EKSAMENSOPPGAVE I INF-1400

Eksamen i	:	**INF-1400 – Objektorientert Programmering**
Eksamensdato	:	**2010-06-01**
Tid	:	09.00-13.00
Sted	:	**Åsgårdveien 9**
Tillatte hjelpemidler	:	**Ingen**

Oppgavesettet er på 4 sider inkl. forside

Kontaktperson under eksamen:
John Markus Bjørndalen,
92671202

Les gjennom hele oppgavesettet før du begynner å løse oppgavene.

Del A, IfI på flyttefot

Institutt for Informatikk er på flyttefot for tiden, men noen har rotet med plasseringen av pappesker og møbler slik at ting ikke har havnet der det skulle. Heldigvis har alt blitt merket slik at vi raskt fikk en liste over hva som havnet hvor. Oppgaven vår nå er å lage et program som representerer eiendelene og rommene og deretter lage noen metoder som hjelper oss å flytte alt inn på rett kontor.

Vi definerer følgende klasser:

- En pappeske, som angir et romnummer for hvor pappesken skal være.
- En stol, som angir et romnummer for hvor stolen skal være.
- Et bord, som angir et romnummer for hvor bordet skal være.
- Et rom med et angitt romnummer samt en liste over eiendeler som har havnet i rommet.
- NTFAK-bygget, som har en liste over rom.

Oppgave 1 (10%)

Listen over klasser er ikke komplett. Vi kan generalisere klassene Pappeske, Stol og Bord ved å introdusere arv og en ny klasse Ting. Legg til denne klassen og angi hvordan du vil endre på klassene som du bestemmer at skal arve fra denne.

Oppgave 2 (15%)

Implementer klassene vi har spesifisert til nå. Du trenger ikke å ta med noen andre metoder enn __init__() siden vi skal jobbe videre med klassene senere.

Oppgave 3 (20%)

a) Tegn opp klassediagrammet for klassene dine. Du trenger ikke å ta med metoder.
b) Tegn opp datastrukturen med et lite antall rom og noen få eiendeler i hvert rom.

Oppgave 4 (10%)

Vi antar at IfI plutselig har fått mange frivillige hjelpere, så vi har en person som er ansvarlig for å hente alle eiendeler til hvert enkelt rom. Hver av hjelperne trenger en liste over hvilke rom disse skal hentes i og hvilke eiendeler som skal hentes i hvert av de angitte rommene..

Lag en metode (finnEiendeler()) for NTFAK-klassen som søker gjennom alle rom etter eiendeler som egentlig skulle ha vært levert til et angitt rom. Metoden tar en parameter (romnummeret som vi leter etter eiendeler til), og returnerer en liste over eiendeler og rommene de forskjellige eiendelene ble funnet i. Listen skal kun inneholde feilplasserte ting.

Som et forslag kan listen være på følgende form:

[(romnummer, ting), (romnummer, ting), ...]

Hvor "romnummer" er rommet tingen ble funnet i, og ting er tingen som ble funnet (eller en referanse til den). Dere skal *ikke* fjerne tingene fra rommene.

Oppgave 5 (15%)

For å rydde opp i kaoset trenger vi også en metode som oppdaterer registeret ved å flytte feilplasserte eiendeler over til et angitt rom (flyttEiendeler(romNr)). Metoden har som eneste parameter hvilket romnummer den skal flytte eiendeler over til.

Skriv også noen metoder som gjør det enkelt å fjerne og legge til eiendeler i rommene samtidig som de beskytter implementasjonen av rom-objektene. Få tydelig fram hvilken klasse du legger metodene inn i.

Del B

Oppgave 6 (10%)

Hvilken hensikt har get_X() og set_X()-metoder i klasser ("getters"/"setters")? Hvilket problem er de ment å løse, og hvordan gjør de det?

Oppgave 7 (10%)

Gitt følgende klassedefinisjon:

```
class Thing:
    def __init__(self):
        self._a = 1
        self._b = 4

    def foo(self, param):
        self._a = self._a + param
        self._b = self._b + param
        return (self._a + self._b)

    def bar(self, param):
        a = self._a + param
        b = self._b + param
        return (a+b)

    def __str__(self):
        return 'a is ' + str(self._a) + ', b is ' + str(self._b)
```

Beskriv *presist* hva Python vil skrive ut når vi kjører følgende kode.

```
it = Thing()
print it.foo(2)
print it.bar(3)
print it
```

Oppgave 8 (10%)

Velg to av uttrykkene/konseptene under og gi en kort beskrivelse av dem.
- encapsulation (innkapsling)
- mutable vs. immutable
- private vs. public (i Python gjør vi dette via konvensjoner).
- polymorphism
- has-a vs. is-a relationship