INF1000 Eksamen 2014 (modifisert)

Oppgave 1 (4 poeng)

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

```
tall = (5+3)*2
tall = tall+2
```

Svar: 18.

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

```
a = "ab"
b = "b" + a
a = a + b
print(a)
```

Svar: abbab

c) Hva er verdien til x etter at følgende kode er utført?

```
x = 6
y = 15
while x < 30 :
    if x < y :
       x = x * 2
    else :
        y += 5
        x -= 5
```

Svar: 38

Oppgave 2 (6 poeng)

a) Hva skrives ut her?

```
def regnUt(a, b):
 if a == b :
    svar = a * 2
 else :
    svar = a
```

```
 return svar

print(regnUt(4, 10))
```

Svar: 4

b) Anta at vi har følgende klasse:

```
#klassen MinKlasse
class MinKlasse :

def __init__(self, startVerdi) :              self._minVerdi =
startVerdi

def leggTilVerdi(self, tillegg):       self._minVerdi += tillegg

def hentVerdi(self) :
  return self._minVerdi
```

Hva blir utskriften på skjermen når programmet nedenfor utføres?

```
#anta at dette er en annen fil som har importert MinKlasse riktig

mittObjekt = MinKlasse(5)
mittObjekt.leggTilVerdi(3)
verdi = mittObjekt.hentVerdi()
mittObjekt.leggTilVerdi(verdi)
print(mittObjekt.hentVerdi())
```

Svar: 16

c) (Vanskelig) Det er en logisk feil i funksjonen *stoerst* nedenfor, som gjør at den ikke alltid returnerer det største av de tre tallene den får som parameter. Gi et eksempel på et kall med tre tall som fører til at funksjonen returnerer et annet tall enn det største.

```
def stoerst(a, b, c) :
 if a>b and a>c :
   svar = a
   return svar
 elif b>a and b>c :
   svar = b
   return svar
 else :
   svar = c
   return svar
```

Svar: 1

~~Oppgave 3 (4 poeng)~~

Oppgave 4 (5 poeng)

Skriv en funksjon `def pris(gratis, alder)`

Dersom parameteren gratis har verdien True, skal funksjonen alltid returnere 0. Dersom parameteren gratis har verdien False og verdien av alder er mindre enn 18, skal funksjonen returnere 100, ellers 200.
Altså skal f.eks. kallet pris (True, 10) returnere 0, kallet pris(False,10) returnere 100 og kallet pris(False, 50) returnere 200.

Svar:

```
def pris(gratis, alder):
  if gratis == True:
    return 0
  elif alder < 18:
    return 100
  else:
    return 200
```

Oppgave 5 (5 poeng)

Skriv en funksjon som har en liste med tall som parameter og som returnerer en verdi av type boolean. Funksjonen skal sjekke om alle verdiene i listen er i stigende rekkefølge (sortert). Dersom alle verdiene er i sortert rekkefølge skal funksjonen returnere true, ellers skal funksjonen returnere false. Du kan anta at alle verdiene i listen er ulike. Funksjonen trenger altså ikke ta hensyn til eventuelle like verdier.

Svar:

```
def sjekk(liste):
  if len(liste) != 0:
    verdi = liste[0]
    for i in range(1, len(liste)):
      if verdi > liste[i]:
        return False
  else:
    return False

  return True
```

Oppgave 6 (6 poeng)

a) Skriv en funksjon med liste av tall som parameter og som returnerer en verdi av type int (heltall). Dersom alle verdiene i listen er like, skal metoden returnere denne verdien. Dersom ikke alle verdiene er like, skal den returnere tallet -1. Du kan anta at listen inneholder minst en verdi.

Svar:

```
def funksjon(liste):
  verdi = liste[0]

  for i in range(1, len(liste)):
    if verdi != liste[i]:
      return -1

  return verdi
```

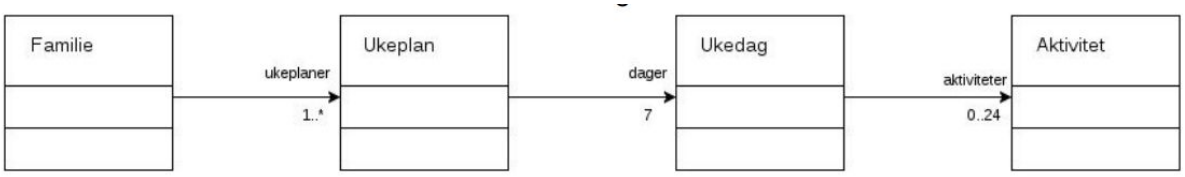
b) Dersom du kaller funksjonen fra a) med en ikke-tom liste av tall og får -1 tilbake, kan du da være sikker på at ikke alle tallene i listen du sendte inn var like? Begrunn svaret.

