UNIVERSITETET I BERGEN

KANDIDAT

241

PRØVE

INF100 0 Innføring i programmering

Emnekode	INF100
Vurderingsform	Skriftlig eksamen
Starttid	28.11.2022 14:00
Sluttid	28.11.2022 18:00
Sensurfrist	--
PDF opprettet	04.11.2023 15:22

Info

Oppgave	Oppgavetype
ℹ	Informasjon eller ressurser
ℹ	Informasjon eller ressurser

Automatisk rettet

Oppgave	Oppgavetype
1	Paring
2	Flervalg
3	Nedtrekk
4	Nedtrekk
5	Nedtrekk
6	Nedtrekk
7	Flervalg

Korte kodesnutter

Oppgave	Oppgavetype
8	Programmering
9	Programmering

Forklaring

Oppgave	Oppgavetype
10	Langsvar
11	Langsvar

Kode

Oppgave	Oppgavetype
12	Programmering
13	Programmering
14	Programmering

1 a = [1, 2.2, "3", "a"]
b = "foo"
c = 42
d = { "a":"b", "b":"c" }
e = True

Velg riktig datatype for uttrykket

	int	dict	None	(-error-)	list	str	bool	fl
f"{c}"	○	○	○	○	○	◉	○	
e and not e	○	○	○	○	○	○	◉	
a[0]	◉	○	○	○	○	○	○	
d["c"]	○	○	○	◉	○	○	○	
a in a	○	○	○	○	○	○	◉	
c < 42	○	○	○	○	○	○	◉	
a[0:1]	○	○	○	○	◉	○	○	
e	○	○	○	○	○	○	◉	
b + 2	○	○	○	◉	○	○	○	
b + "a[0]"	○	○	○	○	○	◉	○	
9 * a[1]	○	○	○	○	○	○	○	
d[a[-1]] == b	○	○	○	○	○	○	◉	

0.5 poeng per riktig svar, 0 poeng for ubesvart eller feil svar.

Maks poeng: 6

2 Hint: bruk gjerne presedenstabellen i kursnotatene for å minne deg selv på hvilken operator som har presedens.

Hvordan plassere parenteser for å få et uttrykk *identisk* med

12 // 2 * 3

Velg ett alternativ

- 12 // (2 * 3)
- (12 // 2) * 3

Hvordan plassere parenteser for å få et uttrykk *identisk* med

x and y or z in a

Velg ett alternativ:

- x and (y or (z in a))
- x and ((y or z) in a)
- (x and y) or (z in a)
- (x and (y or z)) in a
- ((x and y) or z) in a

2 poeng gis for hvert riktig svar, 0 poeng for feil svar eller ubesvart.

Maks poeng: 4

3 Velg riktig verdi for hvert uttrykk.

a	b	c	a or (b and c)	(not a) and b
True	True	True	True (True, False)	False (True, False)
False	False	True	False (True, False)	False (True, False)
False	True	True	True (True, False)	True (True, False)
False	True	False	False (True, False)	True (True, False)
0	"foo"	"bar"	"bar" (True, False, 0, "foo", "bar", (-error-))	"foo" (True, False, 0, "foo", "bar", (-error-))

0 poeng for feil svar, 0.2 poeng for ubesvart, og 0.5 poeng per riktig svar

Maks poeng: 5

4 Velg slik at alle uttrykkene evaluerer til *True*. Pass på at typene blir riktig. Oppslagsverket *d* ser slik ut:

```
d = {
    42 : 0,
    '42' : 'bar',
    'foo' : 42,
    95 : 'foo',
    'bar' : 95,
}
```

d[95] == 'foo' (0, '95', 95, 'bar', 42, '42', 'foo')

'bar' == d['42'] (d.95, d['42'], d(42), d[d[0]], d[42], d.get(95))

0 in d.values() (d, d.items(), [d.get(0)], d.values(), d.keys())

'foo' == d[d['bar']] (d[d['42']], d[d[95]], d[-42], d[d['foo'], d[d[42]], d[d['bar']], d[d[0]])

1 poeng per riktig svar, 0 poeng for ubesvart eller feil svar.

