Front page

UNIVERSITETET I OSLO
Det matematisk-naturvitenskapelige fakultet
Prøve-eksamen i:
IN1900 og IN-KJM1900
Eksamensdag: 7. desember 2017
Tid for eksamen: 12.00-16.00
Vedlegg: 1 (ODESolver.pdf)
Tillatte hjelpemidler: Ingen.

- **VIKTIG!** Oppgavesettet har 4 deler: Del 1 og 2 er felles for IN1900 og IN-KJM1900. Del 3 (oppgave 13-15) er bare for IN1900. Del 4 (oppgave 16-22) er bare for IN-KJM1900.
- Max poengsum er 76 poeng. Flervalgsoppgaver og "hva skrives ut"-oppgaver teller 2 poeng pr riktig svar. På programmeringsoppgaver i del 2 gis det max 5 poeng pr oppgave. På programmeringsoppgaver i del 3 og del 4 gis det max 10 poeng pr oppgave.
- Les hele oppgavesettet før du begynner å løse oppgaver. Hvis du savner opplysninger, kan du legge dine egne forutsetninger til grunn og gjøre rimelige antagelser, så lenge de ikke bryter med oppgavens “ånd”. Gjør da rede for forutsetningene og antagelsene du gjør.
- 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.
- **Merk 1:** Kl 16 blir det servert pizza i biokantinen (1. etg i Kristine Bonnevies hus) til alle som har meldt seg på. Kl 17-19 blir prøveeksamen gjennomgått i plenum i Sophus Lies aud.
- **Merk 2:** Denne prøve-eksamenen har en blanding av norsk og engelsk. Endelig eksamen vil være tilgjengelig på begge språk.

1 Lists

Which one of these does **not** result in a list of length 4?

Select an alternative:

- a = [4]*4
- a = [0, 1] + [0, 1]
- a = [2*i for i in range(1,4)]
- a = [1]*4
- a = [4]*1 + [4]*3

2 Functions

Suppose the following function is defined:

```
def f(x, y, z, w=0):
    return x * y ** z + w
```

Which of these function calls are **not** correct?

Select one or more alternatives:

- f(1, 2, w = 2)
- f(1,2,3)
- f(1,2,3,4)
- f(1, 2, z = 6)
- f(1, 2)
- f(y=3, x=6, 6)

3 What is printed out?

Suppose the following program is executed:

```
x = [k**2 for k in range(1, 100, 3)]
y = x[3] - x[0]
print(y)
```

What is printed out?

Select an alternative:

- 9
- 99
- Nothing is printed out
- 27
- 3

4 What is printed out?

Suppose the following code is executed:

```
x = [2**k for k in range(11)]
y = [x[k+1]/x[k] for k in range(10)]
z = sum(y)
print(z)
```

What is printed out?

Select an alternative:

- 20
- 0
- An error message
- 1024

5 Functions

Suppose the following code is executed:

```
x = 2
y = 4

def f(x, y):
```

```
    z = x * y
    return z

result = f(x, x)
print(result)
```

What is printed out?

Select an alternative:

- 2
- 8
- 4
- 16

6 Test functions

Which of the following statements are correct?

Select one or more alternatives:

- The tolerance used in numerical comparisons in testing should be as small as possible.
- If a test function runs silently (no error message), the function being tested is correct.
- The statement **assert False, 'Error'** halts program execution with an error message.
- Unit testing means to test if the right units are used in mathematical calculations
- Test functions should always have at least one argument
- Test functions should always contain a return statement

7 Polynomial function

Define a Python function **poly(t, a, b, c)** that calculates the value of the polynomial $p(x) = ax^2 + bx + c$ in the point $x = t$. That is, the function should return the value $p(t)$

.

Fill in your answer here

```
1
```

8 Harmonic series

Write a Python function **harmonic(n)** that calculates the value of the expression $1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$ and returns it.

Fill in your answer here

```
1
```

9 Harmonic series II

Write a Python function **harmonic2(n)** which extends the function harmonic(n) in the previous exercise to also return the value of the first term not included in the sum. That is, the function should return both the value $1 + \frac{1}{2} + \cdots + \frac{1}{n}$ and the value $\frac{1}{n+1}$. You can call the function harmonic(n) inside the function harmonic2(n) if you want to.

Fill in your answer here

```
1
```

10 Is k a prime number?

A prime number is an integer k > 1 that is only divisible by itself and 1. In other words, to determine if k is a prime, we only have to show that k is not divisible by any of the integers 2, 3, ..., k-1. To test this in Python, note that k is divisible by j if and only if the Python expression k % j == 0 is True (for example, 12 is divisible by 6 since 12 % 6 == 0 is True). Note: there are much more efficient ways of determining if k is a prime number or not, but we do not consider this here.

