i

IN1000-2017-Prøveeksamen

Prøveeksamen IN1000

Dato 13-20. november 2017
Tid 12:30-12:00
Alle trykte og skrevne hjelpemidler er tillatt.

Les nøye gjennom oppgavene før du løser dem.
Du kan legge dine egne forutsetninger til grunn og gjøre rimelige antagelser, så lenge de ikke bryter med oppgavens «ånd». Gjør i så fall rede for disse forutsetningene og antagelsene.
Poengangivelsen øverst i hver oppgave angir maksimalt antall poeng. Sammenlagt gir alle oppgavene maksimalt 100 poeng. Unngå å bruke en stor del av tiden din på oppgaver som gir deg få poeng.

Gjelder ikke prøveeksamen: Faglærer besøker eksamenslokalet etter ca. 1 time.

1

Oppgave 1a)

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

```
tall = 4+(3*2)
tall = tall-1
```

Maks poeng: 1

2

Oppgave 1b)

Hva er verdien til **j** etter at følgende kode er utført?

```
i = 1
j = 2
while i < 4:
  j = j * i
  i = i + 1
```

Maks poeng: 2

3 # Oppgave 1c)

Hva skrives ut på skjermen når følgende kode utføres?

```
tall = [2, 6, 3, 6, 9]
a = 0
b = 0
i = 0

while i < len(tall):
  if tall[i] > a:
    b = b + tall[i]
    a = tall[i]
  i = i + 1

print(b)
```

.

Maks poeng: 2

4 # Oppgave 1d)

Vi har en funksjon **repeter** som vist nedenfor:

```
def repeter(a):
  a = a + a
  return a
```

Hva skrives ut på skjermen når følgende kode utføres?

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

Maks poeng: 2

5 # Oppgave 1e)

Hva skrives ut på skjermen når koden i pdf-vedlegget utføres?

Maks poeng: 3

6

Oppgave 1f)

Gitt en funksjon **voks** som vist her:

```
def voks(alder):
  alder = alder +1
```

Hva skrives ut på skjermen når følgende kode utføres?

```
pers_alder = 29
voks(pers_alder)
print(pers_alder)
```

.

Maks poeng: 1

7

Oppgave 1g)

Gitt en funksjon **brillesjekk** som vist her:

```
def brillesjekk(styrke):
  ny_styrke = [2.5, 2.75]
  styrke = ny_styrke
```

Hva skrives ut på skjermen når koden nedenfor utføres?

```
pers_styrke = [1.5, 1.5]
brillesjekk(pers_styrke)
print(pers_styrke[0])
```

.

Maks poeng: 1

8

Oppgave 1h)

Gitt en funksjon **brillesjekk2** som vist her:

```
def brillesjekk2(styrke):
  styrke[0] = 1.75
```

Hva skrives ut på skjermen når følgende kode utføres?

```
pers_styrke = [1.5, 1.5]
brillesjekk2(pers_styrke)
print(pers_styrke[0])
```

.

Maks poeng: 1

9

Oppgave 1i)

Hva skrives ut når følgende programsetninger kjøres?
a = [1, 2, 3]
b = a
b[0] += 1
print(a)
Velg ett alternativ

- [1,2,3]
- [1,1,2,3]
- [2,2,3]

Maks poeng: 1

10

Oppgave 1j)

liste = [[5,4], [9,12,3]]
a) Hva er verdien av liste[1][0]?

b) Hva er verdien av liste[0]?

Maks poeng: 2

11

Oppgave 1k)

ordbok = { "b":[4,3,5], "a":[0] }
Hva er verdien av ordbok["a"][0]?

Maks poeng: 1

12

Oppgave 2a)

Hva er galt i følgende kode? (kort forklaring holder - én setning er gjerne nok)

```
def gang_med_to(tall):
    return tall*2

svar = gang_med_to(5,4)
```

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 3

13

Oppgave 2b)

Hva er galt i følgende kode? (kort forklaring holder - én setning er gjerne nok)

```
def hent_pris(alder):
    if alder<18:
        return print(100)
    else:
        return print(200)

antall = 3
pris = hent_pris(18)
totalt = antall*pris
```

Skriv ditt svar her...

