CSCI 2040: Introduction to Python **2022-2023 Term 2**

Lab Assignment 1

Instructor: King Tin Lam Due: 23:59 on Thursday, Febr. 09

Notes

1. Although the lab room provides desktop PCs, bringing your own laptop computer with Anaconda[1] (or Miniconda[2]) and VSCode[3] (or Pycharm[4]) installed is highly recommended. While attendance won't be taken, we highly encourage you to attend the lab session for peer learning and better efficiency. You may discuss the questions with classmates and seek help from our TAs more readily.
2. For those of you using the Windows PC in SHB 924A (NOT recommended) with your CSDOMAIN account[5], please log in and open "Computer" on the desktop to check if an "S:" drive is there. If not, then you need to click "Map network drive", use "S:" for the drive letter, fill in the path "`\\ntsvr6\userapps`" and click "Finish". Then open the "S:" drive, open the **Python3** folder, and click the "IDLE" shortcut to start doing the lab exercises.
3. The programming assignments will be **graded by Python scripts.** If your script does not pass the tests of our grading scripts, you cannot get full marks.
4. The test scripts we provide can be used **on MacOS, Windows and Linux** for your own test. You are able to test the input and output format for an exercise by the corresponding testing script. For example, by running `test_p1.py`, you can test `p1.py` for exercise 1.
5. To use our testing scripts, for Linux or MacOS users, please install package `pexpect` by "`pip install pexpect`"; For Windows users, please install package `wexpect` by "`pip install wexpect`" first. Then put your scripts and the testing scripts under the same file folder and run the test scripts. If you cannot run the test scripts, one way you can try is to change the line "`child = exp.spawn('python3 p1.py')`" to "`child = exp.spawn('python p1.py')`" in each script.
6. Note that the grading scripts will be DIFFERENT from the provided testing scripts. Passing the test scripts we have provided doesn't guarantee full marks for your solution as our grading scripts will test it against more cases.
7. You may assume that all user inputs are always valid and your program for each question need not include code to validate them unless you are required to do so. You may assume that all the corner cases we have not mentioned in this document will not appear in the

[1] An open data science platform powered by Python. `https://www.anaconda.com/products/distribution`

[2] A free minimal installer for conda. `https://docs.conda.io/en/latest/miniconda.html`

[3] A lightweight but powerful source code editor. `https://code.visualstudio.com/`

[4] A powerful Python IDE. `https://www.jetbrains.com/pycharm/download/`

[5] A non-CSE student should ask the TA for a CSDOMAIN account.

hidden grading cases, so you do not have to worry too much about wrong inputs unless you are required to do so.

8. Please follow closely the format of the sample output for each question. Your program should produce **exactly** the same output as the sample (same text, symbols, letter case, spacing, number format, etc.). The output for each question should end with a single newline character, which has been provided by the `print()` function by default.

Exercise 1: Length of a string (20 marks)

The `input()` function is a build-in function that reads a string from the user via console input. The calling syntax of the function is: `input(['prompt'])`. For example, suppose that `x = input('type a number:')` is executed, it shows the message "`type a number:`" and waits for the user to enter some value and hit the Enter key. If the user types "`one`" followed by Enter, the variable x will store the string "`one`".

Write a python script named `p1.py` that prompts the user to enter a string, and prints the length of it accordingly. Below are some sample runs.

Sample 1:

```
Please input a string: csci2040
The length of string "csci2040" is 8
```

Sample 2:

```
Please input a string: python
The length of string "python" is 6
```

Sample 3:

```
Please input a string: I love python!
The length of string "I love python!" is 14
```

Exercise 2: The largest and smallest number in a list (20 marks)

Write a python script named `p2.py` that reads **five** integers from the user's input. You can use `int()` to convert a string to the corresponding integer data type. After that, print the largest number and the smallest number in the list.

Note: You can assume the input can be converted to an integer by `int()` successfully, so there is no need to check the validty of the input.

A sample run is given as follows,

```
Please input the 1st integer: 2
Please input the 2nd integer: 9
```

```
Please input the 3rd integer: 1
Please input the 4th integer: 8
Please input the 5th integer: 13
The largest number is 13 and the smallest number is 1.
```

Exercise 3: Right triangle (20 marks)

For a right triangle, its lengths of three sides x, y, and z satisfy $x^2 + y^2 = z^2$ (Suppose z is the largest).