Svar: Nei. Dersom vi kaller funksjonen med en liste som kun inneholder tallet -1 vil vi returnere tallet -1 selv om alle tallene er like.

Oppgave 7: Familiens ukeplaner (50 poeng)

Du har påtatt deg å lage ukeplaner for hele familiens faste aktiviteter. For å gjøre det enkelt antar vi at alle aktiviteter starter på en hel time (kl 00.00, 01.00, 02.00 etc) og at de alle varer nøyaktig 1 time.

Du skal bruke klassene vist i dette UML klassediagrammet:



a) Klassen Aktivitet skal ha en representasjon med disse to objektvariablene: en String med navn aktNavn som brukes til å beskrive hva slags aktivitet det gjelder (for eksempel "svømming" eller "trener fotball 1A"), og et tall med navn start som inneholder starttidspunktet for aktiviteten.

Klassen skal også inneholde denne konstruktøren:

```
def __init__(self, hva, kl)
     # Innmat
```

Skriv klassen Aktivitet inkludert konstruktør (som må programmeres ferdig).

Svar:

```
class Aktivitet:
  def __init__(self, hva, kl):
    self._aktNavn = hva
    self._start = kl
```

b) Skriv klassen Ukedag med nødvendige objektvariable og dette foreløpige grensesnittet (dvs disse metodene):

```
#Konstruktør
def __init__(self, dag):
```

```
# Oppretter og setter inn ny aktivitet på angitt  klokkeslett, og
gir feilmelding til terminal om tidspunktet er opptatt
def settInn(self, hva, kl):

#Returnerer når på dagen den første aktiviteten starter (-1 hvis
det ikke finnes noen aktivitet denne dagen)
def tidligste(self):

# Returnerer når seneste aktivitet den dagen starter (-1 hvis
ingen aktiviteter)
def seneste(self):

# Beregner og returnerer antall aktiviteter den dagen
def antall(self):
```

c) Klassen Ukedag skal nå utvides med metoden settInnLedig som setter inn en aktivitet for en ledig time den dagen. Hvis kalenderen er full, skal metoden skrive ut en feilmelding. Hvis det ikke er noen aktiviteter fra før den dagen, skal aktiviteten starte kl 12.00. Ellers skal metoden helst sette inn den nye aktiviteten i en ledig time mellom tidligste og seneste aktivitet; hvis det ikke er mulig, skal den settes rett etter den hittil seneste aktiviteten; om heller ikke det er mulig, settes aktiviteten inn rett før den hittil tidligste. Skriv metoden `def settInnLedig(self, hva).`

Svar:

```
from aktivitet import Aktivitet

class Ukedag:
  def __init__(self, dag):
    self._dag = dag
    self._aktiviteter = []
    for i in range (0, 24):
      self._aktiviteter.append(None)

  def settInn(self, hva, kl):
    if self._aktiviteter[kl] is not None:
      print("Klokkeslettet er opptatt.")
    else:
      self._aktiviteter[kl] = Aktivitet(hva, kl)

  def tidligste(self):
    for i in range(0, 24):
      if self._aktiviteter[i] is not None:
```

```
        return i
    return -1

  def seneste(self):
    for i in range(23, -1, -1):
      if self._aktiviteter[i] is not None:
        return i
    return -1

  def antall(self):
    total = 0
    for i in range(0, 24):
      if self._aktiviteter[i] is not None:
        total += 1
    return total

def settInnLedig(self, hva):
    ingen = True
    ferdig = False
    for i in range(0, 24):
      if self._aktiviteter[i] is not None:
        ingen = False
    if ingen:
      self.settInn(hva, 12)
      ferdig = True

    if not ferdig:
      for i in range(self.tidligste() + 1, self.seneste()):
        if self._aktiviteter[i] is None:
          self.settInn(hva, i)
          ferdig = True

    if not ferdig:
      for i in range(self.seneste() + 1, 24):
        if self._aktiviteter[i] is None:
          self.settInn(hva, i)
          ferdig = True

    if not ferdig:
      for i in range(self.tidligste(), -1, -1):
        if self._aktiviteter[i] is None:
          self.settInn(hva, i)
          ferdig = True

    if not ferdig:
```

```
      print("Det er ikke plass til aktiviteten paa denne dagen!")
```

d) Skriv klassen Ukeplan med følgende grensesnitt (dvs metoder):

```
#Konstruktør
def __init__(self, hvem)

#Finner og returnerer dagen med flest aktiviteter
def travleste(self)
```