Format

Words: 0

Maks poeng: 3

14 Oppgave 3a)

Skriv en funksjon **hastighet(fart)** som skal returnere en tekst-streng basert på heltallsverdien (verdi av type int) i parameteren fart. Parameteren **fart** er ment å angi den målte farten til en bil i en 60-sone. Dersom **fart** er 60 eller mindre, skal funksjonen returnere en streng (verdi av type str) som består av "fart:" og den målte farten. F.eks. skal kallet **hastighet(56)** returnere strengen "fart:56". Dersom **fart** er høyere enn 60, skal funksjonen returnere strengen "fart:over 60". Kallene **hastighet(61)** og **hastighet(100)** skal altså begge returnere strengen "fart:over 60".

```
1
```

Maks poeng: 5

15

Oppgave 3b)

a) Skriv en funksjon **sjekkVerdier(tallene, min, max)** hvor **tallene** er en liste av heltallsverdier (liste av verdier av type int), mens **min** og **max** er heltall (verdi av type int). Funksjonen skal sjekke om alle verdiene i lista **tallene** er ekte større (ikke lik) enn **min** og ekte mindre (ikke lik) enn **max**. Dersom alle verdiene er innenfor dette intervallet skal metoden returnere **True**, ellers skal metoden returnere **False**.

b) Beskriv i en kommentar nederst i programkoden hvordan metoden din oppfører seg dersom **min > max**.

```
1
```

Maks poeng: 7

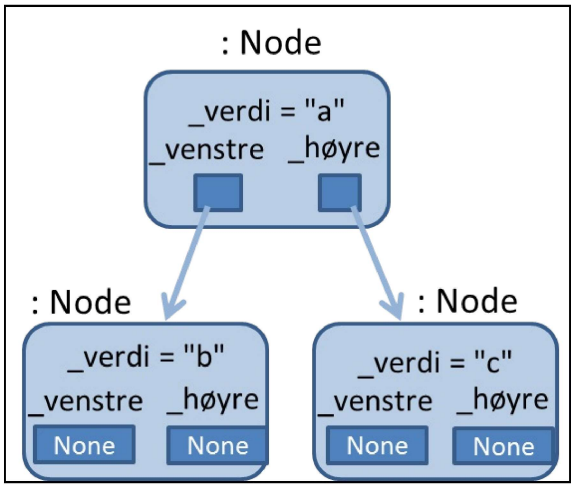
16

Oppgave 3c)

```
class Node :
    def __init__(self, innhold) :
        self._verdi = innhold
        self._venstre = None
        self._hoyre = None

    def settInnHoyre (self, h) :
        self._hoyre = h

    def settInnVenstre (self, v) :
        self._venstre = v
```



Gitt klassen Node som vist i vedlagte kode, skriv en funksjon **hovedprogram()** som oppretter 3 objekter med verdiene "a", "b" og "c" i en struktur som vist i figuren. Du kan anta at klassen Node er importert til programmet ditt.

Skriv ditt svar her...

```
1
```

Maks poeng: 7

17

Oppgave 4 a)

I oppgave 4 a-g (Klasser og objekter) skal du skrive deler av et program for et elektronisk bruktmarked. Den samlede teksten for hele oppgave 4 (a-g) er lagt ved hver deloppgave.

Oppgave 4 a

Skriv klassen Bud med alle metoder som spesifisert i vedlagt dokument.
(Senere deloppgaver ber om andre deler av koden for det elektroniske bruktmarkedet.)

```
1
```

Maks poeng: 5

18

Oppgave 4b)

(fortsettelse oppgave 4) Klasser og objekter)

Oppgave 4 b
Skriv klassen Annonse med alle metoder.

```
1
```

Maks poeng: 10

19

Oppgave 4c)

(fortsettelse oppgave Klasser og objekter)

Oppgave 4 c
Skriv klassen Kategori med alle metoder, som spesifisert i vedlagt dokument.

```
1
```

Maks poeng: 6

20

Oppgave 4d)

(fortsettelse oppgave Klasser og objekter)

Oppgave 4 d
Skriv klassen Bruktmarked med alle metoder og representasjon som spesifisert i vedlagt dokument.

```
1
```

Maks poeng: 8

21

Oppgave 4e)

(fortsettelse oppgave Klasser og objekter)

Oppgave 4 e
Skriv metoden **kraftBud** i klassen **Annonse**, som spesifisert i vedlagt dokument.

```
1
```

Maks poeng: 8

22

Oppgave 4f)

(fortsettelse oppgave Klasser og objekter)

Oppgave 4 f
Skriv et hovedprogram som bruker klassene fra tidligere deloppgaver slik det er spesifisert i vedlagt dokument.

```
1
```

Maks poeng: 8

23

Oppgave 4g)

(fortsettelse oppgave Klasser og objekter)

Oppgave 4 g
Skriv metoden **tellLaveBud** i klassen **Bruktmarked** slik det er spesifisert i vedlagt dokument.

```
1
```

Maks poeng: 5

24

Oppgave 5

1) Skriv en funksjon **arverekke(forfader, etterkommer, forstefodte)**
som kan brukes for å returnere en liste med alle navn i arverekken fra og med **forfader** (av type str) til og med **etterkommer** (av type str) dersom denne kan utledes fra mappingen **forstefodte**. Parameteren **forstefodte** kan antas å være en ordbok (dict) fra foreldre til førstefødte barn. Når man slår opp med et navn (av type str) som nøkkel, får man altså navnet (av type str) på det førstefødte barnet som verdi (dersom denne eksisterer). Personene lagret i **forstefodte** danner ikke nødvendigvis en sammenhengende arverekke.

