420-LCU-05 Programming in Python - Assignment 1

February 5, 2019

Due Date: February 19th, 2018 at 23:59

This is our first regular assignment in the course. To start, we'll mention some general requirements that will apply to all of these assignments:

1. **Identification section** Do this for *every* Python file in every assignment in this course. This section must be either in a comment, with a '#' preceding each line, or enclosed within triple quotes ("'). The grader and I need this section for the *accurate processing of your assignment*. Assignments missing this may lose up to 5% of the total mark.

   Example:

   ```
   """
   Your Name
   420-LCU Computer Programming, Section #
   Monday, September 17th
   S. Hilal, instructor
   Assignment 1
   """
   ```

   Substitute your name, section, and the correct date for the appropriate fields!

2. Always include additional comments with your code. These need not explain every individual line of your program, but consider using comments for the following situations:

   - A brief explanation of a particular variable's purpose, included on the first line where the variable is defined, e.g.:

     ```
     hi = 100 # Define the upper limit of the range.
     ```

   - A note mentioning any website or person you may have consulted with to help with the assignment.
   - A comment describing any constant value that appears in your code.

3. Your submission for assignment will include 2 Python files, which have the extension `.py`. Before submission, these files must be combined into a single ZIP archive file (extension `.zip`). If you do not know how to create a ZIP file, I will demonstrate this in lab.

4. Be sure to respect other instructions specified in the assignment. Part of each assignment is to correctly follow the instructions as closely as possible.

5. Late assignments are accepted up to 1 week from deadline. But late penalty will be applied.

Exercise 1

We will write a program to play a simple game, guessing a secret **integer** the user has selected between 1 and 100. You will be choosing the secret number and your computer program will be guessing. Here is an example transcript of the program, where the user has chosen the number 17:

```
Please think of a number between 1 and 100.
Is your secret number 50?
Enter 'h' if my guess is too high, 'l' if too low, or 'c' if I am correct: h
Is your secret number 25?
Enter 'h' if my guess is too high, 'l' if too low, or 'c' if I am correct: h
Is your secret number 13?
Enter 'h' if my guess is too high, 'l' if too low, or 'c' if I am correct: l
Is your secret number 19?
Enter 'h' if my guess is too high, 'l' if too low, or 'c' if I am correct: h
Is your secret number 16?
Enter 'h' if my guess is too high, 'l' if too low, or 'c' if I am correct: l
Is your secret number 17?
Enter 'h' if my guess is too high, 'l' if too low, or 'c' if I am correct: c
Game over, your secret number was: 17
```

Some hints for your program:

- Your initial end points will be 1 and 100.
- Your program will use *bisection search* to guess the right answer quickly. What will be the first guess each time? How will you calculate subsequent guesses?
- Structure the program to handle the case when a user types in a value other than `l`, `h`, or `c`. Print an error message and ask for input again.
- The program can be written in 15-20 lines of code or so.
- Your program should use the built-in functions `print()`, `input()`, and `int()`. You *may* use others, but they are not necessary.
- You should need at least one `while` loop and one `if` statement.
- Make your program smart and exclude wrong guesses from the range.
- Your solution should mimic the example transcript above as closely as possible.
- Experiment with the program, trying many different ranges for the guessed number. For example, modify the program to allow the guessed number to be in the range 1 to 1024. What is the largest number of guesses the program ever needs to make? How does this relate to bisection search?

Exercise 2[1]

Your have a cell phone and after a month of use, you are trying to decide which price plan is the best for your usage pattern. There are 3 plan options available A, B, or C.

- All plans have the same base price of $10 per month
- Free minutes are for day time only. Evening and week-end minutes are always at a cost.
- each plan has different costs for daytime minutes, evening minutes and weekend minutes.

Plan	**Costs**		
	Daytime	Evening	Weekend
A	100 free minutes, then 15¢/minute	20¢/minute	25¢/minute
B	200 free minutes, then 20¢/minute	25¢/minute	30¢/minute
C	250 free minutes, then 30¢/minute	35¢/minute	40¢/minute

Write a program that will ask the user to input the number of **each type of minutes** and output the cheapest plan for this usage pattern, using the format shown below. The input will be in the order of daytime minutes, evening minutes and weekend minutes. Find cheapest price and print an appropriate message.

Example 1

```
Number of daytime minutes? 254
Number of evening minutes? 10
Number of weekend minutes? 60
Plan A costs 50.10
Plan B costs 41.30
Plan C costs 38.70
choose Plan C.
```

[1] Adapted from the Canadian Computer Competition, 2005.

Example 2

```
Number of daytime minutes? 162
Number of evening minutes? 61
Number of weekend minutes? 66
Plan A costs 48.0
Plan B costs 45.05
Plan C costs 57.75
Choose Plan B.
```

Example 3

```
Number of daytime minutes? 260
Number of evening minutes? 20
Number of weekend minutes? 70
Plan A costs 55.5
Plan B costs 48.0
Plan C costs 48.0
Choose Plan B.  # Note that you choose first cheaper plan.
```

Note that if the result is an even number of dollars, it will probably print as shown in Example 3, with a single zero after the decimal point. This is correct behavior for now.

One important point: Be sure that you define symbols for the constant values used to represent each calling plan. For example, to define the first calling plan, your code should contain a block similar to the following:

```
A_FREE = 100   # Number of free daytime minutes per month.
A_DAYTIME = 15 # Cost of additional daytime minutes, in cents.
A_EVENING = 20 # Cost of evening minutes, in cents.
A_WEEKEND = 25 # Cost of weekend minutes, in cents.
```

It is a common stylistic practice to use ALL CAPITAL LETTERS for "constants" or "parameters" such as these. It is useful to give names to these values to help make your program both more self-explanatory and easier to modify should details change in the future. *Please follow the ALL CAPS convention for constants and parameters in this and all other assignments!*.

Some hints for your program:

- This program may be slightly longer than Exercise 1, but it isn't really any more difficult.
- You will need several `if` statements to implement this correctly.
- Again, try to follow the example output as closely as possible.

Submitting your work

When you have finished both sections, combine the two python files into a single ZIP file and upload that to Omnivox.