Maks poeng: 4

5 Velg slik at alle uttrykkene evaluerer til *True*.

a = [[1, 2, 3], "foo", [42], 95]

a[0] == [1, 2, 3] ("1", [1, 2, 3], [1], "foo", 1)

"foo" == a[1] (a[1], a[2], a.find("foo"), a[4], a[-2], a[3], a[1:2])

95 == a[-1] (a[-0], a[4:], a[-1], a.pop(95), a[3:4])

42 **in** a[2] (a, a[2], a[3], a[4], a[0][2], a[-2][0])

3 == a[::4][-1][-1] (a[::4][-1][-1], a[0, 2], a[0:2], a[0[2]], a[:1][2], len(a[::2]))

1 poeng per riktig svar, 0 poeng for ubesvart eller feil svar.

Maks poeng: 5

6 Velg de riktige linjene slik at utskriften blir:
A
D

x = "foo"

if len(x) <= 3: (if len(x) > 3:, if len(x) <= 3:, elif "foo" == x:, if "x" in x:)

- print("A")

elif "foo" == x: (if x[1] == x[-1]:, if x[1] == x[2]:, if len(x) == 3:, elif "foo" == x:)

- print("B")

if not True: (elif not False:, if not False:, if not True:, elif not True:)

- print("C")

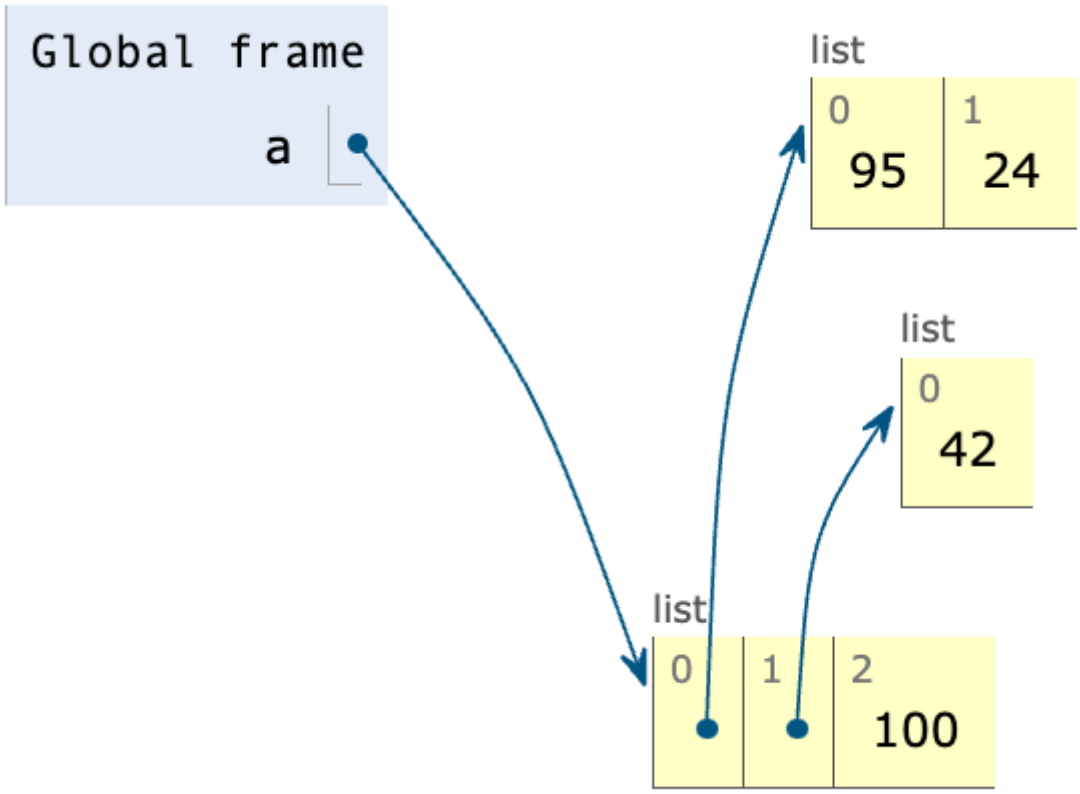
else: (else:, elif len(x) > 100:, if len(x) > 100:, if not True:)

- print("D")

1 poeng per riktig svar, 0 poeng for ubesvart eller feil svar.

