Eksamen 2018

1.0 Følg pseudokoden (30 p, 2 deloppgaver)

Du blir bedt om å lage et program som samler informasjon om eierne av sykler i et register. Dette programmet skal lagre informasjon fra bruker om navn og adresse for eierne av spesifikke sykler, og man skal kunne søke etter eierinformasjon ut fra rammenummer. Skriv programmet etter pseudokoden under.

In [ ]:

```
# Del 1:
# Opprett en tom dictionary med navn "sykler"
# Lagre listen "rammenr" under som en variabel
# For hvert element i listen "rammenr", gjør følgende:
#    - Ta inn et navn fra bruker (navnet på eieren av hver sykkel)
#    - Ta inn en adresse fra bruker (adressen til eieren av hver sykkel)
#    - Legg "rammenummer" som key, og navnet og adressen du tok inn fra bruker som values,
```

In [ ]:

```
# Del 2:
# Definer funksjonen "sok_register", som skal ta inn argumentet "ramme" og gjøre følgende:
#    - Søke i dictionaryen "sykler" etter rammenummeret som blir gitt som argument
#    - Returnere resultatet av søket

# Definer funksjonen "main" som skal:
#    - Be om input fra bruker om hvilket rammenummer han/hun vil søke opp eiers informasjon
#    - Kalle funksjonen "sok_register" og legge resultatet i en variabel med navn
#    - Printe ut resultatet av søket til konsoll

# Funksjonen "main" skal kalles hvis brukeren ønsker å søke på flere rammenummer.
```

In [ ]:

```
rammenr = WBK0132K, VE01562512D, S5A001234, WN17632Z, TA78317B, ME7265T
```

In [ ]:

```
## Del 1:

sykler = {}
rammenr = ['WBK0132K', 'VE01562512D', 'S5A001234', 'WN17632Z', 'TA78317B', 'ME7265T']

for ramme in rammenr:
    navn = str(input('Eier: '))
    adr = str(input('Adresse: '))

    sykler[ramme] = navn, adr
```

In [ ]:

```
# Del 2:

def sok_register(ramme):
    navn = sykler.get(ramme)[0]   ## Henter informasjonen om eieren av som er gitt som argu
    adresse = sykler.get(ramme)[1]
    return navn, adresse

def main():
    valg = str(input('Rammenr: '))
    res_navn, res_adr = sok_register(valg)
    print('Eieren av ramme nr', valg, 'er:', res_navn,', med adresse:', res_adr)

flere = 'ja'

while flere.lower() == 'ja':

    main()

    flere = str(input('Vil du søke etter en ny sykkel? (ja/nei)'))
```

2.0 Kodeforståelse (20 p, 2 oppgaver)

2.1 I koden under er det *to* feil i "main()" som gjør at resultatet av kjøringen ikke blir riktig. Forklar hva feilene er og hvorfor de gjør at kjøringen blir feil. (5p)

In [ ]:

```
# Dette programmet bruker en dictionary over planeter med deres respektive avstand (i milli
# Programmet skal:
# Ved hjelp av funksjoner:
# - Sjekke avstanden til sola for hver planet
# - Hente de fire planetene som er nærmest sola i en egen liste
# - Hente de fire planetene som er lengst fra sola i en egen liste
# Print listen over de nærmeste planetene til konsoll
# Print listen over de planetene lengst unna til konsoll


planeter = {'uranus': 2871,'venus':108,'jupiter':778,'jorda':149,'mars':227,'merkur':57,'ne

# Hent og sorter avstandene
def get_dist(planeter):
    avst = []
    for key,value in planeter.items(): # Iterer over keys og values i dictionaryen
        avst.append(value)

    avstSort = sorted(avst) # Sorter verdiene

    # Bruk den sorterte lista for å finne de fire nærmeste og fire lengst unna
    kortAvst = avstSort[:4]
    langAvst = avstSort[4:]

    return kortAvst,langAvst

# Ut fra listener med avstander, hent navnet på planetene som er nærmest og lengst unna
def get_names(kortAvst,langAvst):
    kortAvstNavn = []
    langAvstNavn = []

    for key,value in planeter.items(): # Iterer over keys og values i dictionaryen
        if value in kortAvst: # Sjekk om value'en eksisterer i listen over de korteste avst
            kortAvstNavn.append(key) # Legg key'en til listen
        else:
            langAvstNavn.append(key) # Alle value'ene som ikke er i lista over de korteste

    return kortAvstNavn, langAvstNavn

def main():
    korteAvst, lengstAvst = get_dist()  # Hent sorterte lister avstander fra dict'en "plane

    kortAvst, langAvst = get_names(lengstAvst,korteAvst) # Ut fra listene over, hent navner

    # Print de to listene med navn på planetene til konsoll
    print('Planetene nærmest sola er: ',kortAvst)
    print('Planetene lengst fra sola er: ',langAvst)

main() # Kall main for å kjøre programmet
```

