun bli skrevet ut en gang etter at løkka er ferdig. Forskjellen i innrykk-nivå er viktig fordi Python bruker innrykk for å bestemme hvilke linjer kode som hører sammen i kodeblokker.

Oppgave 3 *Tacofredag til torsdag*

Stian har et program for å automatisk sende epost til sine kollegaer for å invitere til sammenkomster. For å invitere til fredagstaco skrev han en invitasjon inn i en tekststreng slik:

```
1 invitasjonstekst = """
2 Hei alle! Førstkommende fredag har alle som ønsker
     muligheten til å bli med på årets første
     fredagstaco!
3 Spør du meg så finnes ingen bedre måte å feire
     fredagen på enn med taco!!
4 Vi sees (forhåpentligvis) på fredag,
5 Stian
6 """
```

Etter å ha skrevet inn teksten fant Stian ut at fredag ikke passer bra fordi det overlapper med vaffelkvelden til en kollega. Han vil derfor modifisere invitasjonen slik at det står torsdag i stedet for fredag.

a) Lim invitasjonsstrengen over inn i en `Python` fil

b) Bruk `replace` til å bytte ut alle steder det står "fredag" med "torsdag" i invitasjonsteksten.

c) Skriv ut den nye invitasjonen til terminalen

Løsning oppgave 3 *Tacofredag til torsdag*

a)

```
invitasjonstekst = """
Hei alle! Førstkommende fredag har alle som ønsker
    muligheten til å bli med på årets første
   fredagstaco!
Spør du meg så finnes ingen bedre måte å feire
   fredagen på enn med taco!!
Vi sees (forhåpentligvis) på fredag,
Stian
"""
```

b)

```
ny_invitasjonstekst = invitasjonstekst.replace("
   fredag", "torsdag")
```

c)

```
print(ny_invitasjonstekst)
```

```
Hei alle! Førstkommende torsdag har alle som ø
   nsker muligheten til å bli med på årets første
   torsdagstaco!
Spør du meg så finnes ingen bedre måte å feire
   torsdagen på enn med taco!!
Vi sees (forhåpentligvis) på torsdag,
Stian
```

Oppgave 4 *Måneder med r i navnet*

Mila har hørt at hun bør få i seg ekstra D-vitamin i måneder som inneholder "r". Så hun har lyst til å bruke Python til å løkke gjennom alle månednavnene og skrive ut kun de som inneholder bokstaven "r".

a) Opprett en liste, `månednavn` som inneholder navnet på alle 12 måneder

b) Bruk en `for`-løkke til å løkke igjennom disse månedene og skriv ut hvert månednavn til terminalen

c) Modifiser programmet til å bruke en betingelse for å kun skrive ut de månedene som inneholder bokstaven `r`. (**Hint**: du kan bruke `in`-operatoren for å se om en tekststreng er inneholdt i en annen tekststreng)

Løsning oppgave 4 *Måneder med r i navnet*

a)

```
1 månednavn = ["Januar", "Februar", "Mars", "April",
      "Mai", "Juni", "Juli", "August", "September",
    "Oktober", "November", "Desember"]
```

b)

```
1 for måned in månednavn:
2     print(måned)
```

c)

```
1 for måned in månednavn:
2     if "r" in måned
3         print(måned)
```

Oppgave 5 *Lipogram*

I 1939 tok forfatter Ernest Vincent Wright på seg utfordringen å skrive en hel roman uten å bruke bokstaven "e". Boken heter *Gadsby* og er over 50 000 ord lang! Tekster som unngår en spesiell bokstav kalles et *lipogram* og det å unngå bokstaven "e" er spesielt vanskelig siden det er den mest brukte bokstaven i engelsk. La oss bruke Python til å teste en bit av *Gadsby* og dobbeltsjekke at Wright ikke har sneket inn noen "e"-er.

a) Lagre første avsnitt av *Gadsby* i en variabel ved å kopiere koden under:

```python
gadsby_avsnitt = """If Youth, throughout all history, had had a champion to stand up for it; to show a doubting world that a child can think; and, possibly, do it practically; you wouldn't constantly run across folks today who claim that "a child don't know anything." A child's brain starts functioning at birth; and has, amongst its many infant convolutions, thousands of dormant atoms, into which God has put a mystic possibility for noticing an adult's act, and figuring out its purport"""
```

b) Bruk `in` operatoren til å sjekke om bokstaven `'e'` finnes i avsnittet. Skriv ut resultatet til terminalen

c) **BONUS:** Den andre mest vanlige bokstaven i engelsk er bokstaven "a". Bruk en `for`-løkke til å løkke over hver bokstav i teksten og sjekk om det er en `'a'`. Opprett en variabel som teller antall `'a'`-er i teksten ved å øke med `1` hver gang en bokstav er `'a'`.

