UiT

NORGES
ARKTISKE
UNIVERSITET

EKSAMENSOPPGAVE

Eksamen i:	**INF-1400 Objektorientert programminering**
Dato:	**Mandag 26. mai 2014**
Tid:	**Kl 09:00 - 13:00**
Sted:	**Adm.bygget, B154**

Tillatte hjelpemidler: Ingen

Oppgavesettet er på 5 sider inklusiv forside.

Kontaktperson under eksamen: John Markus Bjørndalen.
Telefon: 90148307

NB! Det er ikke tillatt å levere inn kladd sammen med besvarelsen

Les gjennom hele oppgavesettet før du begynner å løse oppgavene.

Oppgavene kan besvares på Norsk, Engelsk, Svensk eller Dansk.

I besvarelsen kan du bruke Python, pseudokode eller en kombinasjon.

Noen norske termer som er brukt i stedet for de engelske fra boka:

- Klasse - class
- Arv - inheritance
- Atributt - Attribute
- Metode - Method

Del A

Sommeren 2014:

NTfak har begynt å flytte inn i det nye teknologibygget. Uheldigvis kom en pakke fra et ukjent sted med på flyttelasset og vi har enda ikke oversikt over hvordan ting skjedde, men pakken ble åpnet og bygget måtte evakueres. De ansatte er nå til behandling hos veterinær. Svarte biler med diplomatskilt har blitt observert i nærområdet, og amerikanske og russiske "fiskebåter" har søkt nødhavn ved Karlsøya.

Vi må lage et system for å holde øye med spredningen og for å sende inn droner for overvåkning og roboter som kan rydde opp. Heldigvis er mye av arbeidet gjort for oss allerede, men den forrige som jobbet med koden ble infisert før han ble ferdig. Vi må dermed rydde opp litt (han ble litt "kreativ" mot slutten) og implementere noen ting som mangler før vi kan begynne å bruke systemet.

Vi har ikke tilgang til mer informasjon om selve infeksjonen, alt dette er foreløpig klassifisert...

Oppgave 1 - 20%

Vi har følgende klasser i systemet som representerer droner:

- `HelikopterDrone` - som har `observasjoner`, `drivstoff`, `fart` og `mål`
- `FlyDrone` - som har `observasjoner`, `drivstoff`, `fart` og `mål`
- `BallongDrone` - som har `observasjoner`, `drivstoff`, `fart`, `mål` og `gass`

Vi har kun angitt attributter i denne oppgaven, metoder kommer vi til senere.

a) Generaliser klassene over ved å introdusere en ny klasse (`Drone`). Tegn opp klassehierarkiet for den nye klassen og klassene som er angitt over.

Oppgave 2 - 20%

Klassen `Observasjon` (se under) har informasjon for alle typer observasjoner som vi støtter i programmet nå, noe som er unødvendig og gjør det vanskelig å skille observasjoner fra hverandre eller legge til nye typer observasjoner. Vi skal nå bruke spesialisering for å rydde opp i programmet.

```
class Observasjon(object):
    def __init__(self):
        # Felles
        self.posisjon = None
        self.tidspunkt = None
        self.bilde = None  # bilde tatt av dronen

        # Mennesker. mulig_infisert er True kun hvis det er en sannsynlig infeksjon.
        self.mulig_infisert = False

        # Disse attributtene er kun for observasjon av biler
        self.baatreg = None
        self.baatstoerrelse = None
        self.baatfarge = None
        self.baattype  = None


        # Disse attributtene er kun for observasjon av biler
        self.regnummer = None
        self.bilfarge  = None
        self.biltype   = None
```

a) Skriv om koden over slik at vi får ryddet opp og representert de forskjellige observasjonene som nye typer (bil, båt, menneske). Husk å angi hvordan du bruker arv.

Du trenger bare å forholde deg til de attributtene som allerede ligger i `Observasjon` (ikke legg til noen attributter eller metoder). Du trenger heller ikke å kopiere kommentarer fra koden over. Hvis du ønsker det kan du omdøpe attributtene slik at de gir mer mening (bare ikke bli så kreativ at vi ikke forstår hva du mener).

Oppgave 3 - 15%

Anta at vi har en liste over droner (kalt `droner`).

a) Tegn opp `droner` med noen droner (3-4) og observasjoner (5-6) av forskjellige typer. Ta med hvordan objektene refererer til hverandre.

Du velger selv hvilke av dronene som har observert noe og hva.

Oppgave 4- 15%

Vi trenger nå å få plottet inn markører på et oversiktskart for å få oversikt over spredningen av infeksjoner. Vår forgjenger ble ikke ferdig, men han har etterlatt seg en tom funksjon vi må skrive innholdet til.

```
def plott_mulige_infeksjoner(map, droner):
    pass
```

For å plotte en posisjon på kartet kan vi bruke følgende metode på kartet: `map.set_marker(position)`.

a) Fyll ut funksjonen `plott_mulige_infeksjoner` slik at vi setter ut en markør på kartet for alle observerte mennesker som kan være infisert.

Del B

Oppgave 5 - 15%

Klassene under var noe av det siste forgjengeren vår jobbet med. Klassenavnene kan umulig gi mening, men klassene brukes enkelte steder i koden, så vi må finne ut hva den gjør og hvor den feiler.

```
def significant_match(template_list, snarf):
    return True # TODO: can't be bothered today - fix later

class Platypus(object):
    def __init__(self, templates):
        self.templates = templates

    def oinker(self, snarf):
        return significant_match(self.templates, snarf)

class Zebra(Platypus):
    def __init__(self, templates):
        Platypus.__init__(self, templates)

    def oinker(self, snarf):
        return False # Cannot be

    def hoot(self, msg):
        print "cannot hoot - no owls present"


class Bat(Platypus):
    def __init__(self, templates):
        Platypus.__init__(self, templates)

    def hoot(self, msg):
        print "Almost owl...ish"


z = Zebra([1,2,3])
b = Bat([4,5,6])
p = Platypus([7,8,9])
```

For hvert av uttrykkene under, angi hva som skrives ut. Hvis noe feiler, angi hvorfor. Vi antar at `obs` er et objekt av typen `Observasjon`.

a) `print z.oinker(obs)`

b) `print b.oinker(obs)`

c) `print p.oinker(obs)`

d) `z.hoot("detected")`

e) `b.hoot("detected")`

f) `p.hoot("detected")`

Oppgave 6 -15%

Gi en kort beskrivelse av 2 av de følgende uttrykk/konsepter:

- Klasse vs. objekt
- Polymorfi (det holder å angi 1 variant av polymorfi)
- Multiple inheritance
- Mutable/immutable
- getter/setter