2.1 Svar:

Den første feilen er at kallet til get_dist i main-funksjonen ikke gir noe parameter til funksjonen når den kalles. Det riktige kallet skal være "get_dist(planeter)", og feilen gjør at programmet stopper ved denne kodelinjen. Den andre feilen er at rekkefølgen på argumentene er feil i kallet til get_names() i main. Det gjør at verdiene som

lagres i variablene kortAvst og langAvst blir feil, da verdien som gis til variablene kommer fra rekkefølgen på parametrene returneres i i get_names()-funksjonen. Riktig kall skal være "get_names(kort,lang)"

2.2 Resultat av kjøringen (5p)

Hva er resultatet av kjøringen av koden under gitt x? Forklar kort hva koden gjør.

x = ['p','g','e','j','p','s','p','v','e','p','r','m','k','t','a','b','k','t','e']

bak = " "

for i in range(0,len(x),2):

    bak += x[i]

print(bak)

In [ ]:

```
x = ['p','g','e','j','p','s','p','v','e','p','r','m','k','t','a','b','k','t','e']

bak = " "

for i in range(0,len(x),2):

    bak += x[i]

print(bak)
```

2.2 Svar:

Resultatet av koden er: Pepperkake. Koden lagrer først listen over bokstaver som en variabel x. Så opprettes en tom tekst-variabel "bak". En for-løkke går så gjennom hvert 2. element i listen x og legger det til variabelen "bak". Så printes "bak" ut.

3.0 Programmering (50 p)

3.1 Programmering (15 p)

Dette programmet skal gi brukeren informasjon om hvorvidt en bil holdt en gjennomsnittsfart som var høyere enn tillatt (80 km/t) på en strekning på 3500 meter.

Programmet skal gjøre følgende:

- Lag en funksjon "speedFromTime" som tar inn to argumenter: distanse og tid. Funksjonen skal beregne og returnere hastighet (m/s) ut fra antall sekunder og distanse. Hastighet = distanse/tid.
- Lag en funksjon "speedCheck" som tar inn hastighet som argument. Denne skal sjekke om hastigheten var over 80 km/t eller ikke, og returnere True eller False basert på resultatet. Husk at du må gjøre om resultatet fra speedFromTime fra m/s til km/t.
- Lag en funksjon "main" som:

1) Henter inn informasjon fra bruker om registreringsnummer og tiden (antall sekunder) den gjeldende bilen brukte på å kjøre strekningen.

2) Kaller funksjonen "speedFromTime"

3) Kaller funksjonen "speedCheck"

4) Hvis hastigheten var over 80 km/t skal det skrives en setning til bruker om registreringsnummer og hastighet

- Så lenge bruker ønsker å skrive inn ny informasjon skal "main" kalles

In [ ]:

```
### 3.1 Svar

# Definerer funksjonen speedFromTime for å beregne hastighet i m/s
def speedFromTime(distanse, tid):
    hastighet = distanse/tid
    return hastighet

# Definerer funksjonen speedCheck for sjekk av hastigheten
def speedCheck(hastighet):
    if hastighet * 3.6 > 80:
        resultat = True
    else:
        resultat = False
    return resultat

# Definerer funksjonen main som tar inn informasjon fra bruker.
def main():
    regnr = str(input('Registeringsnummer: '))
    tid = int(input('Tid (sekunder): '))
    hastighet = speedFromTime(3500, tid)
    over = speedCheck(hastighet)
    if over:
        print('OVER FARSTGRENSEN! Bil med registreringsnummer', regnr, 'kjørte i,', hastigh
    else:
        print('Bil med registreringsnummer', regnr, 'kjørte i ', hastighet*3.6,'km/t.')

# Kontroller om bruker vil sjekke flere hastigheter
fortsett = 'J'

while fortsett.upper() == 'J':
    main()
    fortsett = str(input('Flere sjekker? J/N'))
```

3.2 Programmering (15 p)

Dette programmet skal generere en liste på 100 tilfeldige tall mellom 3-1000. Print lista ut til konsoll. Lag en funksjon som sjekker om tallene i lista er primtall (altså kun er delelig med 1 og seg selv), og legger primtallene i en ny liste. Print ut lista over primtall til konsoll.

