UiT
NORGES
ARKTISKE
UNIVERSITET

# EKSAMENSOPPGAVE

| | |
|---|---|
| Eksamen i: | Inf-1400 Objektorientert programmering |
| Dato: | 24.05.2016 |
| Klokkeslett: | 9:00-13:00 |
| Sted: | Adm.bygget, B.154 |
| Tillatte hjelpemidler: | Ingen |
| Type innføringsark (rute/linje): | Digitalt |
| Antall sider inkl. forside: | **4** |
| Kontaktperson under eksamen: | **Lars Ailo Bongo** |
| Telefon/mobil: | **92015508** |

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

**NB! Det er studentens ansvar å sørge for at eventuelle bilde vedlegg er lesbar.**

I denne eksamen kan du bruke Python, pseudokode, eller en kombinasjon.

# Oppgave 1

Dette er en modifisert versjon av en tidligere eksamensoppgave.

Året er 2239 og vi har fortsatt ikke mer en tre romstasjoner og en liten koloni på månen. Romfartøyene som reiser rundt i solsystemet er hovedsakelig droner brukt til forskning, og noe kommersielle aktiviteter som gruvedrift.

Og nå har vi nok ett problem.

En feil i et gammelt program fra 2014 førte nesten til en global katastrofe. Heldigvis greide vi å redde jorden ved hjelp av objektorientert programmering i 2234. Dessverre glemte vi å rette den opprinnelige feilen, så vi har nok en gang feilberegnet banen til en komet. I stedet for å suse forbi jorden nær nok til å lage pene farger på himmelen kommer den til å treffe midt på. Antall organismer på jorden, inkludert store deler av menneskeheten, kommer til å bli kraftig redusert.

Heldigvis kan vi bruke samme løsning som sist gang, da vi sendte noen av forsknings og gruvefartøyene for å skyve kometen ut av banen. Operasjonen er igjen vanskelig siden det er en komet som består av is og stein, så vi må skyve forsiktig fra flere steder samtidig. Skulle den knekke opp vil heldigvis tyngdekraften hjelpe oss. Skyver vi de største fragmentene ut av banen vil de mindre fragmentene trekkes etter.

Vi må gjøre en koordinert operasjon med flere fartøy og vi trenger overvåke operasjonen underveis. Til det trenger vi ett program fro å holde styr på fartøyene våre samt kometen og fragmenter av den (den gamle løsningene var dessverre lagret i Fronter så vi har ikke tilgang til den lenger).

Vi definerer følgende klasse:

- KometFragment: fragmenter av kometen. Disse har attributtene *posisjon* og *størrelse*.
- Komet: representerer kometen som en samlet enhet og inneholder en liste over *fragmentene* som tilhører kometen. Hvis kometen ikke har brutt opp enda vil listen inneholde ett stort fragment som tilsvarer hele kometen.
- Operasjon: som inneholder en liste over *romskipene*.
- SensorRomskip: droner med sensorer (kamera, radar, laser, etc) som vi trenger for å overvåke operasjonen. Romskipet vil ha en nåværende *posisjon* og *målposisjon* (stedet dit den skal bevege seg for best bruk av sensorer).
- SkyveRomskip: store droner som er i stand til å skyve på kometen eller fragmentene. Romskipet vil ha attributtene nåværende *posisjon*, referanse til *fragmentet* den skal skyve på, og *målposisjon* for tilfeller der den skal holdes unna de andre og avvente nye mål.

Vi antar at posisjonene er av typen Posisjon som allerede eksisterer og at størrelse angis med et flyttall.

Merk at fokuset i oppgaven er objektorientering og bruk av datastrukturer vi bygger opp. Det ikke anbefalt å utvide oppgaven utover det som er spesifisert nedenfor!

## 1a (10%)

Vi har to typer romskip med noen ting til felles. Dette gjør at vi ønsker å generalisere ved å introdusere arv og legge til en ny klasse Romskip.