Maks poeng: 4

7



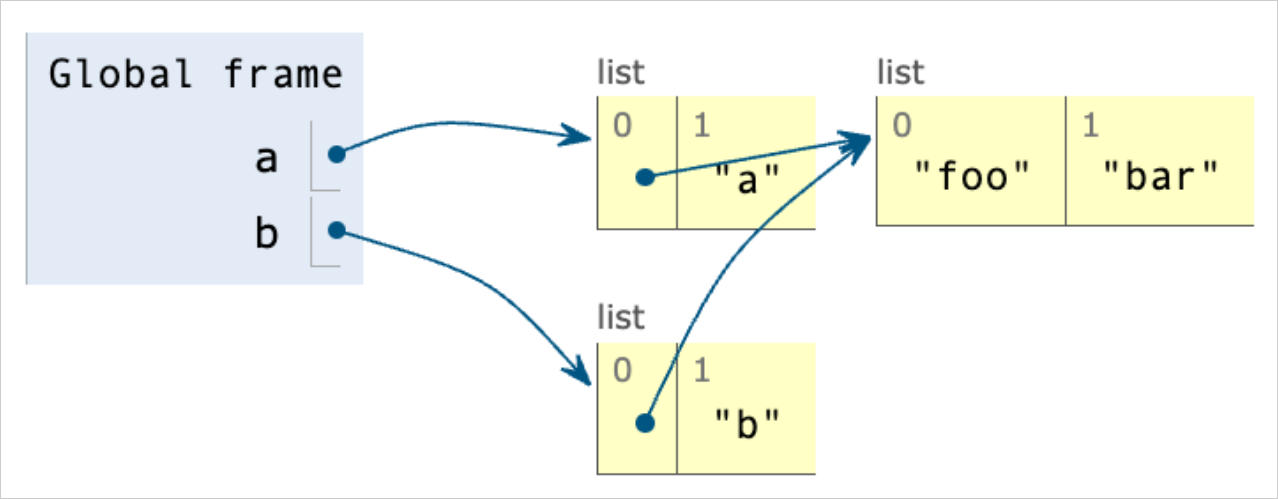
Dersom variabelen a har tilstanden som vist over, hva blir skrevet ut etter setningen **print(a[-3])**?

Velg ett alternativ:

- 24
- 42
- 95
- 100
- [24]
- [42]
- [95]
- [100]
- [95, 24]
- [[95, 24]]
- [[95, 24], 42]
- [[95, 24], [42]]
- [[95, 24], [42], 100]
- Ingen utskrift, det krasjer

Maks poeng: 2

8



Opprett to variabler a og b slik at minnets tilstand blir som vist på bildet over (bilde er hentet fra https://pythontutor.com/)

Skriv ditt svar her

```
a = [["foo","bar"],"a"]
b = [a[0],"b"]
```

Maks poeng: 5

9 Gitt at **a** er en liste med strenger, skriv en funksjon **print_errors_and_warnings(a)** som skriver ut alle strengene i **a** som begynner med "Error" eller "Warning".

For eksempel et kall

print_errors_and_warnings([
"Info: had a good nights sleep before exam",
"Warning: not enough snacks brought for exam",
"Info: went for a walk before exam started",
"Error: went to wrong exam location",
"Debug: laptop_charge=92%, charing_cable_present=True",
"Warning: time management is important",
])

skal skrive ut:

Warning: not enough snacks brought for exam
Error: went to wrong exam location
Warning: time management is important