Write a Python function **is_prime(k)** that returns the value True if k is a prime number, and returns False if k is not a prime number. You can assume that k is a positive integer.

Fill in your answer here

```
1
```

11 Calculating the first primes

Write a Python function **primes(n)** that returns in a list the n first prime numbers. For example, the call primes(5) should return the list [2, 3, 5, 7, 11]. You may use the function is_prime(k) from the previous question.

Fill in your answer here

```
1
```

12 Formatted printing

Write a Python function **primeTable(k, n)** that prints a nicely formatted table with k rows and n columns on the screen. In the first row, the function should print out the first n primes p1, ..., pn. In the second row, the function should print out the *absolute values of the first differences* between the numbers in the first row, i.e. |p2-p1|, |p3-p2|, |p4-p3|, ... In the third row, the function should print out the *absolute values of the first differences* between the numbers in the second row, etc. The call primeTable(5, 5) should result in a table similar to this:

```
2   3   5   7   11
1   2   2   4
1   0   2
1   2
1
```

Fill in your answer here

```
1
```

13 Monte Carlo simulation

An interesting property of a statistical distribution is the probability of observing values larger than a given threshold **theta**. For example, for the (continuous) uniform distribution on the interval [0, 2] the probability of observing values larger than 0.5 can be shown to be 0.25.

Suppose were can generate as many numbers as we like from a given distribution F using the function randomF(n). Here, n is the number of random values we want to generate and randomF(n) returns a list of length n containing the values. We want to use simulation to estimate the probability of observing values larger than a threshold theta. Write a function **probF(theta, n)** that does this and returns the estimated probability.

Fill in your answer here

```
1
```

14 Implementing a queue

A **queue** is a sequence where one can (1) add new elements at the start of the sequence and (2) remove elements from the end of the sequence. Since a queue consists of data (the sequence) as well as functions (adding and removing elements), it is well suited for implementation as a class.

Here is an unfinished implementation of a queue in Python:

```
class Queue:
    def __init__(self, elems):   # Here elems is a list of elements
        <Initiate the queue to contain the elements in elems>

    def add(self, elem):    # Here elem is a single element to add to the queue
        <Add elem at the start of the queue>

    def rem(self):
        <Remove the last element from the queue and return it>

    def length(self):
        <Return the number of elements in the queue>
```

def str(self):
<Return a string of all the elements in the queue from the first one to the
last one, using hyphens (-) to separate the elements from each other.
For example, if the queue consists of two elements "Peter" and "Park", then
str() should return the string "Peter-Park">

Complete the implementation of class Queue. Use a list to hold the elements in the queue.

Fill in your answer here

Format | Σ

Words: 0

15 Modeling with ODEs

This question presents a model for an outbreak of Ebola in Sierra Leone in 2014.
The population is divided into four groups; those that can be infected (S), those that are infected but have not yet developed the diseases, and can not yet infect others (E), those that are sick and can infect others (I), and those that have died from the disease (D).
Let *S(t), E(t), I(t)* and *D(t)* be the number of people in each category. The following system of differential equations describes the dynamics of *S(t), E(t), I(t)* and *D(t)* in a time interval [*0,T]:*

At time *t=0* we have the initial conditions $S(0) = S_0$, $E(0) = E_0$, $I(0) = 0$, $D(0) = 0$. The function p*(t)* and the constants *q* and *r* are assumed known. All constants and functions are >0.

Write a Python function SEID(S0,E0,p,q,r,T), which takes initial values S0, E0, the function p(t), constants q, r, and the end time T as input arguments. Use the class RungeKutta4 in the ODESolver hierarchy to solve the differential equations. The source code for ODESolver is attached as a pdf document. Let the time be given in days. Use ten time steps per day, so that the total number of time points for a simulation over [*0,T*] is 10T+1. The function SEID shall return 5 arrays:

- t, which holds the time points t_k where the numerical solution is calculated,
- S, which contains $S(0), S(t_1), \ldots S(t_n)$,
- E, which contains $E(0), E(t_1), \ldots E(t_n)$,
- I, which contains $I(0), I(t_1), \ldots I(t_n)$,
- D, which contains $D(0), D(t_1), \ldots D(t_n)$.

We reason as follows to set the necessary parameters in the model. At the outbreak of the disease (*t=0)* the population in Sierra Leone was 5.48 million, and we want to study an extreme case where 10% of the population is infected and about to develop the disease. This gives S0=4.93 and E0=0.55. On average it took 10 days from the time of infection to the start of the disease, and 10.38 days from the start of the disease until the person died. This gives *q = 1/10* and *r = 1/10.38.*

The function p*(t)* describes the likelihood of an infected person getting interacting with and infecting a healthy, susceptible person. If nothing is done to slow spreading of the disease we can assume that this is constant, and for the disease outbreak we study it was estimated to *p=0.0233.*

Write the code for calling the function SEID with the given parameters and T=100. Also add code to plot *S(t), E(t), I(t)* and *D(t)* in the same window, with a legend for each curve.

