UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Examination in:	INF1100 — Introduction to programming with scientific applications
Day of examination:	Tuesday, October 12, 2010
Examination hours:	15.00 – 19.00.
This examination set consists of 7 pages.	
Appendices:	None.
Permitted aids:	None.

Make sure that your copy of the examination set is complete before you start solving the problems.

- Read through the complete exercise set before you start solving the individual exercises. If you miss information in an exercise, you can provide your own reasonable assumptions as long as you explain that in detail.
- The maximum possible score on the exam is 25 points. The maximum number of points is listed for each exercise (a correct answer of a subquestion ((a), (b), etc.) gives 1 point).

Exercise 1 (10 points)

What will be the output of the `print` statement in the programs below?

(a)

```
q = 3.141
print '%.1f' % q
```

Solution:

```
3.1
```

(b)

(Continued on page 2.)

```
for i in range(2, 6, 2):
    print i-1
```

Solution:

```
1
3
```

(c)

```
a = 4
b = 10/a + 1
print b
```

Solution:

```
3
```

10/4 gives integer division, resulting in 2, for Python versions 1.x and 2.x. With Python 3.x, 10/4 is 2.5.

(d)

```
A = [14] + [16, 18] + [25, 40]
del A[1]
print A
```

Solution:

```
[14, 18, 25, 40]
```

(e)

```
A = [-1, 9, 2, 5, 19, 21, 33]
print A[4:-1]
```

Solution:

```
[19, 21]
```

(f)

```
values = []
value = 0
stop = 1
incr = 0.2
while value <= stop:
    values.append(value)
    value += incr
for v in values:
    print v
```

(Continued on page 3.)

Solution:

```
0
0.2
0.4
0.6
0.8
1.0
```

(g)

```
print [0.2*i for i in range(6)]
```

Solution:

```
[0.0, 0.2, 0.4, 0.6, 0.8, 1.0]
```

(h)

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

x = 3; a = -2
print '%g' % f(x + a)
```

Solution:

```
-2
```

(i)

```
for i in range(2, 5):
    for j in range(i-1, i+3):
        if i != j:
            print i, j+1
```

Solution:

```
2 2
2 4
2 5
3 3
3 5
3 6
4 4
4 6
4 7
```

(Continued on page 4.)

(j)

```
def branch(argument):
    r = 0
    if argument == -1:
        r = -1
    elif argument == 3:
        r = 3
    elif argument > 3:
        r = -2
    else:
        r = 10
    return r

print branch(6)
```

Solution:

```
-2
```

Exercise 2 (2 points)

What is wrong with the programs below?

(a)

```
a = 9.1
b = raw_input('Give b: ')
print a*b
```

Solution: `b` is a string (`str` object), which must be converted to an `int` or `float`, or another object which allows (left) multiplication by a number. (The program aborts with a `TypeError`.)

(b)

```
1plus1 = 1 + 1
print 1plus1
```

Solution: `1plus1` is not a legal name in Python. Variables cannot start with a digit. (The program aborts with a `SyntaxError`.)

(Continued on page 5.)

Exercise 3 (1 point)

The following program aims to sum the integers $1, 2, \ldots, n$: $\sum_{i=1}^{n} i$. Explain if the result `s` is correct or not.

```
import sys
n = int(sys.argv[1])
s = 0                          # sum of integers
for i in range(1, n, 1):
    s += i
print s
```

Solution: `range(1,n,1)` results in the integers $1, 2, \ldots, n-1$, implying that the program computes $\sum_{i=1}^{n-1} i$, which is the wrong quantity. `range(1,n,1)` must be changed to `range(1,n+1,1)`, or just `range(n+1)`, so that the upper limit of the sum, `n`, is included.

Exercise 4 (4 points)

A function $p(x)$ is given as

$$p(x) = Ae^{-ax}\sin(wx),$$

with $A > 0$ and $a > 0$. Make a program that prints out a table of x and $p(x)$ values. Let the x coordinates in the table be uniformly distributed in $[0, 5/a]$. Read the number of x coordinates in the table, A, a, and w from the command line. Use exception handling to ensure that the user provides enough command-line arguments and that the values are legal.

Solution:

```
def p(x):
    return A*exp(-a*x)*sin(w*x)

from scitools.std import *
# or from numpy import exp, sin
# import sys

try:
    n = int(sys.argv[1])
    A = float(sys.argv[2])
    a = float(sys.argv[3])
    w = float(sys.argv[4])
except IndexError:
    print 'Usage: %s n A a w' % sys.argv[0]
    sys.exit(1)
except ValueError, e:
    print 'Illegal value - cannot convert to number!', e
```

(Continued on page 6.)

