MATEMATISKA INSTITUTIONEN
STOCKHOLMS UNIVERSITET
Avd. Beräkningsmatematik
Kursledare: Lars Arvestad
Examinator: Lars Arvestad

Tentamensskrivning i
DA2005 Programming techniques
7.5 hp
2021-04-21

- Part A has multi-choice questions with at least one correct answer. The wrong answer or the wrong number of answers both give zero points.
- You need to pass Part A (4 correct answers on 8 questions) for your Part B to be graded.
- Part B is a number of problems worth a total of 12 points, which are to be solved using Python 3 code.
- The answers to Part B are handed in as a single `.py` file named like `anonymouscode.py` where `anonymouscode` is the code given to you when registering for the exam in Ladok. You **must** also write your personal code in a comment at the top of your Python file. You **must not** write your own name anywhere in the file!
- Select identifiers for functions, methods, and variables as requested in the problems.
- No `import` statements may be used unless mentioned and requested in the problem. You are free to use any functions defined in the Python standard environment (available at startup), including `len`, `range` and `map`.
- You should write **Python 3** code, and *not* Python 2.7, for example.
- **Resources:** You are allowed a "cheat sheet" for Part A: an A4 paper filled with as much information as you like. It can be written/printed on both sides. Part B is *open book*, so the same rules apply for Part B as in labs and project.
- **Grading thresholds:** E: 10, D: 12, C: 14, B: 16, A: 18, out of maximum 20.

Part A: multi-choice questions (1p per question)

1. Which of the following is `assert` to be used for?

A. Flow control
B. User interaction
C. Error finding
D. Making Python code faster
E. Nothing, `assert` is not part of Python.

2. Which of the following words are specifically about working with files?

A. `open`
B. `for`
C. `exit`
D. `close`
E. `continue`

3. What is the result of `[ [ x[i] for x in [[1, 2, 3], [4, 5, 6]] ] for i in range(3)]`?

A. `[[0, 1, 2], [0, 1, 2]]`
B. `[]`
C. `[[1, 2, 3], [4, 5, 6]]`
D. `[[1, 4], [2, 5], [3, 6]]`
E. `[1, 2, 3, 4, 5, 6]`

4. What values are printed by the code on the right?

A. `'a'` and `'b'`

B. `[0, 1, 2]` and `[3, 4, 5]`

C. `'a'`, `[0, 1, 2]`, `'b'` and `[3, 4, 5]`

D. `{ 'a' : [0, 1, 2], 'b': [3, 4, 5] }`

E. None, you cannot have lists in dictionaries.

```
d = { 'a': [0, 1, 2], 'b': [3, 4, 5] }

for x in d:
    print(d[x])
```

5. Consider the code on the right, what is returned when evaluating the expression `f()`?

A. `10`

B. `11`

C. `9`

D. `None`

E. `8`

```
a = 2
b = 3
c = 4

def f(a = 1):
    b = 5
    return (a + b + c)
```

6. Given the code below, what is the value assigned to `result`?

A. `None`

B. `'[]""'`

C. `'howareyou?'`

D. `'how are you ?'`

E. `'howare?'`

```
mylist = [['how'], '[are]', 'you', '"?"']
result = ''

for x in mylist:
    out = ''
    for c in x:
        if c not in '[]"':
            out += c
    result += out
```

7. What is returned by the call `fcn(0)` given the definition below?

```
import random

def fcn(param):
    x = 0
    while random.random() > 0.5:
        if param > x:
            x += 1
            continue
        else:
            break
    return x
```

A. `0`

B. `1`

C. `2`

D. `10`

E. It cannot be predicted due to the randomness.

8. Given the function below, what is the result of `d([0, [1], [2, 3], [[4, 5]]])`?

A. `0`

B. `1`

C. `2`

D. `3`

E. `None`

```
def d(l):
    if type(l) == list:
        return 1 + max(map(d, l))
    else:
        return 0
```

Part B: coding problems (2p per problem)

Note: you may use `random` in this part.

