PGR107 - Python Programming

Kristiania University College

Final Exam

Academic contact during examination: Hadi Zahmatkesh, hadi.zahmatkesh@oslomet.no, +47 93255316

Technical contact during examination: eksamen@kristiania.no

Exam type: Written home examination in groups (1-5 students)

Support materials: All support materials are allowed

Plagiarism control: We expect your own independent work. Any copies from the internet or other groups will give you an automatic **FAIL** for every involved party (the giver and the taker).

Grading scale: Norwegian grading system using the graded scale A - F where A is the best grade, E is the lowest pass grade and F is fail.

- **A**: Excellent and comprehensive understanding of the topics
- **B**: Very good understanding of the topics
- **C**: Good understanding of the topics
- **D**: Satisfactory understanding of the topics but with significant shortcoming
- **E**: Meets the minimum understanding of the topics
- **F**: Fails to meet the minimum academic criteria

Learning outcomes (knowledge & skills):

The student

- understands problem solving using programming.
- understands the principles of object-oriented programming.
- has overall knowledge of general properties of python programming language such as program flow, loops, and choices.
- has knowledge of Python programming using data structures, functions, classes, objects, and modules.

Submit complete Python codes (.py files) for Questions 1-5.

Question 1 (20 points)

Sensitive information is often removed, or redacted, from documents before they are released to the public. When the documents are released, it is common for the redacted text to be replaced with black bars. In this question, you will write a program that redacts all occurrences of sensitive words in a text file by replacing them with asterisks. Your program should redact sensitive words wherever they occur, even if they occur in the middle of another word. The list of sensitive words will be provided in a separate text file. Save the redacted version of the original text in a new file. The names of the original text file, sensitive words file, and redacted file will all be provided by the user.

*** Note that your program should perform error checking.

*** You may find the replace method for string helpful for this question.

Question 2 (20 points)

Write a class ***Bug*** that models a bug moving along a horizontal line. The bug moves either to the right or left. Initially, the bug moves to the right, but it can turn to change its direction. In each move, its position changes by one unit in the current direction. Provide a constructor

```
def __init__(self, initialPosition)
```

and methods

- def turn (self)
- def move (self)
- def getPosition (self)

Sample usage:

```
bugsy = Bug (10)
bugsy.move () # Now the position is 11
bugsy.turn ()
bugsy.move () # Now the position is 10
```

Your driver program should construct a bug, make it move and turn a few times, and print the actual and expected positions.

Question 3 (20 points)

A string is a palindrome if it is identical forward and backward. For example, “anna”, “civic”, “level” and “hannah” are all examples of palindromic words. Write a function that checks whether or not a string entered by the user is a palindrome. Display the result, including a meaningful output message.

Question 4 (20 points)

Write a program that asks a user to type in two strings and that prints

a. the characters that occur in both strings.
b. the characters that occur in one string but not the other.
c. the letters that don't occur in either string.

Write **three different** functions to do the tasks above.

Sample Run:

Enter the first string: Python

Enter the second string: Programming

Shared characters:

P, n, o

Unique characters

a, g, h, i, m, r, t, y

Non-occuring alphabet letters:

b, c, d, e, f, j, k, l, p, q, s, u, v, w, x, z

Question 5 (20 points)

In this question, you will develop a "Word Guessing Game" using python. This game is about guessing letters (A-Z) to form the word. The word is randomly chosen from a text file containing several words. Your program should read from the text file and choose a word. If the player guesses the right letter that is within the word, the letter appears at its correct positions. The user can get the chance to guess the correct word until ??? turns, then the game is over (??? is equal to the number of characters in the word).

Note that the maximum number of wrong guesses is equal to the length of the word. For example, if the word is 8 characters long, then the user can only have 8 wrong guesses.

Here is a sample run:

```
The word you need to guess has '4' characters.

You have now 4 guesses.

_ _ _ _

Guess a character: a

Sorry that letter is not in the word.

_ _ _ _

you have 3 guess(es) left.

-------------------------

Guess a character: o

_ _ o _

you have 3 guess(es) left.

-------------------------

Guess a character: s

s _ o _

you have 3 guess(es) left.

-------------------------

Guess a character: n

s n o _

you have 3 guess(es) left.

-------------------------


Guess a character: w

You found the word --> "snow"
Congratulations! you won
```

Here is another sample run:

```
The word you need to guess has '4' characters.

You have now 4 guesses.

_ _ _ _

Guess a character: a

Sorry that letter is not in the word.

_ _ _ _

you have 3 guess(es) left.

-------------------------

Guess a character: b

Sorry that letter is not in the word.

_ _ _ _

you have 2 guess(es) left.

-------------------------

Guess a character: c

Sorry that letter is not in the word.

_ _ _ _

you have 1 guess(es) left.

-------------------------

Guess a character: d

Sorry that letter is not in the word.

_ _ _ _

you have 0 guess(es) left.

-------------------------

Sorry! you lost.
The word is --> "tree"
```