```
    sys.exit(1)

if A <= 0:
    #raise ValueError('A must be strictly positive')
    print 'A=%g is not > 0' % A; sys.exit(1)
if a <= 0:
    #raise ValueError('a must be strictly positive')
    print 'a=%g is not > 0' % a; sys.exit(1)

x = linspace(0, 5.0/a, n)
y = p(x)

for xi, yi in zip(x, y):
    print '%10.3f  %12.7f' % (xi, yi)
```

Exercise 5 (1 point)

Extend the program from Exercise 4 with statements for plotting the $p(x)$ function. (You may use the same coordinates as in the table.)
Solution:

```
plot(x, y)
```

(Note that if we did not store the x and $p(x)$ values in arrays in the first program, we would need to create such arrays here before calling `plot`.)

Exercise 6 (4 points)

An arbitrary triangle can be described by the coordinates of its three vertices: (x_1, y_1), (x_2, y_2), (x_3, y_3). The area of the triangle is given by the formula

$$A = \frac{1}{2}\left|x_2y_3 - x_3y_2 - x_1y_3 + x_3y_1 + x_1y_2 - x_2y_1\right|,$$

while the circumference ("omkrets" på norsk) is computed as

$$\begin{aligned} C &= \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2} + \sqrt{(x_3-x_2)^2 + (y_3-y_2)^2} + \\ &\quad \sqrt{(x_1-x_3)^2 + (y_1-y_3)^2}\,. \end{aligned}$$

Write a function `triangle(corners)` that returns the area A and the circumference C of a triangle whose vertices are specified by the argument `corners`, which is a nested list of the vertex coordinates. For example, `corners` can be `[[0,0], [1,0], [0,2]]` if the three corners of the triangle have coordinates $(0,0)$, $(1,0)$, and $(0,2)$. Demonstrate how to call the `triangle` function to compute the area and the circumference of the triangle with vertices $(-3,0)$, $(3,0)$, and $(0,4)$.
Solution:

(Continued on page 7.)

```
from math import sqrt

def triangle(corners):
    x1 = corners[0][0]; y1 = corners[0][1]
    x2 = corners[1][0]; y2 = corners[1][1]
    x3 = corners[2][0]; y3 = corners[2][1]
    A = 0.5*abs(x2*y3 - x3*y2 - x1*y3 + x3*y1 + x1*y2 - x2*y1)
    C = sqrt((x2-x1)**2 + (y2-y1)**2) + \
        sqrt((x3-x2)**2 + (y3-y2)**2) + \
        sqrt((x1-x3)**2 + (y1-y3)**2)
    return A, C

corners = [(-3, 0), (3, 0), (0, 4)]
A, C = triangle(corners)
print 'Area:', A, 'Circumference:', C
```

(Output: `Area:  12.0 Circumference:  16.0`)

Exercise 7 (3 points)

Make a function for solving the system of difference equations

$$s_j = s_{j-1} + a_{j-1}, \tag{1}$$
$$a_j = -x^2\left((2j+1)2j\right)^{-1} a_{j-1}, \tag{2}$$

with initial conditions $s_0 = 0$ and $a_0 = x$. In the function, store only the newest two s_j and a_j values (i.e., do not store all the s_j and a_j values in arrays). The function should take two arguments, x and N, and return the two values s_N and $|a_N|$. Write a main program that prints out the value of s_{20} for $x = \pi$.
Solution:

```
def sin_diffeq(x, N):
    aj_prev = x
    sj_prev = 0
    index = range(N+1)
    for j in index[1:]:
        sj = sj_prev + aj_prev
        aj = -x**2/float((2*j+1)*(2*j))*aj_prev
        sj_prev = sj
        aj_prev = aj
    return sj, abs(aj)

N = 20
from math import pi
x = pi
```

```
s_N, a_N = sin_diffeq(x, N)
print s_N
```

END

(Continued on page 9.)