Forside

UNIVERSITETET I OSLO
Det matematisk-naturvitenskapelige fakultet
Prøve-eksamen i: IN1900, INF1100, IN-KJM1900
Vedlegg: Ingen.
Tillatte hjelpemidler: Ingen.

Les gjennom hele oppgavesettet før du begynner å besvare oppgavene. Eksamen består av flervalgsoppgaver og tekstoppgaver hvor du skal skrive små programmer eller tolke programmer og skrive hva som vil skrives ut på skjermen. Hvis du savner opplysninger, kan du legge dine egne forutsetninger til grunn og gjøre rimelige antagelser, så lenge de ikke bryter med oppgavens "ånd". I tekstoppgaver bør du da gjøre rede for forutsetningene og antagelsene du gjør, for eksempel i kommentarer til koden.

De fleste oppgavene gir kort kode med lite behov for kommentarer, med mindre man gjør noe komplisert eller ikke-standard (anbefales ikke; men i så fall skal kommentarene forklare ideene bak program-konstruksjonene slik at det blir lett å vurdere koden).

En oppgave kan be deg skrive en funksjon. Et hovedprogram der man kaller funksjonen er da ikke påkrevd, med mindre det er angitt. I denne typen oppgaver kan du også anta at nødvendige moduler er importert utenfor funksjonen. I andre tilfeller der du blir spurt om å skrive et program, er eksplisitt import av moduler en viktig del av besvarelsen.

1 Hva skrives ut?

Anta at følgende program utføres:

```
x = [k**2 for k in range(0, 100, 3)]
y = x[3] - x[0]
print(y)
```

Hva skrives ut?

Velg ett alternativ

- 144
- 9
- 81
- 27
- 3

Maks poeng: 2

2 Hva skrives ut?

Hva skrives ut på skjermen når dette programmet utføres?

```
a = range(100)
b = range(100)
c = 0
for i in range(1, 100):
    c += b[i] - a[i-1]
print(c)
```

Velg ett alternativ

- 1
- 99
- 101
- En feilmelding
- 81
- 0

Maks poeng: 1

3 Hva skrives ut?

Hva skrives ut på skjermen når dette programmet utføres?

```
s = 'TGAAGTCATA'
d = {'A': 0, 'C': 1, 'G': 2, 'T': 3}
cnt = [0, 0, 0, 0]
for k in range(len(s)):
    cnt[d[s[k]]] += 1
print(cnt)
```

Velg ett alternativ

- ['A', 'C', 'G', 'T']
- [4, 1, 2, 3]
- [1, 2, 3, 4]
- [4, 2, 1, 3]
- [1, 3, 4, 2]

Maks poeng: 1

4 Hva skrives ut?

Hva skrives ut på skjermen når dette programmet utføres?

```
x = {}
for k in range(-5, 6):
    x[k] = 2**k
print(x[-1] * x[1])
```

Velg ett alternativ

- 32.0
- -1.0
- 1.0
- 2.0
- En feilmelding

Maks poeng: 1

5 Hva skrives ut?

Hva skrives ut på skjermen når dette programmet utføres?

```
def repl(s, d):
    new_s = ''
    for i in range(len(s)):
        if s[i] in d:
            new_s = new_s + d[s[i]]
        else:
            new_s = new_s + s[i]
    return new_s

s = 'ACGGCTA'
d = {'A': 'G', 'G': 'A'}
print(repl(s, d))
```

Velg ett alternativ

- GCACTG
- GCTCGCA
- GCAAGTC
- GCAACTG
- GCACTGA

Maks poeng: 1

6 Hva skrives ut?

Hva skrives ut på skjermen når dette programmet utføres?

```
from math import exp
class FClass:
    def __init__(self, ksi):
        self.ksi = ksi

    def __call__(self, s):
        ksi = self.ksi
        if ksi != 0:
            ans = (1+ksi*s)**(ksi+1)
        else:
            ans = exp(-exp(-s))
        return ans

    def F(self, x):
        return x**2;

F = FClass(1)
```

print(F(1))

Velg ett alternativ

- 3
- 2
- Det skrives ut en feilmelding
- 1
- 4
- Det skrives ikke ut noe

Maks poeng: 1

7 Hva er riktig?

Anta at vi har definert listen

a = [0, 1, 2, 4, 9, 16]

Hvilke av disse uttrykkene får da verdien True?