Forklar kort hvordan du kan bruke arv når du legger til Romskip og hvordan du endrer de andre Romskip-klassene. Det er kun behov for å fokusere på arv og attributtene til klassene siden vi skal jobbe med metodene senere.

## 1b (15%)

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

## 1c (10%)

Tegn opp datastrukturene med et lite antall romskip av begge typer og ett lite antall kometfragmenter. Du trenger ikke å angi verdier for attributter som posisjon og størrelse Noen av skyveromskipene skal skyve på noen kometfragmenter.

## 1d (15%)

Vi må ha ett system for å plassere ut SensorRomskip på angitte steder rundt kometen. Vi skal implementere en metode:

```
Def plasserSensorRomskip(self, posisjoner):
      # kode
```

Posisjoner er en Python-liste med de posisjonene som vi skal sende SensorRomskip til.

plasserSensorRomskip skal gå gjennom romskiplisten som ligger i Operasjon og finne alle SensorRomskip i denne listen. For hver posisjon skal den velge ut et sensorromskip og kalle settMålPosisjon(posisjon) for å sette ett nytt mål.

For SkyveRomskip skal vi også ha en settMålPosisjon, men den trenger å gjøre mer enn sensorromskipets settMålPosisjon. Når skyveromskipet blir sendt til en posisjon skal det samtidig avbryte skyvejobben den eventuelt er i gang med. Dette kan vi gjøre ved å sette *fragment* attributten i skyveromskipet til None.

Anngi hvilken klasse du vil plassere plasserSensorRomskip i og legg til koden for metoden i denne. Skriv også koden for de to versjonene av settMålPosisjon og angi hvilke klasser du har lagt koden i.

# Oppgave 2

I denne oppgaven skal dere lage ett program for biologisk klassifisering, som er «… en metode for å kategorisere organismer i et hierarkisk system. Kategori-baserte systemer bruker en gitt antall nivåer i hierarkiet slik som Rike, familie, slekt og art.» [Wikipedia]

For hver art trenger vi lagre følgende informasjon, der teksten etter kolon er verdiene for mennesker:

- Vitenskapelig navn: Homo sapiens
- Norsk navn: Menneske
- Biologisk klassifikasjon:
    - Rike: Dyreriket
    - Rekke: Ryggstrengdyr
    - Klasse: Pattedyr
    - Orden: Primater
    - Familie: Menneskeaper
- IUCN rødliste: LC (gyldige verdier er EX, EW, CR, EN, VU, NT, LC)
- Nasjonal rødliste: LC (gyldige verdier er EX, EW, CR, EN, VU, NT, LC)
- Habitat: Terrestrisk
- Utbredelse: Hele jorden
- Bilde: (gyldige filformater er PNG, JPEG, TIFF)
- Beskrivelse: (tekst som beskriver arten)

Programmet skal tilby følgende funksjonalitet:

- Vis informasjon om en art.
- Vis informasjon om en art på en liten skjerm, ved å skalere ned størrelsen på bildet og bruke en layout tilpasset små skjermer, Merk at det ikke er nødvendig å huske korrekte navn på metoder fra graf biblioteker her.
- Søk etter vitenskapelig navn.
- Søk etter norsk navn.

## 2a (45%)

Implementer ett skjellet for programmet ved hjelp av objektorientering, der du har med klassene, attributtene, metodene, og de viktigste data strukturene. Merk at du har begrenset med tid for denne oppgaven, så innholdet i de fleste funksjonene bør være pseudokode og/eller kommentarer som beskrives overordnet hva funksjonen gjør.

## 2b (5%)

Det er beregnet at det finnes tusen milliarder arter på jorden ($10^{12}$). Beskriv hvordan du kan optimalisere søke funksjonene hvis programmet har informasjon om alle $10^{12}$ artene. Merk at denne oppgaven kun teller 5% for karakteren, så du bør ikke bruke for mye tid på denne.