Lab 02 - Creating a CS1301 Homework

CS 1301 - Intro to Computing - Fall 2021

Important

- Due Date: **Tuesday, November 23rd, 11:59 PM**.
- You may work with up to one (1) other person on this assignment. If you choose to do so, you and your partner's name will both be attached to the same submission. **Please read the Lab Rules and Submission Process sections.**
- Resources:
  - TA Helpdesk
  - Email TAs or use Ed Discussion

Purpose

In this lab, you will see what it's like to be a TA for this course. You will be writing a few homework-style problems about any of the topics taught in this class so far. **The 5 best problems (picked by the TAs) will be featured on the HW11 assignment!**

This assignment is very different from your typical homeworks, so read the entire document before you begin.

Document Overview

Explanation

One of our responsibilities as TAs is to create homework assignments about the various topics in this course. In this assignment, you will be stepping into our world and writing homework problems about any of the topics you have encountered in the class so far. Feel free to add unique themes and stories to connect all your problems together. **This assignment will be completed entirely on Gradescope.**

Lab Rules

In this assignment, you may either work individually or with **one** other student currently in this class **with the same professor as you**. Either way, this assignment involves writing 4 homework problems (and 1 additional problem for extra credit)

Each homework problem you write must follow a specific structure (explained in the **How to Write a Homework Problem** section of this document). Additionally, you must provide a solution function and sample test case to your problem in a solution file (more on this later).

All submissions must be unique. While you and your partner will be submitting the same assignment (more details in the Submission Process section), if your submission is identical to another group or individual's submission, this will be considered as a case of plagiarism. Additionally, copying homework problems from this semester or previous semesters verbatim will be considered as plagiarism. These submissions will be given a 0 and will be reported to the Office of Student Integrity.

How to Write a Homework Problem

Each homework problem will ask you to write a function. All problems have **6 important parts**, which we have broken down for you below. Each part will be included in your submission on Gradescope as part of both the problem description and a solution file, which are explained later.

Make sure that your problem is not trivial. All trivial problems will receive 0 points. Read the Trivial Problem section of this document for more details.

Function Name

This is the name of the function that the student will have to write. The function name must be valid in Python, and should be somewhat descriptive of the problem at hand. For example, a problem about counting apples should be called something like `countApples` instead of something non-descriptive like `hwProblem1`.

Function Params and Types

Here, you will provide a description of what your function will take in. You can describe each param by first providing a descriptive variable name (remember, this must be a valid variable name in Python), followed by the type of the param. For example, suppose we have a hypothetical function that takes in two parameters. The first parameter is a list of fruits (each fruit is a string). The next parameter is the price of an apple, as an integer. Below are a few ways we can describe our function params:

> fruitList (list), applePrice (int)
> listOfFruit (list), priceOfApple (int)

List the params in the order that they will be fed into the function header. Make sure to separate each param by a comma, and put the type of the param in parenthesis next to it.

Function Return Value and Type

This is a description of what value the function will return, along with the type of the value. You can either provide a variable name or describe the output. For example, if our function returns a price (as a float), our return value could be:

> totalPrice (float)
> price (float)
> total price of apples (float)
> apple price (float)

Function Description

This is the most important portion of the problem. Here, you will describe the problem. Give some background/context, explain what is in the function params, and explain what must be returned. Be clear on what values will be passed in, along with what values will never be passed in (for example, if only positive integers are valid inputs). Also, be clear about the expected output for any special cases. Good problems provide a story to make the problem understandable, and the best problems have stories that link together to create a connection between problems.

Solution Code

For each problem that you write, you will need to write a solution function in Python and add it to a solution file for that particular problem. Essentially, you need to write the function that would work as a solution to the problem you described. Be sure that your function header exactly matches with the function name and function param names you gave above.

Test Case

To demonstrate how your function works, provide an example testcase showing the input and an expected output. First, show the function call with all the necessary arguments. On the following line, show the expected output. For example, let's say we have a function called `totalApplePrice` that takes in a list of strings and the price of an apple (an integer), counts each `"apple"`, and returns the total price of all the apples. Below is an example test case:

```
>>> totalApplePrice(['melon', 'apple', 'apple', 'berry'], 3)
6
```

You only need to provide one test case for each problem. This test case will be included in your solution file (see below)

Sample Homework Problem

Each homework problem in Gradescope has 2 portions: a problem description textbox, and a solution file upload. Below is an example of a single completed homework problem. We expect each of your problems to be written in a similar format.