Skriv ditt svar her

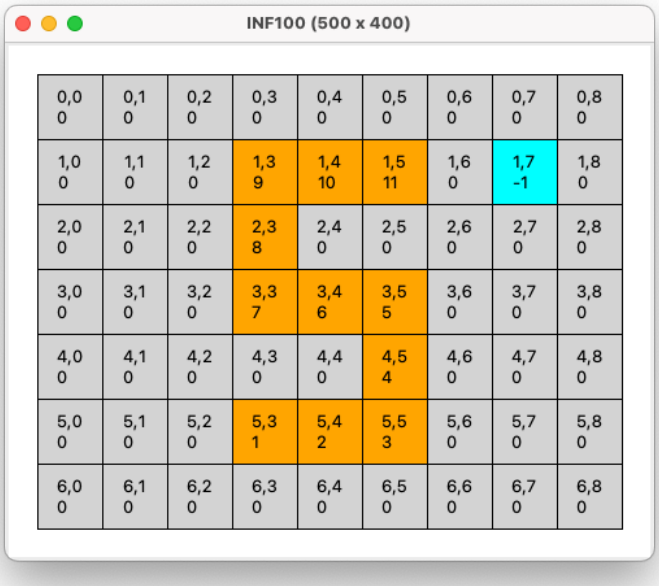
```
def print_errors_and_warnings(a):
    err_and_war_string = ""
    for line in a:
        if line.startswith("Warning") or line.startswith("Error"):
            err_and_war_string += f"{line}\n"
    print(err_and_war_string.rstrip())


a = [
    "Info: had a good nights sleep before exam",
    "Warning: not enough snacks brought for exam",
    "Info: went for a walk before exam started",
    "Error: went to wrong exam location",
    "Debug: laptop_charge=92%, charing_cable_present=True",
    "Warning: time management is important",
]


print_errors_and_warnings(a)
```

Maks poeng: 10

10



- Forklar forskjellen på en for-løkke over indekser i en liste og en for-løkke over elementer i en liste. Gi et eksempel på hvorfor man noen ganger ønsker å bruke en løkke over indeksene i listen fremfor en løkke over elementene i listen.
- Vis til eksempler på begge deler i det vedlagte løsningsforslaget til lab8: snake.py (hint: søk etter kodeordet "for")

Ca tre avsnitt, helst ikke mye mer enn 300 ord.

Skriv ditt svar her

---hentet fra mitt svar på "quizen" 'Forklaringer og refleksjoner'---

Indeks er posisjonen til et gitt element i en list eller ett symbol i en streng e.l.
Mens elementene i listen referer til den gitte "verdien" eller "innholdet" i listen.
Forskjellen er liten, sett bort fra litt ulik syntaks/semantikk, så har man litt mer kontroll over indekserte-lister.
Et eksempel kan være å returner plassen til et gitt element i en liste:

```
color_list = ["blue" , "red" , "green", "yellow"]
for i in range(len(color_list)):
    if color_list[i] == "green":
        print("Green is at index:" , i)
```

--> Green is at index: 2

Eksempel på funksjon som referer til elementene i liste:

```
def get_max_value_in_2dlist(a):
    max_value = 0
    for row in a: #sjekker elementet i listen
        for value in row:
            max_value = max(max_value, value)
    return max_value
```

Eksempel på funksjon som referer til indeksen i liste:

```
def position_of_value_in_2dlist(list2d, value):
```

```
    for row in range(len(list2d)): #indeks row mens den er mindre enn lengden av listen
        for col in range(len(list2d[row])):
            if list2d[row][col] == value:
                return (row, col)
```

Ord: 167

Maks poeng: 10

11

```
def count_a(s):
    for c in range(len(s)):
        count = 0
        if c == "a":
            count += 1
        return count
```

Koden over skal telle hvor mange ganger tegnet "a" opptrer i en streng **s**, men gir feil svar. Forklar hva som er feil, og hva man kan gjør for å fikse funksjonen.

Ca to-tre avsnitt, helst ikke mye mer enn 200 ord.

Skriv ditt svar her

I linje 2 er c definert som en indeks, altså heltall fra 0 opptil lengden av strenges som "testes", slik at på linje 4 så sjekker man om disse tallene er lik bokstaven "a", som det såklart ikke vil være. Vi kan fikse dette med å skrive linje 2 om slik: for c in s.

Vi møter nå et nytt problem, siden telleren på linje 3 settes til 0 hver gang løkken kjøres, slik at den er ubrukelig. Vi kan fikse dette med å flytte den opp/ut av løkken.

