Programming for Data Science
DS2002

Wednesday, 11th of January 2023
9:00-13:00

ITE, Halmstad University
Contact: Slawomir Nowaczyk

Rules:

- The exam has **10 questions**, each corresponding to one of the topics covered in the course, and each worth **10 points** (however, the questions are **not** necessarily of **equal** difficulty).
- Please answer each question on a **separate** sheet of paper – and make sure each is clearly marked with your anonymous code and the question you are answering. An answer is expected to take 1-2 pages, depending on your writing style.
- Especially when writing code, use comments and explain your reasoning. It is easy to make small mistakes, but if we know what your intention was, we can better understand your actual knowledge and possibly give partial credit.
- The questions where you are asked to write code test two things: your algorithmic thinking, i.e., your ability to give step-by-step instructions to a computer; and your knowledge of key programming/Python concepts. Your solutions do not have to be perfectly correct in terms of syntax, and (of course!) they are not well-tested.
- In order to pass this exam, you need to get at least a **total of 50 points, as well as** at least **2.5 points in every question**!
- The final grade in the course is a combination (weighted average) of exam and lab results.

Tools allowed:

- Pen/pencil
- Eraser
- English dictionary (as long as it is a paper one and there are no notes in it)

No calculators or other electronic devices are allowed!

Question 1) Key Programming Concepts

A. Explain (briefly) the concept of variables and their scope. What is the difference between *global* and *local* variables? Give at least one example of a Python construct which creates a new scope for variables, and provide code demonstrating that two variables with the same name in different scopes can be used without affecting each other, as opposed to variables in the same scope.
B. What is the difference between *mutable* and *immutable* objects? Give three examples for each. What are the key pros and cons of them?
C. Explain the concept of efficiency in programming: why it is important, how to measure and compare efficiency, etc. Give at least one example of a programming problem that can be solved in different ways, with varying efficiency. Please limit your answer to half a page or so (maximum one page).
D. In general, program execution is “sequential”. Explain what does it mean and which programming constructs introduce exceptions to this general rule – and briefly describe what those exceptions are.
E. Code readability is paramount when programming. Give three examples of basic techniques that increase code readability, and explain each in one sentence.

Question 2) Variables

A. In Python, variables can hold values of different types. Write an example code where you define five different variables, each initialised with a value of a different type. Pick one of these types, and describe an operation that is suitable for that type, but not for any other of your examples.
B. What will be the result of running the following code? Explain your answer.

```
myVar = 5
otherVar = "aaa"
print("myVar = 7")
print("myVar =", myVar)
print(otherVar, "bbb", otherVar + "ccc", "otherVar" + "ccc")
thirdVariable = myVar + 1 + otherVariable
```

C. Write a function that takes two integers, the number of hours and the number of minutes, and converts them to seconds. For example, 2 hours and 11 minutes is 7860 seconds.
D. What will be the result of running the following code? Explain your answer.

```
newVar = [1,2,3]
def myFunction(arg = 5):
    print(newVar)
    newVar.append(arg)
myFunction()
newVar[0] = -1
myFunction(7)
newVar[1] = -5
myFunction(9)
```

Question 3) Conditionals

A. What will be the result of running the following code? Explain your answer.

```
a, b, c = 1, 2, 3
if a < b:
      print("I'm thinking")
      if b < c:
            print("a is largest")
      else:
            print("b is largest")
      print("I don't know")
      if a > 0:
            print("Positive")
else:
      print("I'm really confused")
print(a, b, c)
```

B. The goal is to create a function that returns the majority vote in a list (a majority vote is an element that occurs *more* than half the times in a list). The following code attempts to solve this task, however, it contains a number of errors (anything from mistakes that will make it not run at all, through returning the wrong result, all the way to just bad style). Find and fix as many of them as possible. For each mistake, explain what is wrong and why.

```
def MinorityVote(list):
    counter = {}
    while elem in list:
        counter.elem += 1
    half = len(list) * 2
    for elem in counter:
        if counter[elem] >= half:
            return counter
    return counter[0]
```

C. Create a function that takes a list of numbers as its only argument and returns a list with only the numbers that are less than or equal to zero. For example,
`selectNegative([1, 2, -5, 3, -7, 0, -1, 1])` → `returns [-5, -7, 0, -1]`

D. Explain briefly the `else` statement and why it is important/useful.

Question 4) Loops

A. Write a function that returns the elements on odd positions in a list, for example:
`odd_elements([1,2,3,4,5])` → `[2,4]`

B. Rewrite the following code using a "while" loop instead (explain your answer):

```
myString = "abcd"
for letter in myString:
      print(letter.upper())
```

C. Write a function that finds the *last* word in a given list that contains a given letter, for example:
`findLast('a', ['abc', 'bcd', 'xyz', 'zzz'])` → `returns "abc"`
`findLast('a', ['abc', 'bcd', 'aaa', 'bbb'])` → `returns "aaa"`

D. The “Atbash cipher” is an encryption method in which each letter of a word is replaced with its “mirror” letter in the alphabet, so A <=> Z; B <=> Y; C <=> X; etc. Fill in the blanks in the function below that takes a string and applies the Atbash cipher. For example, `atbash("Apple")` → `"Zkkov"`

```
letters = 'abcdefghijklmnopqrstuvwxyz'
def atbash(text):
    result = ""
    _____
        if letter.islower():
            index = letters.index(letter)
            code = letters[_____]
        elif letter.isupper():
            index = letters.index(letter.lower())
            code = letters[_____].upper()
        else:
            _____
        result += code
    _____
```

Question 5) Functions

A. Explain the concept of a function. What is the purpose of functions, and why are they so important in programming? How to define, and how to use one? Keep your explanations brief (preferably no more than 3-5 sentences per question).