Her er en funksjon som sjekker om tall i en liste er primtall:

def is_prime(n):

```
    for i in range(3, n):

        if n % i == 0:

            return False

    return True
```

In [5]:

```
## Oppgave 3.2 svar
import random

# Genererer liste på 100 unike tilfeldige tall mellom 3 og 1000
tilfeldige = random.sample(range(3,1000), 100)
print('100 tilfeldige tall: ',tilfeldige)

# Funksjon som sjekker som et tall er primtall
def is_prime(n):
    for i in range(3, n):
        if n % i == 0:
            return False
    return True

# Funksjon som sjekker om tallene i ei liste er primtall, og legger primtallene i en ny lis

def get_listePrimtall(liste):
    primListe = []
    for tall in liste:
        if is_prime(tall):
            primListe.append(tall)
    return primListe

# kaller funksjonen listePrim og skriver resultatet til konsoll
primtall = get_listePrimtall(tilfeldige)
print('Primtallene i listen over er: ', primtall)
```

```
100 tilfeldige tall:  [73, 635, 690, 765, 308, 130, 372, 125, 612, 613, 791,
965, 178, 759, 121, 352, 143, 452, 619, 58, 946, 592, 197, 924, 541, 11, 25
4, 122, 241, 154, 71, 26, 86, 871, 906, 85, 261, 603, 823, 515, 429, 70, 41
2, 679, 537, 983, 517, 659, 994, 573, 152, 745, 137, 641, 52, 740, 54, 828,
580, 109, 836, 210, 126, 682, 147, 405, 648, 467, 93, 235, 177, 223, 135, 11
3, 218, 240, 897, 380, 432, 883, 850, 344, 571, 732, 591, 772, 788, 601, 22
1, 6, 621, 317, 426, 631, 937, 199, 620, 629, 746, 431]
Primtallene i listen over er:  [73, 613, 619, 197, 541, 11, 241, 71, 823, 98
3, 659, 137, 641, 109, 467, 223, 113, 883, 571, 601, 317, 631, 937, 199, 43
1]
```

3.3 Programmering (20 p)

Stave navn baklengs

Del 1 (10p):

Lag en funksjon som tar inn et navn. Funksjonen skal snu rekkefølgen på bokstavene i navnet, og returnere navnet stavet baklengs. La brukeren skrive inn et navn som skal sendes til funksjonen.

In [ ]:

```
def baklengsNavn(navn):
    return navn[::-1]

navn = str(input('Skriv inn et navn: '))
navnBaklengs =  baklengsNavn(navn)

print(navn.lower(),'baklengs er: ', navnBaklengs.lower())
```

Del 2 (10p):

Denne delen av programmet skal bruke koden fra del 1. Nå skal du skrive en funksjon som lar brukeren skrive inn navnet fra del 1 i koden baklengs, og så sjekker om rekkefølgen på bokstavene er riktig sammenlignet med resultatet fra funksjonen fra del 1. Hvis hele navnet blir stavet riktig baklengs skal det printes ut en beskjed om det til bruker, men hvis navnet er feil stavet så skal han/hun få mulighet til å prøve på nytt.

In [ ]:

```
def baklengsNavn(navn):
    return navn[::-1]

def gjettBaklengs(navn):
    riktig = False
    while not riktig:
        gjettNavn = input('Hva blir navnet du skrev inn baklengs?')
        if gjettNavn.lower() == baklengsNavn(navn).lower():
            riktig = True
            print('Riktig!')
        else:
            print('Feil! Prøv igjen:')
            riktig = False

def main():
    navn = input('Skriv inn et navn: ')
    gjettBaklengs(navn)

main()
```