Velg ett eller flere alternativer

- a[3] + a[3] == [4, 4]
- zip(a,a)[2] == (2,2)
- a[1] + a[-1] == 17
- sum(a[1:2]) == 3
- a + a = [0, 2, 4, 8, 18, 32]

Maks poeng: 3

8 Sammenlikning av to funksjoner

Anta at f(x) og g(x) er to funksjoner som er definert for x $\in [0, 1]$. Det er noen ganger nyttig å avgjøre om den første funksjonen dominerer den andre, dvs om f(x) $\geq$ g(x) for alle x $\in [0, 1]$. I denne oppgaven skal du lage en funksjon i Python for dette. Mer presist skal du lage en Python-funksjon **larger(f, g)** hvor argumentene **f** og **g** er funksjoner som tar et float-argument og returnerer en float-verdi (se eksempel på slike funksjoner lenger ned). Vi ønsker at **larger(f,g)** skal returnere True hvis f(x) $\geq$ g(x) for alle x $\in [0, 1]$ og False i alle andre tilfeller. I praksis kan vi ikke sjekke *alle* verdier x $\in [0, 1]$, så programmet skal i stedet sjekke om betingelsen f(x) $\geq$ g(x) er oppfylt for alle x = i*h hvor i = 0, 1, ..., n og h = 1/n. Du kan i ditt løsningsforslag sette konstanten n = 1000. Her er et eksempel på bruk:

```
def f(x):
    return x**2

def g(x):
    return 0

print(larger(f,g))    # Skal skrive ut True siden x^2 >= 0 for alle  x ∈ [0, 1]
```

Skriv ditt svar her...

```
1
```

Maks poeng: 5

9 Testfunksjon for funksjonen i forrige oppgave)

For å teste om funksjonen **larger(f,g)** i forrige oppgave fungerer, skal du nå lage en testfunksjon. Testfunksjonen skal sjekke oppførselen til **larger(f,g)** for to ulike valg av funksjonene **f** og **g**. Det ene er

$$f(x) = x + 1 \text{ og } g(x) = x$$

og det andre er

$$f(x) = x - 0.5 \text{ og } g(x) = (x - 0.5)^2$$

I det første eksemplet er riktig svar (expected) True, og i det andre eksemplet er riktig svar (expected) False. Forklar hvorfor dette er riktige svar, og skriv testfunksjonen **test_larger()**.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

10 Beregne verdien til en rekke

Lag en funksjon **s(n)** i Python hvor **n** er et positivt heltall og hvor verdien som returneres fra funksjonen er verdien til uttrykket $\frac{1}{2} + \frac{2}{3} + \frac{3}{4} + \cdots + \frac{n-1}{n}$.

Skriv ditt svar her...

```
1
```

Maks poeng: 5

11 Testfunksjon for funksjonen i forrige oppgave

Skriv en testfunksjon for funksjonen **s(n)** som du laget i forrige oppgave. Testfunksjonen skal teste om **s(n)** gir riktig svar for de tre spesialtilfellene n = 1, 2, 3.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

12 Oppslagstabeller (dictionaries)

Anta at vi har utført følgende setninger:

a = {1:2, 2:3, 3:4, 4:1}

b = {4:5, 5:6, 6:1}

I hvilke av alternativene nedenfor blir resultatet True?

Velg ett eller flere alternativer

- a[3] == b[4]
- (abs(len(b)-3) == 0
- [a[k] for k in a] == [2,3,4,1]
- a[1]+a[2] == b[4]
- a[0] == 1:2
- b[a[3]] == 1

Maks poeng: 2

13 Stykkevise konstante funksjoner

En stykkevis konstant funksjon har formen

$$f(x; a_0, a_1, \ldots, a_n) = \begin{cases} y_0, & a_0 \leq x < a_1 \\ y_1, & a_1 \leq x < a_2 \\ & \ldots \\ y_{n-1}, & a_n \leq x \leq a_n \end{cases}$$

