INF1001
Prøveksamen 2016 - Løsningsforslag

Oppgave 1.1

Hva er verdien til tall etter at følgende kode er utført?

```
tall = (5*5)-4
tall = tall-1
```

Svar:
20

Oppgave 1.2

Anta at følgende programsetninger utføres. Hva skrives ut på skjermen?

```
a = 10
b = 1
i = b
while i<a:
    b = b+i
    i=i+2
print(b)
```

Svar:
26

Oppgave 1.3

Hva skrives ut her?

```
serie = "0"
for i in range(5,10):
    serie = serie + str(i)
print("serie=" + serie)
```

Svar:
serie=056789

Oppgave 2.1

Gitt følgende kode. Hva returneres fra funksjonskallet min_funksjon(32, 6)?

```
def min_funksjon(n, m):
    x = 0
    i=n
    while i>=0:
        x=i
        i=i-m
    return x
```

Svar:
2

Oppgave 2.2

Anta at følgende program utføres, hva skrives ut?

```
class Student:
    def __init__(self, navnet):
        self._navn = navnet
    def faaNavn(self):
        return self._navn
navnet = "Grete"
s = Student ("Ole")
p = Student ("Marit")
print(p.faaNavn() + " og " + s.faaNavn())
```

Svar:
Marit og Ole

Oppgave 3.1

Skriv binærtallet 1000 1011 som et desimaltall

Svar:
139

Oppgave 3.2

Skriv desimaltallet 39 som et heksadesimalt tall.

Svar:
27

Oppgave 3.3

Skriv summen av de to binærtallene 110 og 100 som et binærtall.

Svar:
1010

Oppgave 3.4

Skriv det heksadesimale tallet 2E som et desimaltall.

Svar:
46

Oppgave 4.1

Skriv ferdig metoden under. Metoden tar inn tre heltallsverdier som argumenter, og skal returnere det tallet som verken er størst eller minst av de tre tallene i parameterne a, b og c. Du kan anta at de tre tallene sendt inn som argumenter er ulike.

def median(a, b, c):

Svar:

```
def median(a, b, c):
    if (a < b and b < c) or (a > b and b > c):
        return b
    elif (b < a and a < c) or (b > a and a > c):
        return a
    return c
```

Oppgave 5.1

Du skal skrive en funksjon som tar en liste av heltallsverdier som parameter og som returnerer en liste av heltallsverdier. Metoden skal lage en ny liste som er dobbelt så lang som den i parameteren, og kopiere over verdiene i parameterlisten til annenhver plass (fra og med indeks 0) i den nye listen. De øvrige verdiene i den nye listen skal være 0. Til slutt skal funksjonen returnere den nye listen.

Svar:

```
def dobbelListe(liste):
    nyListe = []
    for i in liste:
        nyListe.append(i)
        nyListe.append(0)
    return nyListe
```

Oppgave 6.1

Følgende kode leser inn fra tekstfil hvor mye henholdsvis Peter og Pål har hatt i ferieutgifter. Koden kjører og gir riktig svar, men det er en del unødvendige gjentakelser. Du skal skrive en modifisert versjon av programmet som skriver ut det samme som det opprinnelige, men med mindre gjentakelser i koden. Dette skal du gjøre ved å skrive og gjøre bruk av en prosedyre (eller funksjon) som kan kalles for å erstatte det som er av felles funksjonalitet i den opprinnelige versjonen av programmet. Merk forøvrig at formålet med oppgaven utelukkende er å vise at man behersker fornuftig introdusering av prosedyrer, så det er ikke nødvendig å vurdere eventuelle andre aspekter ved oppgaven eller koden.

```
fn_peter = "Peter.txt"
tot_peter=0
for line in open(fn_peter):
    utgift_peter = int(line)
    tot_peter += utgift_peter
print("Peter har brukt: ", tot_peter)
fn_paul = "Paul.txt"
tot_paul=0
for line in open(fn_paul):
    utgift_paul = int(line)
    tot_paul += utgift_paul
print("Paul har brukt: ", tot_paul)
```

Svar:

```
def les_inn_total(filnavn):
    total = 0
    for line in open(filnavn):
        utgift = int(line)
        total += utgift
    return total

fn_peter = "Peter.txt"
tot_peter= les_inn_total(fn_peter)
print("Peter har brukt: ", tot_peter)

fn_paul = "Paul.txt"
tot_paul=les_inn_total(filnavn)
print("Paul har brukt: ", tot_paul)
```

Oppgave 7

Herr Glum lager en julekalender til barna sine hvert år. Kalenderen inneholder en gave for hver dag i desember, frem til og med julaften 24.12. Det går på omgang mellom barna hvem som får lov å åpne dagens gave – når alle har åpnet en gave hver, er det førstemann sin tur igjen. Nå ønsker herr Glum seg et program som kan hjelpe ham med holde rede på hvilke barn som får hvilke gaver, og hvor mye gavene hvert barn får, har kostet.