Fill in your answer here

```
1
```

16 Newtons metode: hva er sant?

Hva er sant om Newtons metode?
Velg ett alternativ:

- Newtons metode er iterativ, det vil si basert på gjentakelse av en prosedyre inntil visse kriterier er nådd
- Newtons metode gir alltid rett svar
- Newtons metode er en metode for å finne eksakte løsninger av en likning f(x)=0
- Newtons metode gir maks 10 gjeldende siffer i svaret.

17 Eulers metode: hva er sant?

Hva er sant om Eulers metode?

Velg de alternativene som er riktige påstander:

- [ ] Eulers metode er alltid eksakt
- [ ] Eulers metode fungerer best på likninger av formen f(x)=0
- [ ] Metoden gir løsningen på et gitter av tidspunkter
- [ ] Det er en metode for å finne tilnærmede løsninger av initialverdiproblemer

18 Eulers metode

Betrakt initialverdiproblemet

$\frac{d}{dt}y(t) = -c \cdot y(t), \quad y(0) = 1, \quad 0 \leq t \leq t_{\text{final}}$

Vi setter y_0=1.

Hvilken av de nedenstående formlene definerer Eulers metode med steglengde h?

- $y_{j+1} = y_j + c \cdot y_j$
- $y_{j+1} = y_j - hc \cdot y_j$
- $y_{j+1} = y_j + h \cdot y_j$
- $y_{j+1} = y_j + hc \cdot y_j$

19 Implementering av Eulers metode

```python
import numpy as np

n = 100
t_vec = np.linspace(0,1,n+1)
h = t_vec[1]-t_vec[0]

c = 0.5

y = np.zeros(n+1)
y[0] = 1.0

for j in range(n):
    y[j+1]=y[j] + c*y[j]

print(y)
```

Betrakt initialverdiproblemet

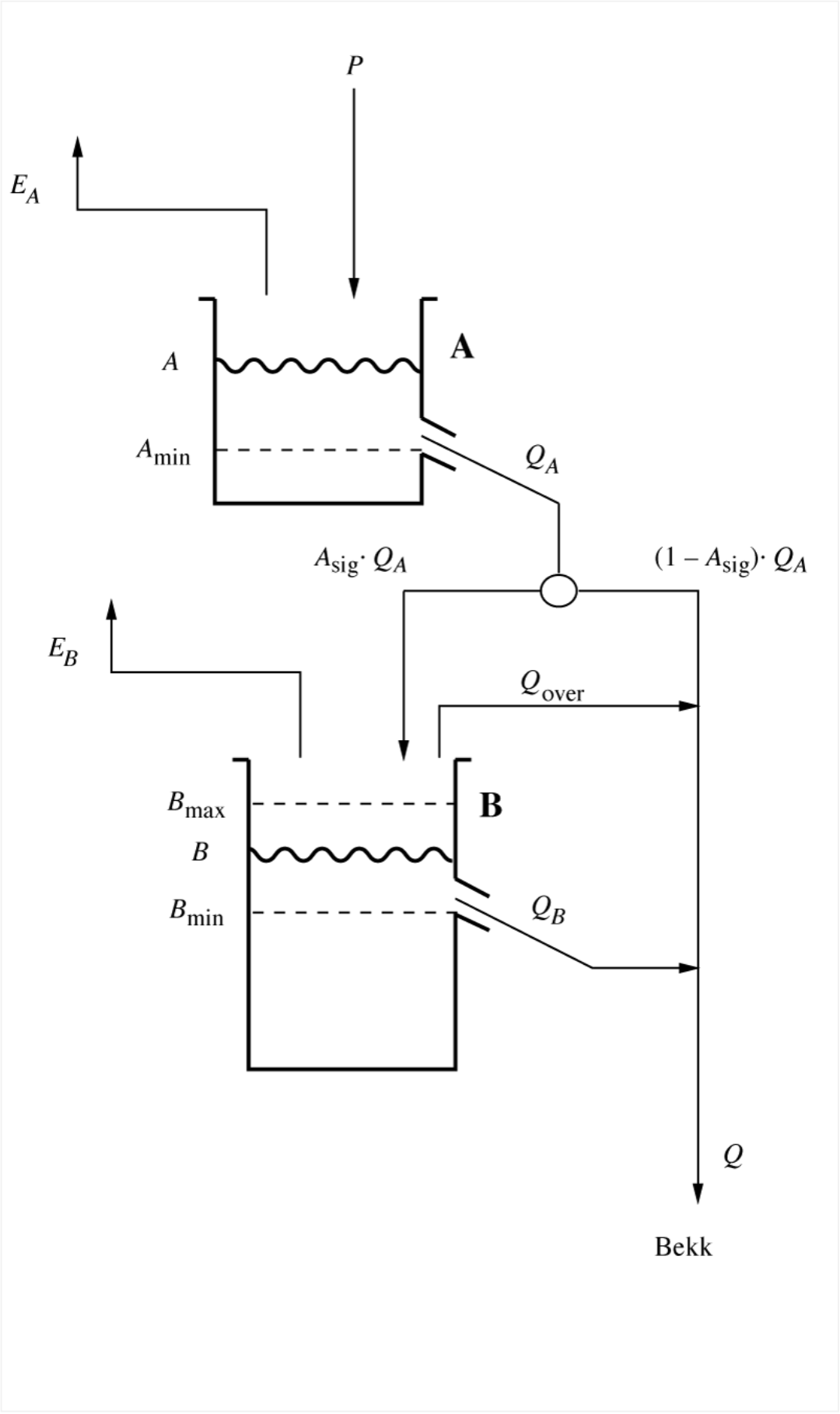
$$\frac{d}{d}y(t) = -c \cdot y(t), \quad y(0) = 1, \quad 0 \leq t \leq 1$$