Her er x variabel til f, mens $a_0, \ldots, a_n$ er parametre til f. Vi har i kurset sett hvordan en matematisk funksjon med parametre kan implementeres som en klasse. Lag en klasseimplementasjon for stykkevise konstante funksjoner som den ovenfor (hvor n er et vilkårlig positivt heltall). Kall klassen **PiecewiseConstant** og implementer den slik at vi f.eks. kan skrive

```
f = PiecewiseConstant([0.5, 2.8, 3.7],  [3, 5.5, 6, 8])
print(f(6))
```

Den første programsetningen skal da lage et objekt svarende til funksjonen

$$f(x; a_0, a_1, \ldots, a_n) = \begin{cases} 0.5, & 3 \leq x < 5 \\ 2.8, & 5 \leq x < 6 \\ 3.7, & 6 \leq x \leq 8 \end{cases}$$

mens den andre programsetningen skal skrive ut funksjonsverdien i x = 6 (altså verdien 3.7). Merk: det finnes ferdige løsninger i Python for å lage stykkevise konstante funksjoner, f.eks. i modulen scitools.std. Du skal ikke benytte slike (og trenger ikke å kjenne til dem) når du løser denne oppgaven.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

14 Numerisk integrasjon

Det bestemte integralet av en funksjon $f(x)$ kan estimeres med Trapezoidformelen

$$\int_a^b f(x)\,dx \approx C + h\sum_{i=1}^{n-1} f(a+ih)$$

der $C = (h/2)(f(a) + f(b))$, $h = (b-a)/n$ og *n* > *0* er et heltall som bestemmer nøyaktigheten til estimatet (bedre nøyaktighet jo større *n* er). Lag en Python-funksjon **integrate(f, a, b, n)** som bruker formelen over til å beregne det bestemte integralet til en funksjon. Metoden skal returnere den beregnede verdien til det bestemte integralet.

Skriv ditt svar her...

```
1
```

Maks poeng: 5

15 Klasser

Skriv en klasse i Python som implementerer funksjonen g(x; w) = sin(wx) + cos(wx). Her er x en variabel, mens w er en parameter til funksjonen. Klassen skal inneholde nøyaktig tre metoder: en konstruktør (for å sette verdien til w), en metode __call__(self, x) for å beregne verdien til g i et punkt x, og en metode __str__(self) som returnerer en string med funksjonsuttrykket. Hvis f.eks. w er initialisert til 1.5, så skal sistnevnte metode returnere stringen 'sin(1.5 x) + cos(1.5 x)'. Når du har definert klassen, skal du også skrive noen programlinjer (som vi tenker oss ligger i samme program, men ikke inni klassen) som først oppretter et objekt av klassen og foretar initialiseringen w = 3.1415, og som deretter beregner og skriver ut på skjerm verdien til funksjonen g(x; w) i punktet x = 1, og som til slutt (via metoden __str__) skriver ut på skjerm funksjonsuttrykket.

Skriv ditt svar her...

```
1
```

Maks poeng: 5

16 Filhåndtering

a) Skriv en funksjon **read_file(filename)** i Python som leser en tekstfil med filnavn angitt i argumentet til funksjonen, legger innholdet i tekstfilen inn i en oppslagstabell (dictionary), og som til slutt returnerer oppslagstabellen. Se lenger ned for mer detaljer om filformatet og innlesingen av filen.

b) Skriv noen programsetninger (som vi tenker oss ligger i samme program som funksjonen du laget i punkt a) som først leser filen "hovedsteder.txt" ved hjelp av funksjonen du laget i punktet over og som deretter går i løkke hvor:

- brukeren spørres om navnet på et land og dette leses inn fra terminalvinduet
- oppslagstabellen brukes til å finne tilhørende hovedstad
- navnet på hovedstaden skrives ut i terminalvinduet

Løkken skal gå helt til brukeren svarer *stopp* under innlesningen av navn på land. Dersom brukeren oppgir et ord som verken er *stopp* eller en nøkkel i oppslagstabellen, skal det skrives ut en feilmelding.