Programmet skal kunne lese inn data om gavene fra en fil som herr Glum oppdaterer etter hvert som han handler inn gaver i tiden før desember. Filen inneholder 24 linjer, en for hver gave: På hver linje står først navnet på gaven (en tekststreng) avsluttet med komma, deretter prisen (et flyttall). Du skal skrive et program i Python som leser inn filen med alle gavene, og deretter hjelper Herr Glum med å holde orden på gavefordelingen.

7.1

Skriv en klasse Gave med to instansvariabler som forteller hva som er i gaven (gavens navn), og hvor mye den har kostet. Klassen skal ha en konstruktør med parametere som angir verdier for instansvariablene. Foruten konstruktøren skal klassens grensesnitt omfatte tre metoder: En som returnerer gavepris; en som returnerer gavenavn; og en metode __**str**__ som returnerer gavens navn og verdi som en lesbar **String**.

Svar:

```
class Gave:
    def __init__(self, navn, pris):
        self._navn = navn
        self._pris = pris

    def hentPris(self):
        return self._pris

    def hentNavn(self):
        return self._navn

    def __str__(self):
        return self._navn + " " + str(self._pris) + " kr"
```

7.2

Klassen barn skal ha en datarepresentasjon for barnets navn, alle gavene barnet har åpnet, og totalverdien av gaver barnet har åpnet. Grensesnittet til klassen skal være en konstruktør med barnets navn som parameter, en metode som returnerer barnets navn, em netode som returnerer totalverdien av alle gaver barnet har mottatt, en metode **apneGave** som legger til en ny gave og oppdaterer totalverdien av gaver barnet har fått, og en metode **skrivBarn** som skriver ut på terminal barnets navn, en linje for hver av barnets gaver og til slutt totalverdien av barnets gaver. Skriv klassen **Barn** med alt innhold.

Svar:

```
class Barn:
    def __init__(self, navn):
        self._navn = navn
        self._totVerdi = 0
        self._gaver = []

    def hentNavn(self):
        return self._navn

    def hentTotal(self):
        return self._totVerdi

    def apneGave(self, gave):
        self._totVerdi += gave.hentPris()
        self._gaver.append(gave)

    def skrivBarn(self):
        print(self._navn + ":")
        for gave in self._gaver:
            print(gave)
        print("Total verdi: " + self._totVerdi)
```

7.3

Vi skal utvide programmet med en klasse **Julekalender** med følgende metoder i grensesnittet:

- En konstruktør med parametere for
  - Navn på alle barna i familien
  - Navn på gavefilen der informasjon om gavene ligger lagret

  Konstruktøren skal opprette objekter for alle barna i familien og opprette selve julekalenderen med gaver.
- En metode **nyDag** som åpner en ny gave og oppdaterer datastrukturen (inkludert hvilken dag og hvilket barn)
- En metode **gaveOversikt** som skriver en oversikt over barna, deres åpnede gaver og totalpris per barn på skjermen

For øvrig har klassen blant annet følgende private innhold:

- En metode **_lesGavefil** som leser inn gaver med pris fra gavefilen. Filformatet er beskrevet ovenfor.
- En liste **_kalender** med referanser til en gave for hver dag
- En liste **_apnere** med referanser til hvert av barna i familien
- Et heltall **_nesteApner** som holder rede på hvem sin tur det er til å åpne
- Et heltall **_dag** som holder rede på hvilken dag som skal åpnes neste gang

Svar:

```
class Julekalender:
    def __init__(self, barneNavn, gavefil):
        self._apnere = []
        for navn in barneNavn:
            self._apnere.append(Barn(navn))

        self._kalender = []
        self._nesteApner = 0
        self._dag = 0

        self._lesGavefil(gavefil)

    def _lesGavefil(self, filnavn):
        file = open(filnavn)
        for line in file:
            words = split(",")
            gave = Gave(words[0], float(words[1]))
            self._kalender.append(gave)

    def nyDag(self):
        if self._dag > 23:
            print("Alle gavene er åpnet!")
            return

        apner = self._apnere[self._nesteApner]
        apner.apneGave(self._kalender[self._dag])
        self._dag += 1
        self._nesteApner += 1

        if self._nesteApner >= len(self._apnere):
            self._nesteApner = 0

    def gaveOversikt(self):
        for apner in self._apnere:
            apner.skrivBarn()
```

7.4

Vi skal nå utvide klassen Julekalender med historikk fra tidligere år. I første omgang er hensikten å redusere sjansen for at et barn får samme gave flere år på rad. Du kan anta at samme type gave alltid vil ha samme navn, slik at det er lett å sammenligne.

For å representere alle gaver som har vært gitt tidligere, utvider vi klassen Julekalender med en dictionary **_historikk** der nøkkel er barnets navn, og verdien er en liste med gaver dette barnet har fått. Du kan anta at følgende setninger er lagt til konstruktøren i klassen Julekalender (du trenger ikke endre dette i besvarelsen din):

```
self._historikk = {}
self._lesHistorikk("Historikk.txt")
```

Metoden **_lesHistorikk** leser data fra en fil der hver linje representerer et barn og har formatet

barnenavn gavenavn gavenavn gavenavn gavenavn . . .