Vi møter et siste problem på siste linje, siden return er plassert med feil innrykk og vil returnere telleren etter første iterasjon. Dette fikser vi med å flytte return et innrykk mot venstre. Vi ender da opp med denne koden:

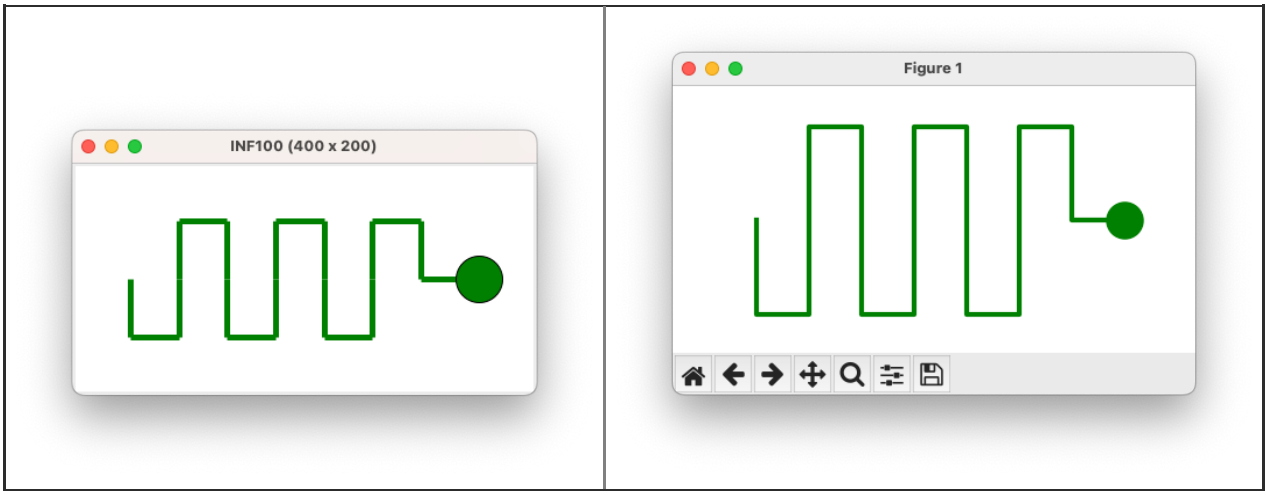
```
def count_a(s):
    count = 0
    for c in s:
        if c == "a":
            count += 1
    return count


s = "Dette er en lang setning for å teste om koden funker som tiltenkt med fire aaaer"
x = count_a(s)
print(x)
```

Ord: 169

Maks poeng: 10

12



Bruk uib_inf100_graphics -rammeverket eller matplotlib og lag et program som tegner en av figurene over.

For å få fulle poeng, må koden som tegner slangen

- benytte løkker for å skape repeterende mønstre, og
- være dynamisk, slik at antall "svinger" på slangen kan endres ved å endre på én variabel i koden.

Man kan få opptil 60% uttelling på oppgaven dersom man "hardkoder" slangen inn i bildet uten bruk av løkker. Det er ikke nødvendig at plassering eller størrelser blir nøyaktig like, så lenge mønsteret er riktig.

Skriv ditt svar her

```
from uib_inf100_graphics import *


def redraw_all(app, canvas):
    # ---variabler---
    mellom = 100
    antall = 6  # kun partall > 0

    # ---korte vertikale linjer---
    canvas.create_rectangle(mellom, 400, mellom + 20, 700, fill="green", width=0)
    canvas.create_rectangle(
        mellom * (antall + 1),
        100,
        mellom * (antall + 1) + 20,
        420,
        fill="green",
        width=0,
    )

    # ---lange vertikale linjer---
    for i in range(2, antall + 1):
        canvas.create_rectangle(
            mellom * i, 100, mellom * i + 20, 700, fill="green", width=0
        )

    # ---horisontale linjer, topp og bunn---
    for i in range(1, antall, 2):
        canvas.create_rectangle(
            mellom * i + 20, 680, mellom * (i + 1) + 20, 700, fill="green", width=0
        )
        canvas.create_rectangle(
            mellom * (i + 1) + 20,
```

```
            100,
            mellom * (i + 2) + 20,
            120,
            fill="green",
            width=0,
        )

    # ---kort horisontale linje på slutten---
    canvas.create_rectangle(
        mellom * (antall + 1) + 20,
        400,
        mellom * (antall + 2) + 20,
        420,
        fill="green",
        width=0,
    )

    # ---sirkel---/---hode til slangen---
    cx, cy = (
        mellom * (antall + 2),
        410,
    )
    r = 50
    canvas.create_oval(cx - r, cy - r, cx + r, cy + r, fill="green", width=0)


run_app(width=1000, height=800)
```