Svar:

```
class Ukeplan:
  def __init__(self, hvem):
    self._hvem = hvem
    self._dager = [ Ukedag("Mandag"), Ukedag("Tirsdag"),
Ukedag("Onsdag"),
                    Ukedag("Torsdag"), Ukedag("Fredag"),
Ukedag("Lordag"),
                    Ukedag("Sondag") ]
  def travleste(self):
    maks = 0
    for i in range(0, 7):
      if self._dager[i].antall() > self._dager[maks].antall():
        maks = i
    return self._dager[maks]
```

e) Klassen Ukeplan skal nå utvides med metoden skrivUt.

 # Metoden skal sørge for at ukeplanen skrives ut, for eksempel slik: Ukeplan for Aud:
 # Mandag Kl 9: Forelesning Kl 10: Forelesning Kl 12: Lunsj Onsdag Kl 17: Trening

Du vil sannsynligvis trenge nye metoder i flere av de andre klassene også; skriv disse metodene og angi i hvilken klasse den enkelte metoden hører hjemme. Selve formatteringen av utskriften er ikke viktig, så ikke bruk mye tid på det.

Svar:

```
# I klassen Ukeplan.
def skrivUt(self):
  print("Ukeplan for " + self._hvem + ":")
  for dag in self._dager:
```

```
    dag.skrivUt()

# I klassen Ukedag.
def skrivUt(self):
  print(self._dag + ":")
  for akt in self._aktiviteter:
    if akt is not None:
      print(" ", akt)

# I klassen Aktivitet.
def __str__(self):
  return "Kl " + str(self._start) + ": " + self._aktNavn
```

f) (Vanskelig) Skriv klassen Familie med konstruktør og nødvendige objektvariabler, i tillegg til metoden skrivAktiviteter, som skal skrive ut en liste over alle de ulike aktivitetene (fra aktNavn-variabelen) familien er med på.

Aktiviteter med samme navn (dvs har lik verdi i aktNavn-variabelen) skal kun skrives ut én gang selv om de forekommer flere ganger hos samme eller forskjellige familiemedlemmer.

Hint: Bruk en liste til til å holde orden på hvilke aktivitetsnavn som har vært skrevet ut.

Svar:

```
class Familie:
  def __init__(self):
    self._ukeplaner = []
  def skrivAktiviteter(self):
    print("Aktiviteter:")
    skrevet = {}
    for plan in self._ukeplaner:
      for dag in plan.hentUkedager():
        for akt in dag.hentAktiviteter():
          if akt is not None and akt.hentNavn() not in skrevet:
            print(akt)
            skrevet[akt.hentNavn()] = True

# I klassen Ukeplan.
def hentUkedager(self):
  return self._dager

# I klassen Ukedag.
def hentAktiviteter(self):
  return self._aktiviteter

# I klassen Aktivitet.
```

```
def hentNavn(self):
  return self._aktNavn
```

~~Oppgave 8 (10 poeng)~~

Oppgave 9 (10 poeng)

Dersom du kaster 3 terninger, er det 6*6*6=216 mulige utfall av antall øyne på de tre terningene (1-1-1, 1-1-2, …, 6-6-6). Bare i 6 av disse 216 utfallene er det samme antall øyne på alle de tre terningene (1-1-1, 2-2-2 osv). Skriv de nødvendige programlinjene for å printe ut alle kombinasjoner av antall øyne på de tre terningene på terminalen. Print til slutt ut hvor mange kombinasjoner som hadde minst 2 like terninger. Merk at 1-1-2 og 1-2-1 i denne sammenhengen er to ulike kombinasjoner, slik at begge skal printes ut og telle med i antall kombinasjoner med minst 2 like terninger.

Svar: [Mangler]