Løsning oppgave 5 *Lipogram*

a)

```python
gadsby_avsnitt = """If Youth, throughout all
   history, had had a champion to stand up for it;
    to show a doubting world that a child can
   think; and, possibly, do it practically; you
   wouldn't constantly run across folks today who
   claim that "a child don't know anything." A
   child's brain starts functioning at birth; and
   has, amongst its many infant convolutions,
   thousands of dormant atoms, into which God has
   put a mystic possibility for noticing an adult'
   s act, and figuring out its purport"""
```

b)

```python
if "e" in gadsby_avsnitt:
  print("Jeg fant en e!")
else:
  print("Ingen e!")
```

c)

```python
antall_a = 0
for bokstav in gadsby_avsnitt:
  if bokstav == "a":
    antall_a += 1
print(antall_a)
```

Bonusoppgaver: Løkker, betingelser og tekstbehandling

Oppgave 6 *Tolke feilmeldinger*

I denne oppgaven blir du presentert for noen av de vanligste feilmeldingene man kan få når man programmerer. Her trenger du ikke programmere noe, men les meldingen nøye, og beskriv hva du tror kan føre til en slik feilmelding. Finn også ut hvilken linje feilen er på.

a)

```
File "test.py", line 4
    if svar = "Oslo":
            ^
SyntaxError: invalid syntax
```

b)

```
Traceback (most recent call last):
  File "test.py", line 5, in <module>
    if tall > 10:
TypeError: '>' not supported between instances of
   'str' and 'int'
```

c)

```
File "test.py", line 5
    if tall < 0
               ^
SyntaxError: invalid syntax
```

d)

```
 File "test.py", line 6
    print(f"Hei {navn}!")
    ^
IndentationError: expected an indented block
```

Løsning oppgave 6 *Tolke feilmeldinger*

a) Her bruker vi enkel `=`, men for likhetsbetingelse må vi bruke dobbelt likhetstegn (`==`). Feilen er på linje 4.

b) Vi prøver å sammenligne størrelsen på en variabel med type `str` og en med type `int`. Feilen er på linje 5.

c) Her har vi glemt et kollon. feilen er på linje 5

d) Her har vi glemt et innrykksnivå (en blokk med kode som skulle vert

rykket inn et hakk er ikke det). Feilen er på linje 6.

Oppgave 7 *For-løkker for tallrekker*

Kodesnutten under bruker en `for`-løkke til å skrive ut tallene fra 0 til (men ikke med) 10.

```
1 for tall in range(0, 10, 1):
2     print(tall)
```

modifiser koden for å skrive ut følgende tallrekker:

a) Skriv ut alle tall fra 0 til (men ikke med) 42

b) Skriv ut alle partall fra 0 til og med 20

c) Skriv ut hele 7-gangen fra 0 til og med 70

Løsning oppgave 7 *For-løkker for tallrekker*

a)

```
1 for tall in range(43):
2     print(tall)
```

b)

```
1 for tall in range(11):
2     print(2*tall)
```

c)

```
1 for tall in range(11):
2     print(7*tall)
```

Oppgave 8 *For-løkker for hånd*

For hver løkke, gå igjennom for hånd og forutsi hva som skrives ut. Etterpå kan du kjøre løkkene og se hva om du hadde rett:

a)

```
for tall in range(1, 5):
    print(tall)
```

b)

```
sum = 0
for tall in range(1, 5):
    sum += tall
    print(sum)
```

c)

```
produkt = 0
for tall in range(1, 5):
    produkt *= tall
    print(produkt)
```

d)

```
produkt = 1
for tall in range(1, 5):
    produkt *= tall
    print(produkt)
```

e)

```
x = 1
for _ in range(10):
    x *= 2
    print(x)
```

Løsning oppgave 8 *For-løkker for hånd*

a)

```
1
2
3
4
```

b)

```
1
3
6
10
```

c)

```
0
0
0
0
```

d)

```
1
2
6
24
```

e)

```
2
4
8
16
32
64
128
256
512
1024
```

Oppgave 9 *Røverspråk-generator*

Røverspråket er et kodespråk som ble popularisert av Astrid Lindgrens romaner om mesterdetektiven Kalle Blomkvist.

For å snakke røverspråket må du ta hver konsonant du skal si, legge på en 'o' og så gjenta konsonanten. Si for eksempel at du skal si ordet katt. Her er bokstavene 'k' og 't' konsonanter, så på røverspråket ville du sagt: kok-a-tot-tot.

Vi skal nå skrive et program som kan oversette en setning til røverspråk for oss. Du kan enten følge deloppgavene under for å løse problemet steg for steg. Eller du kan prøve deg frem på egenhånd for en større utfordring.