9. One assignment in the exam from 2021-03-12 was about implementing a class `Dice` for six-sided dice. A possible solution is here:

```
class Dice:

    def roll_die(self):
        return random.randint(1,6)

    def roll_dice(self,n):
        sum = 0
        for i in range(n):
            sum += self.roll_die()
        return sum
```

Extend the code for `Dice` with a constructor that let us set an attribute `number_of_sides`, which determines how many sides the dice has, allowing for other values than 6. The allowed side counts are `4`, `6`, `8`, `12`, and `20`. The dice should be six-sided if the user does not give a specific number. If the user gives a number which is not allowed (i.e., not 4, 6, 8, 12, or 20) then an exception should be raised.

The methods `roll_die` and `roll_dice` should also be modified so that they work with the new dice sizes.

Examples: The code

```
d = Dice()
print(d.roll_die())
print(d.roll_dice(2))

t20 = Dice(20)
print(t20.roll_die())
print(t20.roll_dice(2))
```

may yield the following output:

```
4
7
16
27
```

Note: as dice rolls are random, you will probably get other output than in the example.

10. Inspired by the code for `Dice`, write a class `ArbDice` (for "arbitrary dice") where the user can also specify what values the sides should have. This is done by providing a list with as many elements as there are sides on the dice. A dictionary should associate side numbers (a number between `1` and `number_of_sides`) with corresponding side values. If the constructor is called without a list of values, then `1` to `number_of_sides` should be used as values in the dictionary.

The methods `roll_die` and `roll_dice` should be modified to work with arbitrary sides. A suitable exception should be raised if the user supplies a list with the wrong number of side values.

Example: The code

```
d4 = ArbDice(4,[10,20,30,40])
print(d4.roll_die())
print(d4.roll_dice(2))

d6 = ArbDice()
print(d6.roll_die())
print(d6.roll_dice(2))
```

may yield the following output:

```
30
80
1
9
```

11. Write a function `mask(s1, s2)` that takes two strings `s2` and `s2` and switches letters in `s1` for `*` in the order they appear in `s2`, returned in a new string.

    Note: lower or upper case must not make a difference.

    Example:

    ```
    [In] : print(mask('Hello, how are you Anders?', 'ehoaar'))
    [Out]: H*llo, **w *re you *nde*s?
    [In] : print(mask('Hello, how are you Anders?', 'H,HAAR?A'))
    [Out]: *ello* *ow *re you *nde*s*
    ```

    In the first example, note that `o` in `you` is not switched out since it was already used for `o` in `how`.

12. A simple method for text encryption is to write the characters of a text into a matrix with a set number of columns. All characters are filled in row by row, and then you read out the encrypted text column by column.

    Example: To encrypt `'Secret text'` with `3` columns, we create a matrix like this:

`'S'`	`'e'`	`'c'`
`'r'`	`'e'`	`'t'`
`' '`	`'t'`	`'e'`
`'x'`	`'t'`	

    The encrypted text is then `'Sr xeettcte'`.

    Write a function `matrix_encrypt(s,cols)` where `s` is a string and `cols` is a positive integer representing the number of columns for the matrix, and the encrypted text is returned.

    Tests:

    ```
    [In] : print(matrix_encrypt('Secret text',3))
    [Out]: Sr xeettcte
    [In] : print(matrix_encrypt('Secret text',2))
    [Out]: Sce eterttx
    [In] : print(matrix_encrypt('Secrettext',5))
    [Out]: Stetcerxet
    ```

    Note: if the length of `s` is not evenly divisible by `cols`, then there should not be spaces or other characters at the end of the encrypted string.

13. Write a function `matrix_decrypt(s,cols)` for matrix decryption. That is, given an encrypted string `s` and an integer `cols`, return the clear-text.

    Tests:

    ```
    [In] : print(matrix_decrypt('Sr xeettcte',3))
    [Out]: Secret text
    [In] : print(matrix_decrypt('Sce eterttx',2))
    [Out]: Secret text
    [In] : print(matrix_decrypt('Stetcerxet',5))
    [Out]: Secrettext
    ```

14. Write a higher-order function `encrypt_file(f,enc)` that takes a string `f` and an encryption function `enc`. Every line of the file `f.txt` should be encrypted with `enc` and the result should be written to `f_encrypted.txt`.

    Example: If the file `myfile.txt` contains

    ```
    secret texts
    writteninmy
    secret files
    ```

    then the expression `encrypt_file('myfile',lambda x: matrix_encrypt(x,3))` should put

    ```
    sr xeettctes
    wtnmrtiyien
    sr leefectis
    ```

    into the file `myfile_encrypted.txt`.