Vi setter *c* = 0.5. Illustrasjonen viser et program som løser Eulers metode for dette problemet, men programmet har en bug i Euler-loopen. Finn denne bugen og korriger programmet så det blir korrekt.

Skriv inn ditt svar:

```
1
```

20 Birkenesmodellen



Illustrasjonen viser den hydrologiske delmodellen av birkenesmodellen.

a) Hvilke symboler representerer de ukjente variablene som vi finner med Eulers metode i den hydrologiske delmodellen?

b) Hvilke størrelser er strømmer?

c) Beskriv hva rollen til Q_{over} er. (Du trenger ikke skrive formelen for Q_{over}.)

Skriv inn ditt svar

Format | Σ

Words: 0

21 Elektronøytralitet

I enhver makroskopisk vannløsning er det ingen netto ladning. Dette kalles elektronøytralitetsprinsippet. Anta at vi løser vanlig bordsalt NaCl i vann. Da får vi en løsning med Na^+ og Cl^-. Hvilken likning beskriver elektronøytralitet for denne løsningen?

Velg ett alternativ:

- $[\mathrm{Na^+}] + [\mathrm{Cl^-}] = 0$
- $2[\mathrm{Na^+}] = [\mathrm{Cl^-}]$
- $\frac{[\mathrm{Na^+}]}{[\mathrm{Cl^-}]} = \mathrm{K_{NaCl}}$
- $[\mathrm{Na^+}] = [\mathrm{Cl^-}]$

22 Hva printes?

```python
f = lambda x: x**2 - 4
df = lambda x: 2*x

x = 1.0

for k in range(8):
    derivert = df(x)
    funksjon = f(x)

    if derivert == 0.0:
        raise ValueError, "df(x)=0"

    x = x - f(x)/df(x)

    print("x_%d = %.16f" %(k+1, x))

```

Illustrasjonen viser et Pythonskript som bruker Newtons metode for å løse likningen

$x^2 = 4.$

Studér skriptet nøye. Hva printes av dette skriptet?

Vink: I denne oppgaven er det ikke meningen at du skal regne ut hva som skjer på papir. Du kan løse oppgaven med kunnskapen du har om Newtons metode samt ved å studere skriptet.

1)

x_1 = 2.5000000000000000
x_2 = 2.0499999999999998
x_3 = 2.0006097560975609
x_4 = 2.0000000929222947
x_5 = 2.0000000000000022
x_6 = 2.0000000000000000
x_7 = 2.0000000000000000
x_8 = 2.0000000000000000

2)

x_1 = 2.5000000000000000
x_2 = 2.0499999999999998
x_3 = 2.0006097560975609
x_4 = 2.0000000929222947
x_5 = 2.0000000000000022
x_6 = 2.0000000000000000
Traceback (most recent call last):

```
  File "newton.py", line 11, in <module>
    raise ValueError, "df(x)=0"
ValueError: df(x)=0
```

3)

```
x_1 = -1.0000000000000000
x_2 = -1.8750000000000000
x_3 = -1.9629629629629630
x_4 = -1.9843750000000000
x_5 = -1.9920000000000000
x_6 = -1.9953703703703705
x_7 = -1.9970845481049562
x_8 = -1.9980468750000000
```

Velg ett alternativ:

- Alternativ 1
- Alternativ 2
- Alternativ 3