La oss begynne med å lage et program som klarer å oversette "katt" til "kokatottot". Etter vi har fått til det kan vi gjøre programmet mer generelt.

a) Lag en strengvariabel som inneholder strengen "katt"

Lag deretter en for-løkke som går igjennom hver bokstav i strengvariabelen og skriver den ut med print. Om du kjører koden din bør du se

```
k
a
t
t
```

b) Neste steg er å sjekke om hver bokstav er en vokal eller en konsonant før vi skriver den ut. For å gjøre det bør du opprette en ny strengvariabel som heter `konsonanter` som inneholder alle konsonantene (`"bcdfghjklmnpqrstvwxz"`).

Du kan nå bruke betingelsen `if bokstav in konsonanter:` for å sjekke om en bokstav er en vokal eller ikke. Lag en if-else test som gjør dette for hver bokstav. Om bokstaven er en konsonant, skriver du ut konsonanten, så en 'o' og så konsonanten på nytt. Om bokstaven er en vokal, skriver du den bare rett ut. Når du kjører koden din bør du nå se

```
kok
a
tot
tot
```

c) Nå gjenstår det bare å få alle utskriftene på samme linje. For å gjøre dette skal vi istedenfor å skrive ut hver bokstav separat, først oversette hele ordet, og deretter skrive ut hele det nye ordet i ett. For å gjøre dette, opprett en ny tekstvariabel som er tom (du kan gjøre dette ved å f.eks skrive `røverspråk = ''`. I løkka, bytt `print`-kommandoene med å istedet legge bokstavene til den nye strengvariabelen din (du kan bruke `+=` for å gjøre dette). På denne måten bygger du opp strengen bokstav for bokstav. Husk å skrive ut den nye strengvariabelen din til slutt! Når du er ferdig bør koden din gi følgende resultat:

```
kokatottot
```

d) Prøv å erstatte k̈attm̈ed en lengre setning, og skriv ut hva denne blir på røverspråket. Måtte du justere noe? Når du har fått til dette steget har du skrevet ditt eget program for å oversette noe til røverspråket!

Løsning oppgave 9 *Røverspråk-generator*

a)

```
tekst = "katt"

for bokstav in tekst:
    print(bokstav)
```

b)

```
konsonanter = "bcdfghjklmnpqrstvwxz"

for bokstav in tekst:
    if bokstav in konsonanter:
        print(bokstav + "o" + bokstav)
    else:
        print(bokstav)
```

c)

```
røverspråk = ""

for bokstav in tekst:
    if bokstav in konsonanter:
        røverspråk += bokstav + "o" + bokstav
    else:
        røverspråk += bokstav

print(røverspråk)
```

d)

```python
tekst = "Jeg lager fine Python-programmer!"

røverspråk = ""

for bokstav in tekst:
    if bokstav in konsonanter:
        røverspråk += bokstav + "o" + bokstav
    else:
        røverspråk += bokstav

print(røverspråk)
```

Oppgave 10 *Navn og alder*

Husk at du kan hente ut deler av en tekststreng ved å skrive for eksempel

```python
tekst_del1 = tekst[5:]
enkel_bokstav = tekst[4]
tekst_del2 = tekst[:4]
```

Her skal vi bruke dette til å hente ut litt nyttig informasjon fra en setning.

a) Opprett en variabel `setning` som inneholder «Hei, jeg heter Emilie og er 25 år gammel.» Du kan bytte ut navn og alder med dine egne.

b) Bruk indeksering til å hente ut navnet, slik at du får en variabel `navn` som har verdien «Emilie». Skriv den ut for å sjekke at det blir riktig.

c) Bruk indeksering til å hente ut alderen, slik at du får en variabel `alder` som har verdien «25». Skriv den ut for å sjekke om det blir riktig.

d) Endre navn og alder i setningen. Fungerer programmet ditt fortsatt? Er det noe du må endre på? Tenk gjerne på antall bokstaver det er i navnet, og antall sifre i alderen.

(Her er det meningen å bare fikle litt, ikke å lage en løsning som fungerer for alle navn og aldre.)

Løsning oppgave 10 *Navn og alder*

```
setning = "Hei, jeg heter Emilie og er 25 år gammel."
```

```
navn = setning[15:21]
print(navn)
```

```
alder = setning[28:30]
print(alder)
```

Her må vi bare justere litt:

```
setning = "Hei, jeg heter Mia og er 9 år gammel."

navn = setning[15:18]
print(navn)

alder = setning[25:26]
print(alder)
```

Generelt er det vanskelig å lage et program som tar hensyn til alt. Hva skjer f.eks. om du ikke tar med «Hei»? Hva ville utfordringene vært dersom dette ble gitt som `input`?