Om innlesningen av fil: du skal anta at hver linje i filen består av to eller flere ord som er atskilt av en eller flere blanke tegn. Hver linje i filen skal leses og legges inn i oppslagstabellen med det første

ordet som nøkkel og resten av linjen (fra og med første ikkeblanke tegn) som tilhørende verdi. Hver linje blir da et "key-value pair" i oppslagstabellen. Her er et eksempel på en fil som skal kunne leses av funksjonen du lager:

```
Norge        Oslo
USA           Washington, D.C.
Frankrike    Paris
Brunei       Bandar Seri Begawan
Malaysia     Kuala Lumpur
Chile         Santiago
```

Merk: du skal i denne oppgaven anta at det første ordet på hver linje (som blir til en nøkkel i oppslagstabellen) ikke inneholder blanke tegn. For å håndtere landsnavn med blanke tegn kan vi tenke oss at de blanke tegnene er erstattet av et spesialtegn, f.eks. Sri_Lanka.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

17 Plotte en funksjon

Du skal i denne oppgaven lage et Python-program **plot_func.py** som skal kjøres fra kommandolinjen. Programmet skal plotte en funksjon $f(x)$ som brukeren oppgir som et kommandolinje-argument, for $x \in [0, 1]$. Eksempel: hvis vi kjører programmet fra kommandolinjen med følgende kommandoer:

Terminal> python plot_func.py sin(x)

Terminal> python plot_func.py "3*x2 + cos(2*x)"**

så skal programmet først plotte funksjonen sin(x) fra x=0 til x=1 og deretter plotte funksjonen $f(x) = 3x^2 + \cos(2x)$ fra x=0 til x=1. Du kan anta at funksjonen som oppgis på kommandolinjen bare har en variabel og at denne heter x. Du kan også anta at funksjoner som inneholder mellomrom oppgis som en streng omgitt av anførselstegn, som angitt over.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

18 SIR-modellering

Nederst i oppgaven finner du en delvis implementasjon av to klasser ProblemSIR og SolverSIR for å løse et spesielt system av differensiallikninger (ODE'er) numerisk. Utgangspunktet er SIR-modellen i læreboka beskrevet i øvelse E.41 og likningsløseren beskrevet i øvelse E.42 (ikke vedlagt her siden dere har tilgang til læreboka på prøveeksamen). SIR-modellen er imidlertid modifisert litt og har andre parametre. Du kan anta at modellen er som følger:

$$
\begin{aligned}
S'(t) &= -\alpha \cdot S(t) \cdot I(t) + \beta \cdot R(t) \\
I'(t) &= \alpha \cdot S(t) \cdot I(t) - \gamma \cdot I(t) \\
R'(t) &= -\beta \cdot R(t) + \gamma \cdot I(t)
\end{aligned}
$$

Din oppgave er nå å implementere ferdig klassene ProblemSIR og SolverSIR nedenfor. Du må skrive ferdig metodene __init__ og __call__ i klassen ProblemSIR, og du skal også skrive ferdig metoden plot i klassen SolverSIR. Sistnevnte metode skal plotte løsningene for S(t), I(t) og R(t) i ulike farger i samme plott for de gitte parametrene (α, β, γ) i modellen. Den skal i tillegg plotte (som stiplede kurver i tilsvarende farger) løsningene for S(t), I(t) og R(t) når parameteren α reduseres med 10% eller økes med 10%. Det ferdige plottet skal altså ha tilsammen tre kurver for hver av de tre variablene S(t), I(t) og R(t): en heltrukken og to stiplede.

```
import ODESolver
import numpy as np
import matplotlib.pyplot as plt
class ProblemSIR:
    def __init__(self, alpha, beta, gamma, S0, I0, R0, T):
        # Not yet implemented

    def __call__(self, u, t):
        # Not yet implemented


class SolverSIR:
    def __init__(self, problem, dt):
        self.problem, self.dt = problem, dt

    def solve(self, method=ODESolver.RungeKutta4):
```

```
            self.solver = method(self.problem)
            ic = [self.problem.S0, self.problem.I0, self.problem.R0]
            self.solver.set_initial_condition(ic)
            n = int(round(self.problem.T/float(self.dt)))
            t = np.linspace(0, self.problem.T, n+1)
            u, self.t = self.solver.solve(t)
            self.S, self.I, self.R = u[:,0], u[:,1], u[:,2]

        def plot(self):
            # Not yet implemented

    # Example of use
    problem = ProblemSIR(alpha=0.005, beta=1e-5, gamma=0.002, S0=100, I0=1, R0=0, T=50)
    solver = SolverSIR(problem, dt=0.2)
    solver.solve()
    solver.plot()
```

Skriv ditt svar her...

```
1
```

Maks poeng: 10