Dersom **forfader** og **etterkommer** ikke hører sammen i en felles arverekke, skal funksjonen returnere en tom liste.

Ved bruk av funksjonen, skal følgende kodesekvens i Python resultere i at verdien til **personer** blir en liste med navnene "Halfdan","Harald","Eirik":

```
barn = {"Halfdan":"Harald", "Christian":"Hans", "Harald":"Eirik"}
personer = arverekke("Halfdan", "Eirik", barn)
```

2) Beskriv, i en kommentar i funksjonen, hvordan funksjonen din oppfører seg dersom flere av personene i arverekken har nøyaktig samme navn.

```
1
```

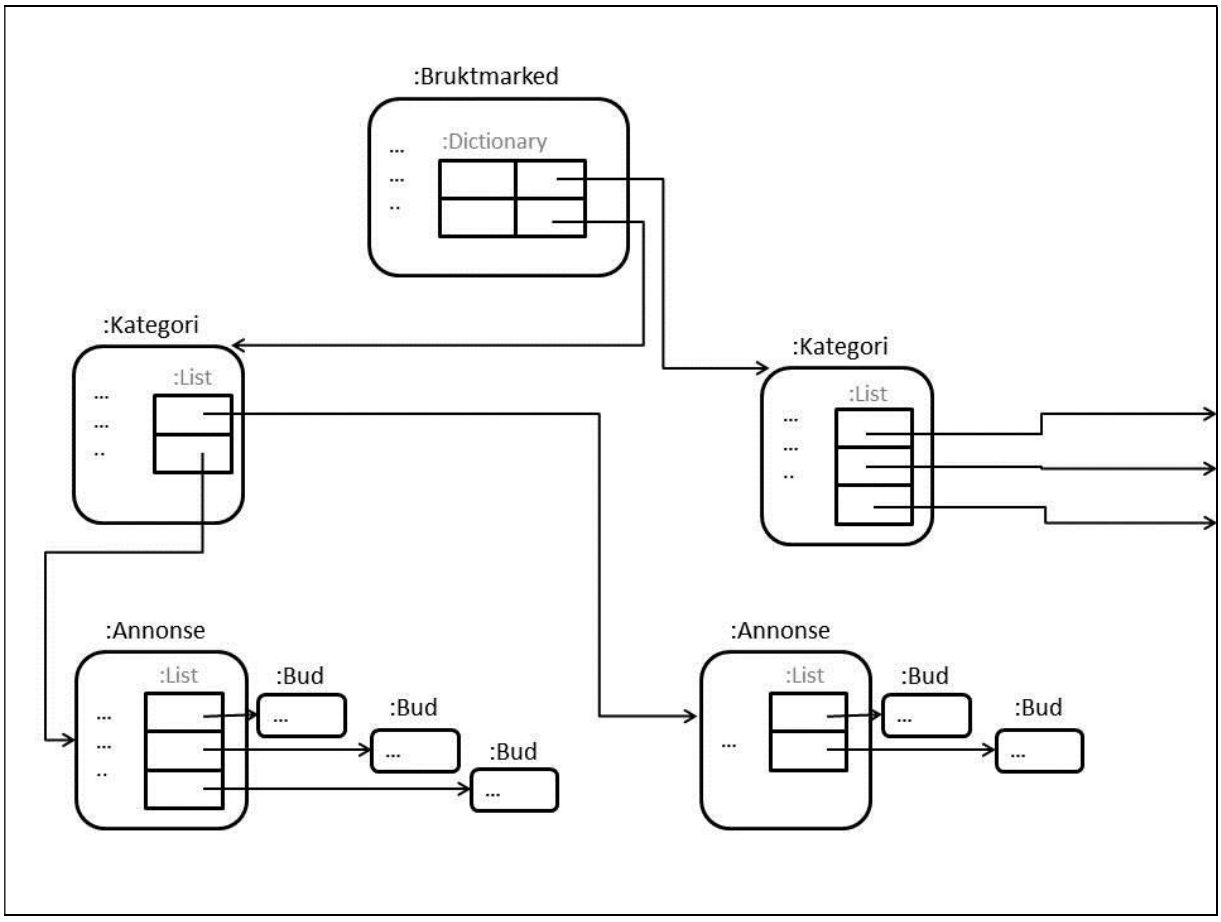
Maks poeng: 8

Question 17
Attached

Oppgave 4 Objektorientering (samlet tekst deloppgaver a-g)

Du skal implementere en pilot for en elektronisk markedsplass for kjøp og salg av brukte gjenstander. Implementasjonen skal baseres på tegningen nedenfor, som antyder hvordan datastrukturen med objekter av ulike klasser og referanser mellom dem vil kunne se ut under kjøring. Objektene er tegnet som rektangler med avrundede kanter. Hvert objekt er merket med sin klasse, for eksempel slik :Bud. Referanser er tegnet som piler fra variabel til objekt. Denne oppgaven inneholder deloppgaver a) til g), som til sammen beskriver en første versjon av dette programmet.