B. Explain the differences between the three functions defined below. Give at least three examples of different usages that are correct for some and incorrect for others.

```
def myFunction1(a, b, c):
      return a + b + c
def myFunction2(a, b, c = 0):
      return a + b + c
def myFunction3(a, b):
      c = 0
      return a + b + c
```

C. Create a function that takes a string (containing only uppercase letters) as an argument and returns the Morse code equivalent. For example:

`encode_morse("TEST")` → `"- . ... -"`

Assume that the encoding of individual letters is given in a dictionary as follows:

```
char_to_dots = { 'A': '.-', 'B': '-...', 'C': '-.-.', 'D': '-..', ... }
```

D. What will be the result of running the following code (justify your answer):

```
myList = [1,2,3,4]
def myFunction(lst):
      lst.append(0)
      print(lst)
print(myList)
```

Question 6) Python Language

A. Explain the difference(s) between these two programs:

```
if myNr > 5:
    if elem == "a":
        print("yes")
    else:
        print("no")
print("done")
```

```
if myNr > 5:
    if elem == "a":
        print("yes")
else:
    print("no")
    print("done")
```

B. A function is defined as:

```
a = 5
def myFunc():
     return a + 1
```

What is the difference between:

```
aa = myFunc()
```

and:

```
bb = myFunc
```

Explain your answer. Give an example code that makes use of variables `aa` and `bb`.

C. Explain the following code:

```
[var.upper() for var in "abcdef" if var not in "aeoui"]
```

D. The following code demonstrates *name aliasing*:

```
myList = [1,2,3,4]
otherList = myList
thirdList = otherList.copy()
myList[0] = 100
print(myList, otherList, thirdList)
```

What will be the result of running this code? Motivate your answer.

E. Explain what `import` means in Python. Why is it useful? Give two examples of using `import`.

Question 7) Data Structures

A. Assume that you need a data structure to store user information (such as username, real name, address, and so on). The main task this data will be used for is retrieving all this information during the login process. Which is the best data structure to use: list, dictionary, string, numpy array, pandas dataframe, or something else? Ignore how each *user* is represented, the question is only about the *collection* of users. Justify your answer.

B. What are the key differences between string and list in Python? Since every string could be instead represented as a list of single characters, what is the purpose of "string"? Give at least two examples of situations where strings have benefits over lists, and at least two examples where lists have benefits over strings.

C. Write two versions of a function for checking whether value `5` exists in a collection stored in the variable `myCollection`. In one version, `myCollection` is a list, and in the other, `myCollection` is a dictionary. Is one solution better than the other, or do they each have their strengths and weaknesses in this situation? Justify your answer.

D. Explain key differences between Python list, numpy `array` and pandas `DataFrame`.

Question 8) Object-Oriented Programming

A. Define a simple class for storing information about an “employee”. Make sure it contains at least five relevant attributes. One of them should be “salary”. Add a method called “raise” that increases salary by a percentage given as an argument, with a default value of 10%.
B. For your class defined above, create two objects/instances. Store them in a list, and write a function which gives a salary raise to all employees in that list.
C. Describe the concept of *inheritance*, explain what it is useful for, and create a short code showcasing this concept.
D. Explain the importance of the `__init__` method. Give examples of three more such special methods, and describe their meaning.

Question 9) Recursion

A. Write a recursive function `power(n,k)` that calculates `n` to the power of `k` by repetitive multiplications. In other words, $n^k = n * n^{k-1}$ (for most values of k).
B. Write two functions, one recursive and one using a `for` loop, for calculating a sum of all elements in a list given as an argument.
C. Write a recursive function that creates all permutations of a given list.
A permutation is a specific ordering of all elements in the list. For example, there are six permutations for [A, B, C]: ABC, ACB, BAC, BCA, CAB, and CBA. One way of creating all the permutations of a given list is to take the first element and add all the permutations of the rest, then take the second element and all the permutations of the rest, and so on. For the [A, B, C] example, it would be:
[A] + permutations([B,C])
[B] + permutations([A,C])
[C] + permutations([A,B])
(Hint: this is a difficult question, so – especially here – please explain your reasoning and train of thought. We don’t expect most students to be able to solve it perfectly; thus, it is essential that we can understand from your answer how close you were to the correct solution).

Question 10) Libraries

A. Write a code to create a two-dimensional numpy array with values from 1 to 6. Print all values that are larger than 3. Calculate the sum of all elements, the sum of all rows, and the sum of all columns.
B. Write a code to create a pandas dataframe from a CSV file "iris.csv". Print the size of the data; list the column names; print the data from columns *sepal_length* and *sepal_width* only (skipping all the others); calculate the mean value of each column; and print only the data where *sepal_length* is larger than 5.
C. Assume that you have iris data stored in a padas dataframe. Draw three different plots showing the data in different ways.
D. Choose three modules from Python standard library and briefly describe (in 2-3 sentences) what their purpose is and give examples of functionality that they offer.