Oppgavesett for uke 8 time 1

Oppg 1

```
1  class Matbutikk
2      def __init__(self):
3          self._varer = []
```

a) Utvid klassen `Matbutikk`, slik at den får en metode for å hente lista med varer.
b) Skrive en metode for å legge til et `Vare`-objekt i lista varer.
c) Trenger vi flere parametere enn `self` i konstruktøren over? Begrunn svaret.
d) Skriv klassen `Vare`, med instansvariabler for vareid og varenavn. init-metoden (konstruktøren) tar inn `vareid` og `varenavn`. `Vare` skal ha to metoder; én for å hente vareid og én for å hente varenavn.
e) Skriv et hovedprogram med en `Matbutikk`, og opprett og legg til tre `Vare`- objekter. Skriv ut vareid sammen med varenavn for de tre varene på tre forskjellige linjer. Dere blir nødt til å kalle relevante metoder underveis.

Oppg 2

```
1   class Student:
2       def __init__(self, navn, studieprogram):
3           self._navn = navn
4           self._studieprogram = studieprogram
5
6       def hent_navn(self):
7           return self._navn
8
9       def endre_studieprogram(self, nytt_program):
10          self._studieprogram = nytt_program
```

a) Hva er det `self` refererer til i init-metoden (konstruktøren) og i metodene `hent_navn` og `endre_studieprogram`?
b) Hva slags variabler er `navn`, `studieprogram` og `nytt_program`?
c) Hvordan skiller variablene i b) seg fra variablene `_navn` og `_studieprogram`?
d) Hvorfor trenger vi `self` i metodene `hent_navn` og `endre_studieprogram`?
e) Hvorfor trenger vi ikke å inkludere `self` som argument når vi kaller metodene `hent_navn` og `endre_studieprogram` på et `Student`-objekt?

Oppg 3

Gitt hovedprogrammet nedenfor, tegn datastrukturen slik den ser ut etter linjen `forelder.få_barn(barn)`. En datastrukturtegning viser opprettede objekter og sammenhengen mellom dem. Tips: Hvordan datastrukturer tegnes i IN1000 ligger bakerst i ukas forelesningsslides, også fra i fjor. Sammenlign deres tegning med Python Tutors. (Merk at Python Tutor tegner på en litt annen måte enn vi gjør i IN1000)

```
1   class Forelder:
2       def __init__(self):
3           self._barn = None
4
5       def få_barn(self, barn):
6           self._barn = barn
7
8   class Barn:
9       def __init__(self, fødselsdato, fødselsvekt):
10          self._fødselsdato = fødselsdato
11          self._fødselsvekt = fødselsvekt
12
13  def hovedprogram():
14      forelder = Forelder()
15      barn = Barn("01012024", 3500)
16      forelder.få_barn(barn)
17
18  hovedprogram()
```

Oppg 4

```
1   class A:
2       def __init__(self, a):
3           self._a = a
4
5   class B:
6       def __init__(self, b):
7           self._b = b
8
9   class C:
10      def __init__(self, c):
11          self._c = c
12
13  def hovedprogram():
14      c = A("B")
15      a = B("C")
16      b = C("A")
17      b = a
18      a = c
19      c = b
20
21  hovedprogram()
```

a) Hva refererer henholdsvis variablene `a`, `b` og `c` til i hovedprogrammet?
b) Hvorfor bruker vi ikke self noe sted i hovedprogrammet?
c) Er argumentene vi gir, gyldige? Forklar hvorfor/hvorfor ikke.

Oppg 5

```
1   class KaffeLatte:
2       def __init__(self, ant_espressoshots):
3           self._antall_espressoshots = ant_espressoshots
4
5       def hent_antall_espressoshots(self):
6           return self._antall_espressoshots
7
8   def hovedprogram():
9       bestillinger = []
10      for i in range(5):
11          dobbel_latte = KaffeLatte(2)
12          bestillinger.append(dobbel_latte)
13
14      for bestilling in bestillinger:
15          print(bestilling)
16
17  hovedprogram()
```

a) Hva vil skrives ut i programmet? Kom opp med et svar og begrunn hvorfor **før** dere kjører programmet.

b) Endre programmet, slik at linjen `dobbel_latte = KaffeLatte(2)` står under linjen `bestillinger = []` i stedet for inni for-løkka. Hva skrives ut nå? Kom opp med et svar og begrunn hvorfor før dere kjører programmet.

c) Hvor mange objekter av klassen `KaffeLatte` finnes i hovedprogrammet i a), og hvor mange finnes det i hovedprogrammet i b)? Begrunn hvorfor.