Om du hopper over en deloppgave er det likevel viktig at du leser hele denne teksten (den ligger samlet som en pdf-fil vedlagt hver deloppgave). I hver av deloppgavene kan du bruke klassene og metodene fra tidligere deloppgaver, også om du ikke har programmert dem.



Oppgave 4a) (5 poeng)

Klassen **Bud** skal ha en konstruktør med parametere for initiering av instansvariablene **budgiver** (navn) og **budStr** (budstørrelse). Dersom budstørrelse oppgis negativ eller 0, skal instansvariabelen settes til 1. Klassen **Bud**s grensesnitt skal videre tilby metodene **hentBudgiver** og **hentBudStr** som returnerer verdien i de respektive instansvariablene.

Skriv klassen **Bud** med alle metoder.

Oppgave 4b) (10 poeng)

Skriv klassen **Annonse** med alle metoder. Pass på å lagre budene på en annonse i rekkefølge, slik at et nytt bud alltid har høyere indeks enn et bud som er kommet inn tidligere på samme annonse. Klassen **Annonse** har følgende grensesnitt:

```
def __init__(self, annTekst):      # konstruktør

def hentTekst(self) :              # returnerer annonseteksten

def giBud(self, hvem, belop) :     # oppretter og legger til
                                   # nytt bud i en liste over
                                   # bud på annonsen

def antBud(self):                  # returnerer antall bud

def hoyesteBud(self) :             # returnerer høyeste bud.
                                   # Hvis to er like høye
                                   # vinner det som kom først
```

Oppgave 4c) (6 poeng)

Skriv klassen **Kategori** (alle metoder) med følgende grensesnitt:

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

def nyAnnonse(self, annTekst) : # oppretter og plasserer en ny
                              # annonse i kategorien,returnerer
                              # referanse til den nye annonsen
def hentAnnonser(self) :      # returnerer annonse-listen i
                              # kategorien
```

Oppgave 4d) (8 poeng)

Skriv klassen **Bruktmarked** (alle metoder) med følgende grensesnitt. Bruk en Dictionary for administrasjon av kategorier i bruktmarkedet.

```
def __init__(self) :          # Konstruktør

def nyKategori(self, katNavn) : # Opprett ny kategori i
                              # bruktmarkedet. Hvis
                              # kategorien allerede finnes,
                              # returner metoden None. Ellers
                              # returneres en referanse til den
                              # nye kategorien

def finnKategori(self, katNavn) : # returnerer referanse til
                              # oppgitt kategori, eller None om
                              # den ikke finnes
```

Oppgave 4e) (8 poeng)

Utvid klassen Annonse med følgende metode (som du skal implementere):

```
def kraftBud(self, hvem, belop, max) :# Oppretter nytt bud som
                                      # legges til listen over bud på
                                      # annonsen. Hvis ingen høyere bud
                                      # finnes settes budets belop som
                                      # oppgitt, ellers settes beløpet
                                      # til det hittil høyeste budet
                                      # for annonsen + 1, men
                                      # ikke høyere enn max.
```

Oppgave 4f) (8 poeng)

Skriv et hovedprogram som gjør følgende: Opprett et objekt av klassen Bruktmarked. Opprett en kategori «sykkellykt» på bruktmarkedet ditt, og legg inn en annonse med teksten «New Yorker sykkellykt» i denne kategorien. Legg deretter inn tre bud på den annonsen:

- Et normalt bud fra Peter på 42
- Et normalt bud fra Ann på 0
- Et kraftbud fra Mary på 40 med maks 50.

Test at høyeste bud i kategorien «sykkellykt» har forventet budgiver og beløp.

Oppgave 4g) (5 poeng)

Det hender at kunder byr lavere enn et bud som allerede er kommet inn på en annonse. Legg til (og implementer) en metode **tellLaveBud** i klassen **Bruktmarked** som teller opp slike bud for alle annonser i alle kategorier, og returnerer totalantallet. Utvid klassene **Kategori** og **Annonse** med nye metoder som trengs for å implementere **tellLaveBud**.