Please write a python script named `p3.py` that asks the user to input 3 integers. Then the script checks whether the three integers can be the lengths of a right triangle. If yes, the script prints "`The three integers can from a right triangle.`"; otherwise, the script prints "`Sorry. The three integers cannot from a right triangle.`".

Sample 1:

```
Please input three integers: 3 4 5
The three integers can form a right triangle.
```

Sample 2:

```
Please input three integers: 6 8 10
The three integers can form a right triangle.
```

Sample 1:

```
Please input three integers: 3 4 4
Sorry. The three integers cannot form a right triangle.
```

Exercise 4: Isogram word (20 marks)

An isogram is a word that has no duplicate letters. For example, "isogram" is an isogram, while "letter" is not because there are 2 "e"s in "letter".

Write a python script named `p4.py` that reads a string from the user's input. Then the script checks whether the string is an isogram, and if so, the script prints the line "`The input is an isogram. Program ends!`" and ends the program. Otherwise the script prints the line "`Your input is not an isogram!`" and keeps asking the user to input a string by "`Please input a string:`" in the next line.

Note: Here we consider **case-insensitive** letters. We treat the lowercase letter and uppercase letter the same. For example, "lEtter" is not an isogram because we treat "`e`" and "`E`" the same. And we assume the input is exactly one word and you don't need to check the validness.

Below are some sample runs.

Sample 1:

```
Please input a string: python
Your input is an isogram. Program ends!
```

Sample 2:

```
Please input a string: PRogram
Your input is not an isogram!
Please input a string: python
Your input is an isogram. Program ends!
```

Sample 3:

```
Please input a string: student
Your input is not an isogram!
Please input a string: Csci2040
Your input is not an isogram!
Please input a string: Program
Your input is not an isogram!
Please input a string: python
Your input is an isogram. Program ends!
```

Exercise 5: Price calculation **(20 marks)**

Suppose there is a store having a mega sale. If the price of one item is below 100 (contains 100), it will be 20% off; if the price of one item is between 100 and 200 (contains 200), it will be 30% off; if the price of one item is between 200 and 500 (contains 500), it will be 40% off; if the price of one item is above 500, it will be 50% off. And if customers buy 3 or more items, there will be an extra 20% off based on the discount price above.

For example, if we buy three items with price 120, 300 and 600, the final price will be $(120 * 0.7 + 300 * 0.6 + 600 * 0.5) * 0.8 = 451.20$

Write a python script and named `p5.py` that help customers calculate the final price. First the script will ask the customer to input the price of one item by "`Please input the price:`", then print the current price by the line "`The current total price is xx.`" (xx should be replced by a number) and ask the customer to input the price of the next item by "`Please input the price:`" in the next line. If the customer inputs -1, the script will print "`The final total price is xx. Goodbye!`" and end the program. If the customer inputs a price, the script will print the current total price and keep asking the customer to input price as described above.

Note: The output amount should be rounded to two decimal places. You can use "`0:.2f.format(a)`" or f-string "`{a:.02f}`" if you want to output 2 decimal points of `a`.

Please follow the format of the sample runs below.

Sample 1:

```
Please input the price: 100
The current total price is 80.00.
Please input the price: -1
The final total price is 80.00. Goodbye!
```

Sample 2:

```
Please input the price: 120
The current total price is 84.00.
Please input the price: 200
The current total price is 224.00.
Please input the price: -1
The final total price is 224.00. Goodbye!
```

Sample 3:

```
Please input the price: 120
The current total price is 84.00.
Please input the price: 300
The current total price is 264.00.
Please input the price: 600
The current total price is 451.20.
Please input the price: -1
The final total price is 451.20. Goodbye!
```

Submission rules

1. Please name the script files with the **exact** name specified in this assignment and test all your scripts. Any script that has any syntax error will not be marked.
2. Please pack all your script files as a single archive named as

   `<student-id>_lab1.zip`

   For example, `11550123451_lab1.zip`, i.e., just replace `<student-id>` with your own student IDs.
3. Upload the zip file to your blackboard (`https://blackboard.cuhk.edu.hk`),
   - Subject of your file should be `<student-id>_lab1`.
   - No later than 23:59 on Thursday, Febr. 09

4. You may discuss how to solve the problems at an idea level but never share solution code (even partially) among classmates. Both the students sending code to others or copying code from others are regarded as committed plagiarism.
5. Marks will be deducted if you do not follow the submission rules. Anyone caught plagiarizing would get 0 score!