Maks poeng: 10

13 Helsesekretæren Frode har fått i oppgave å lese journal-notater som legen har skrevet om pasientene sine og registrere pasientens medisinbruk i et regneark. På bakgrunn av dette har han laget en CSV-fil med fire kolonner: personnummer, medisinnavn, startdato og sluttdato. En rad i regnearket inneholder informasjon om at en gitt pasient brukte en gitt medisin fra og med startdato, til og med sluttdato. Frode bruker komma som skilletegn i CSV-filen, og ingen celler har innhold som inneholder komma eller hermetegn. Eksempel på innhold i filen Frode lager:

personnummer,medisin,startdato,sluttdato
01010111111,vitaminbjørner,2022-08-22,2022-11-28
22020222222,fluortabletter,2022-01-11,2022-03-03
01010111111,tran,2022-06-01,2022-06-30

Sjefen til Frode skal gjøre en statistisk analyse, og er ikke fornøyd. Sjefen mener at CSV-filen burde vært presentert på et helt annet format med fire *andre* kolonner: nemlig personnummer, medisin, dato, og *endring*; der **1** i endrings-kolonnen betyr at pasienten *begynner* å ta medisinen på den gitte datoen, og **0** betyr at pasienten *slutter* å ta medisinen på denne datoen. For eksempel skulle det samme datasettet som over blitt presentert på denne måten:

personnummer,medisin,dato,endring
01010111111,vitaminbjørner,2022-08-22,1
01010111111,vitaminbjørner,2022-11-28,0
22020222222,fluortabletter,2022-01-11,1
22020222222,fluortabletter,2022-03-03,0
01010111111,tran,2022-06-01,1
01010111111,tran,2022-06-30,0

Legg merke til at én rad i det opprinnelige regnearket tilsvarer to rader i det nye.

Grunnen til at sjefen vil ha CSV-filen på denne formen er fordi hun skal gjøre en statistisk analyse, og det statistiske verktøyet hennes krever at data kommer i dette formatet. Hun har nå hyret deg som konsulent, og er villig til å betale deg 15 000 kroner for et program som omgjør fra det første formatet til det andre (dette er uansett billigere enn å la Frode gjøre jobben manuelt).

Skriv et program som leser inn en fil på det første formatet (som Frode har laget) og produserer en fil på det andre formatet (som sjefen vil ha).

Merknader

- Det originale datasettet inneholder sensitive personopplysninger, så du får ikke lov til å se det. Du får utlevert en eksempelfil sample_input.csv og expected_output.csv som du kan teste med, men poenget er selvfølgelig at programmet skal virke på det originale datasettet.
- Hverken Frode eller sjefen hans kan programmere, så du må inkludere nøyaktige instruksjoner som forteller dem hvordan de kan kjøre programmet. Legg instruksjonene inn som kommentarer øverst i programmet (dette er verdt opp til 2 poeng).
- På grunn av strenge regler for hvilken software som kan installeres på Helse Vest sine datamaskiner, kan du ikke anta at noen tredjepartsmoduler (som f. eks. pandas) er tilgjengelige. Koden din må derfor klare seg med det som finnes i Python sitt standardbibliotek.