Du skal nå utvide klassen Julekalender med metoden _lesHistorikk (ikke _skrivHistorikk):

```
# leser data fra fil til instansvariabelen _historikk
def _lesHistorikk(self, histfilnavn) :

# skriver ut oppdatert historikk (inkludert årets gaver) på fil
def _skrivHistorikk (self, histfilnavn) :
```

Svar:

```
def _lesHistorikk(self, histfilnavn):
    file = open(histfilnavn)
    for line in file:
        words = line.split()
        navn = words[0]
        self._historikk[navn] = []
        for gavenavn in words[1:]:
            self._historikk[navn].append(gavenavn)
```

7.5

Metoden **avvergetLike** i klassen Julekalender kalles for å avverge at et barn får en gave det har fått i tidligere år. Den kalles hver dag før metoden apneGave. Dersom gaven og barnet som står for tur sammenfaller med noe som er gitt tidligere år, byttes gaven som står for tur med en gave for en senere dag. Hvis metoden ikke klarer å hindre at et barn får en gave det har fått før, skal den returnere **False**. Dersom barnet ikke har fått gaven som står for tur tidligere, eller du klarer å avverge det ved å bytte, skal metoden returnere **True**.

Skriv metoden **avvergetLike** i klassen **Julekalender**

```
# Prøver å avverge at et barn får en gave det har fått tidligere
# Returnerer true om den lykkes, false om den ikke lykkes

def avvergetLike (self) :
```

Svar:

```
def avergetLike(self):
    gaveFraDag = self._dag
    barnNavn = self._apnere[self._nesteApner]

    while gaveFraDag < 24:
        gaveNavn = self._kalender[gaveFraDag]

        if gaveNavn not in self._historikk[barnNavn]:
            midlertidig = self._kalender[self._dag]
            self._kalender[self._dag] = self._kalender[gaveFraDag]
            self._kalender[gaveFraDag] = midlertidig
            return True

        gaveFraDag += 1
    return False
```

Oppgave 8.1

I spillet Yatzy får man poeng for ulike kombinasjoner av verdier på fem terninger. En av kombinasjonene som gir poeng kalles "hus" og krever at tre at terningene viser en verdi (er like) og at de to resterende terningene viser en annen (lik) verdi. Altså at man blant de fem terningene har tre like og to like. Det beste huset man kan ha er tre seksere og to femmere.

Skriv en funksjon **besteHus(t)** som tar inn en liste av heltalls-verdier som parameter, og returnerer True dersom listen t bestar av tre verdier 6 og to verdier 5 (i vilkarlig rekkefølge). Ellers skal funksjonen returnere False. Du kan anta at du alltid får inn en liste av lengde 5, der hver verdi er større eller lik 1 og mindre eller lik 6.

Altså skal f.eks. følgende assert-statement ikke feile:

assert besteHus([5,6,6,5,6]) == True

Svar:

```
def besteHus(t):
    femmere = 0
    seksere = 0
    for i in t:
        if i == 5:
            femmere += 1
        elif i == 6:
            seksere += 1

    return seksere == 3 and femmere == 2
```

Oppgave 8.2

Skriv en metode hus(t) med samme parameter og returverdi som i Oppgave 8.1, men der metoden returnerer True for alle terningkombinasjoner som er hus (ikke bare hus av tre seksere og to femmere).

Svar:

```
def hus(t):
    kast = [0, 0, 0, 0, 0, 0]
    for i in t:
        kast[i-1] += 1

    if 2 in kast and 3 in kast:
        return True
    return False
```

Oppgave 9.1

Er dette en personopplysning? Begrunn med henvisning til Personopplysningsloven.

Mann, født 1990. Mangler fast bopæl. Straffedømt for narkotikabruk.

Svar:
Nei. Dette kan ikke knyttes til en enkeltperson og er derfor ikke en personopplysning i hht. §2.1.

Oppgave 9.2

Nevn tre sentrale hensyn til informasjonssikkerhet som ifølge Personopplysningsloven skal ivaretas ved behandling av personopplysninger.

Svar:
Konfidensialitet, integritet og tilgjengelighet. Se §13.

Oppgave 9.3

Som ledd i arbeidet mot forsikringssvindel har norske forsikringsselskaper fått konsesjon for lagring av følgende opplysninger om sine skadeoppgjør i et felles, sentralt skaderegister: Forsikringstakers fødselsnummer, saksnummer, bransjekode, selskap, skadetype, dato, saksbehandlers initialer og skadeland.

Konsesjonen omfatter ikke tillatelse til registrering av sensitive personopplysninger.

Diskuter om noen av opplysningene konsesjonen omfatter kan komme til å omfatte sensitive personopplysninger slik at spesiell varsomhet bør utvises ved registrering av disse i det felles registeret.

Svar:
Skadetype vil kunne omfatte sensitive personopplysninger da det kan gi opplysninger om helseforhold. Se §2.8.