Problem Description

The problem description includes all required portions of a standard homework problem. Here's a description for a sample problem that you would type in Gradescope:

> You're playing a video game on your computer where you can move a character to the left using the "<" key and to the right using the ">" key. You're given a string that represents a series of keystrokes (containing only "<" and ">") that you played to move your character. Write a function called moveCharacter that can take this string and return whether or not your character will end up in the same place as where it started before any of the keystrokes are applied. Return True if the character will be in the same place, otherwise return False.

Solution File

To create a solution file, download the `solution_template.py` file from Canvas. Rename the file to be `HWPX.py`, where `X` is the homework problem number. For example, the solution file for Homework Problem #2 should be named `HWP2.py`. Then, you can fill out the template with a solution function and test cases. Upload the completed file to the appropriate question in Gradescope. Below is a sample solution file for the problem described above:

```
#### HOMEWORK PROBLEM 2

"""
Function Name: moveCharacter()
Parameters: keystrokes (str)
Returns: whether character is in same place (boolean)
"""

#### SOLUTION:
def moveCharacter(keystrokes):
    movement = 0
    for key in keystrokes:
        if key == '>':
            movement += 1
        elif key == '<':
            movement -= 1
```

```
    return movement == 0

#### TEST CASE(S):
moveCharacter("<><><<<<<><><<")
# False
moveCharacter("<<<<<<>>>>>>")
# True
```

Trivial Problems

A trivial problem is a question that poses little to no challenge for the student. These problems take no effort to solve or write. Any problem considered trivial by the grading TA will be given 0 points. If you have a question about whether or not your problem is trivial, feel free to ask a TA via email or Ed Discussion (in a private post). Below are examples of problems that are trivial:

> Write a function that takes in an integer and return `True` if it's even, or `False` if it's not.

> Write a function that takes in two integers, and return the product of the two.

> Write a function that takes in a list of strings, and a target string. Return the index of the target string in the list. If the list does not contain the target string, return -1.

> Write a function that takes in a string, and prints out this string three times.

As you can see, all the functions above are extremely simple to implement, and any problem like them would receive 0 points.

Submission Process

Lab02 will be **completed entirely on Gradescope**. It is available in Gradescope as an assignment titled **Lab02**. Carefully read all instructions and fill out the necessary fields in the assignment to complete it. **You may edit your submission as many times as you want until the submission deadline.**

If you choose to work with a partner, you will need to add that partner to your submission. To do so, find the Lab02 assignment on Gradescope. Click the **Group Members** button in the bottom right, go to the **Add Student** section, look up your partner's name, and finish by clicking the **Add** button. This one submission will now count for both you and your partner. Your partner can be in either section of the class. Once you have successfully added your partner, the same submission will apply to both of you. **Remember, you may only work with a student who has the same professor as you.** Otherwise, Gradescope cannot add your partner to the submission.

Every time you resubmit the assignment or make any changes, please make sure that both you and your partner are still added to the same submission.

Extra Credit

You can earn up to 25 extra credit points on this assignment by writing one extra problem. Make sure to write this problem in the **Homework Problem #5 - Extra Credit** section on the Gradescope assignment.

Grading Rubric

This assignment has a total of 100 points available (excluding extra credit). Each homework problem will be worth 25 points. Submissions must have at least 4 problems, resulting in a total of 100 points. **Remember that every trivial problem will receive 0 points.** Below is a rubric for each homework problem:

Item Description	Points (Partner)
Descriptive function name	1
Descriptive function params and types	2
Descriptive function return values and type	2
Function description	5
Sample test case(s) in `HWPX.py`	5
Solution code for problem in `HWPX.py`	10
Total	**25**

Final Total (with extra credit): 125/100 points

Submissions that demonstrate the following will receive a zero:

- Submission shows evidence of large portions of copying from any previous homeworks.
- Submission shows evidence of copying from other submissions of students in the class (excluding the same submission between partners)
- Submissions that contain any inappropriate content or vulgarity

Appendix A: Topic List

For inspiration, below is a list of some of the topics we have learned in this class so far. Your problems may be about any of the topics below, or you can even combine multiple topics in the same problem.

- Boolean values
- Logical operators (`and` , `or` , `not`)
- Conditional statements (`if` , `elif` , `else`)
- Nested conditional statements
- `for` loops and `while` loops
- Nested loops
- `break` / `continue` / `pass`
- Range
- Compound data types
- String slicing and indexing
- Looping through a string
- String comparison
- Tuples
- Lists (creating lists and accessing items)
- List slicing and indexing
- Appending/deleting on lists
- `in` and `not in` operators
- Aliasing and cloning
- List methods: `sort` , `insert` , `index` , `reverse` , `remove`
- Nested Lists
- Reading from files
- Writing/appending to files
- Enumerate
- `try` / `except` / `finally`
- Dictionaries
- APIs
- Recursion
- Object Oriented Programming