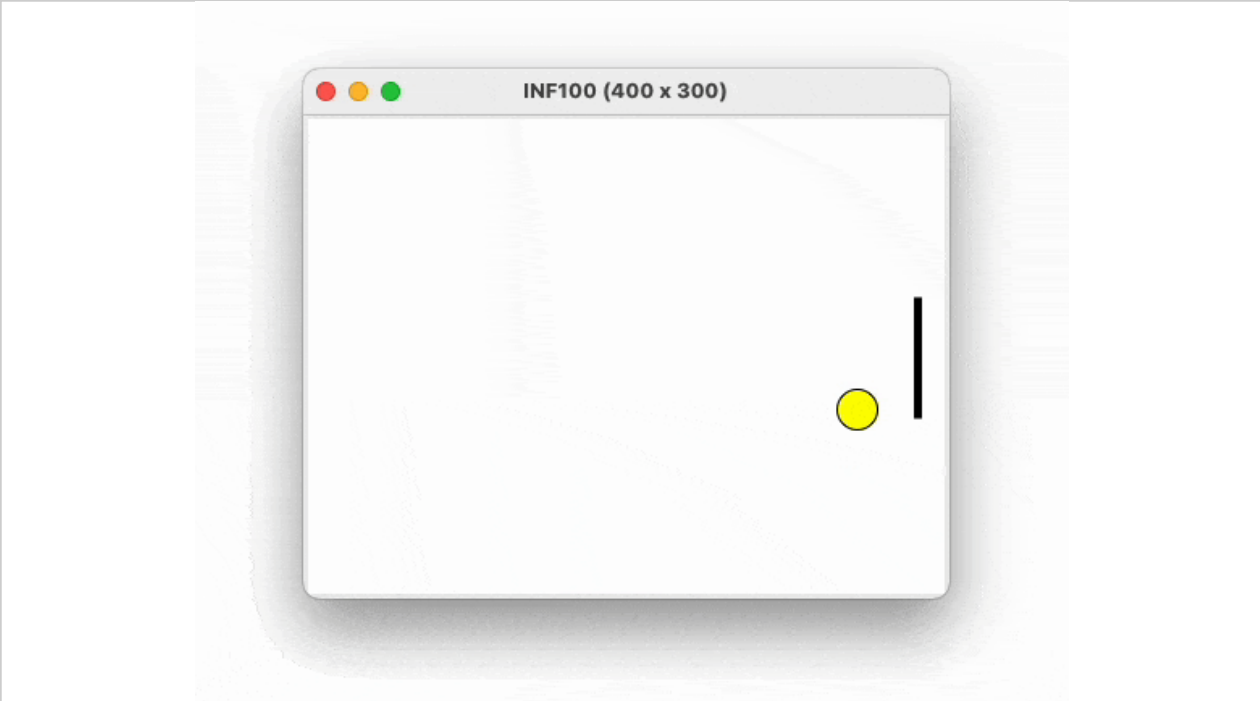
Skriv ditt svar her

```
1  from csv import reader
2
3  input_file = "sample_input.csv"  # ENDRE DENNE TIL 'RIKTIG FILSTI'/'ØNSKET FIL' SOM
```

```
4  output_file = "test_output.csv"  # ENDRE DENNE TIL 'RIKTIG FILSTI'/'ØNSKET FIL' SOM
5
6
7  output_data = f"personnummer,medisin,dato,endring\n"
8  with open(input_file, "rt") as file:
9      input_data = list(reader(file))
10     del input_data[0]
11     for line in input_data:
12         person_nr = line[0]
13         medsin = line[1]
14         startdato = line[2]
15         sluttdato = line[3]
16         output_data += f"{person_nr},{medsin},{startdato},1\n"
17         output_data += f"{person_nr},{medsin},{sluttdato},0\n"
18
19 with open(output_file, "wt") as file:
20     file.write(output_data)
21
```

Maks poeng: 10

14



I denne oppgaven skal du lage spillet "pong" for én person, som vist over.

Du skal ta utgangspunkt i og modifisere koden for en sprettende figur i kursnotatene.

Hint:

- Opprett en racket (8 poeng)
  - Modellen
    - Ha variabler for racketen sin posisjon og racketen sin størrelse i app. For eksempel variabler racket_x og racket_y for punktet midt på racketen, samt racket_height for racketen sin størrelse.
  - Kontrollen
    - Endre på racketen sin y-posisjon i key_pressed (eller i mouse_moved(canvas, app) hvis du ønsker det i stedet) (du skal ikke endre på x-posisjonen til racket).
      - Racket skal ikke kunne gå utenfor skjermen på toppen eller på bunnen
  - Visningen
    - Tegn racketen basert på posisjon og størrelse definert i app. Kan du regne ut y_top og y_bottom på bakgrunn av racket_y og racket_height?
- Dersom ballen går utenfor skjermen bak racketen skal den ikke sprette, men spillet går over i "game over" -modus. (4 poeng)
- La ballen sprette når den treffer racketen (3 poeng).
  - Hint: ikke ha for mye på én linje; opprett heller en hjelpefunksjon og/eller hjelpevariabler som brukes for å sjekke om ballen treffer racketen.

Skriv ditt svar her

```
from uib_inf100_graphics import *


def app_started(app):
    app.square_left = app.width // 2
    app.square_top = app.height // 2
    app.square_size = 50
    app.dx = -4
```

```
    app.dy = 5
    app.is_paused = False
    app.timer_delay = 30  # millisekunder
    app.racket_y = 150
    app.racket_size = 100
    app.step = 10
    app.state = "active"
    app.score = 0


def key_pressed(app, event):
    if app.state == "active":
        step = app.step
        lowest = 0
        heighest = app.height - app.racket_size
        if event.key == "p":
            app.is_paused = not app.is_paused
        if event.key == "Up" and app.racket_y > lowest:
            app.racket_y -= step
        if event.key == "Down" and app.racket_y < heighest:
            app.racket_y += step
        elif event.key == "s":
            do_step(app)
    if app.state == "gameover":
        if event.key == "r":
            run_app(width=800, height=400)


def timer_fired(app):
    if not app.is_paused:
        do_step(app)


def do_step(app):
    if app.state == "active":
        # Flytt horisontalt
        app.square_left += app.dx

        # Sjekk om firkanten har gått utenfor lerretet, og hvis ja, snu
        # retning; men flytt også firkanten til kanten (i stedet for å gå
        # forbi). Merk: det finnes andre, mer sofistikerte måter å håndtere
        # at rektangelet går forbi kanten...
        if app.square_left < 0:
            # snu retningen!
            app.square_left = 0
            app.dx = -app.dx
        elif app.square_left > app.width - app.square_size:
            app.square_left = app.width - app.square_size
            app.dx = -app.dx

        # Flytt vertikalt på samme måte
        app.square_top += app.dy
        if app.square_top < 0:
            # snu retningen!
            app.square_top = 0
            app.dy = -app.dy
        elif app.square_top > app.height - app.square_size:
            app.square_top = app.height - app.square_size
            app.dy = -app.dy

        if (app.square_left + app.square_size) >= (app.width - 5):
            if app.square_top > (app.racket_y + app.racket_size):
                app.state = "gameover"
            if (app.square_top + app.square_size) < app.racket_y:
                app.state = "gameover"
            else:
                app.score += 1
                if abs(app.dx) < 30:
                    app.dx -= 10  # sprett - sett i hermetegn for normal oppførsel
```

```
                    app.dx = 10  # Spiller, sett i hjørnetegn for normal oppførsel,
                        med "la ballen sprette", den spretter jo uansett?
78                  # kunne ha lagt til att den spretter når den treffer racket-en
                        fast hastighet etterhvert, men ser ikke poenget
79
80
81  def redraw_all(app, canvas):
82      # tegn firkanten
83      canvas.create_rectangle(
84          app.square_left,
85          app.square_top,
86          app.square_left + app.square_size,
87          app.square_top + app.square_size,
88          fill="yellow",
89      )
90      canvas.create_rectangle(
91          app.width - 10,
92          app.racket_y,
93          app.width,
94          app.racket_y + app.racket_size,
95          fill="black",
96      )
97      if app.state == "active":
98          # tegn teksten
99          canvas.create_text(
100             app.width / 2,
101             20,
102             text="Use ARROWS 'UP' and 'DOWN' to play",
103         )
104     canvas.create_text(
105         app.width / 2,
106         40,
107         text=f"Score: {app.score}",
108     )
109     if app.state == "gameover":
110         canvas.create_text(
111             app.width / 2,
112             app.height / 2,
113             text="Game Over!",
114             font="Times 50 bold",
115         )
116         canvas.create_text(
117             app.width / 2,
118             app.height / 2 + 50,
119             text="Press 'R' to restart",
120         )
121
122
123 run_app(width=800, height=400)
124
```

Maks poeng: 15