CPS109 Project

Putting it all together

Learning Objectives:

In this project, you will practice using every Python construct we've seen so far to solve a problem that you define yourself. These include, but need not be limited to:

1. Conditions and decision making using if/else statements.
2. Working with sequences, such as lists, tuples, and strings.
3. Sequence iteration using for loops, general iteration using wile loops.
4. Importing namespaces, using their functions, writing your own functions.
5. Performing file I/O, reading input from a file, writing output to a file.

Introduction:

Every year enrolment in CPS109 gets larger, and students come from a wider range of backgrounds – not just computer science. We in the computer science department would like to see this trend continue, since we believe that computational thinking and programming are fundamental skills that all university students should obtain. This year we have students from the following programs: Computer Science, Science, Math, Arts, Engineering, Financial Math, Business, RTA, Biomedical Science, Graphic Communication, International Economics, Creative Industries, Hospitality, Journalism, Media Production, and Nutrition.

Clearly there are applications for programming in all these areas, and each of you may have a problem in mind to **which you would like to apply** your new Python programming skills. The problem might be from everyday life, like how long it will take to pay off a loan given the interest rate and the payment schedule, or from your intended profession, or from your pastime or your other courses. In this assignment, we ask you to choose or invent your own problem and solve it.

What to do:

1) Come up with a problem you wish to solve and describe it in a paragraph or two. Your description should be technical and precise – no vague handwaving. The clearer the problem description, the easier it will be to solve in Python.

(Those of you in the computer science program will take a software engineering course in 2nd year and learn all about the art of defining requirements and specifications)

2) Write a Python program called **cps109_a1.py** that solves the problem you defined in part 1. At minimum, your program must meet the following requirements. Feel free to exceed these requirements – this is only a lower bound:

a. In a comment at the top of your source file you should paste the problem description you wrote in part 1.

b. At least 20 lines of ***meaningful*** comments (not including the problem description) and 60 lines of code (not including whitespace).

   Note: The 60-line requirement is to prevent your program from being too simple/trivial. Do not write inefficient code or needlessly expand your syntax for the sake of hitting 60 lines. You're all well versed in padding high school essays I'm sure, but do not extend this habit to programming. **Writing succinct, readable, efficient code is your prime directive.** If you're having trouble hitting 60 lines, you should expand the scope of your problem.

c. Your program must demonstrate each of the following Python elements at least once:

   - Variable declarations and assignments
   - Arithmetic expressions (can include function calls)
   - Use of if/elif/else statement
   - Use of sequence types and their operations (strings, lists, tuples)
   - Use of a for loop for sequence iteration (a list comprehension works here also)
   - Use of a while loop for general iteration (driven by a condition)
   - A user-defined function for solving some meaningful subproblem.
   - Print statements for displaying output (user input is allowed, but not required)
   - Read input from a file OR write output to a file.

d. Finally, your program must of course solve the problem you defined in part 1).

3) Run your program and take a **clear screenshot** of your IDE's terminal, showing the output of your program. This image should have the filename **cps109_a1_output.xxx**, where xxx is any common digital format extension (png, jpg, etc.). Additionally, include a copy of the file your program uses as input, and/or the file your program creates as output.

Marking scheme:

Out of 30 marks:

3 marks	Clear and concrete problem description
2 marks	20 lines of (meaningful) comments
6 marks	60 lines of (meaningful) code
1 mark	Use of variables and arithmetic expressions
2 marks	Use of if/elif/else statements (with conditions)
2 marks	Use of sequence iteration (for loop, comprehension, etc.)
2 marks	Use of general iteration (while loop)
3 marks	Use of at least one user-defined function
2 marks	Use of print statements to display program output
4 marks	Use of files for input and/or output
3 marks	Program adequately solves the described problem

What to Submit:

This project will be submitted on D2L, not Gradescope. The submission drop box can be found under Assessment->Assignments.

Submit the following items:

- The Python source file containing your code (cps109_a1.py).
- The screenshot of your IDE showing the code alongside the output in the terminal.
- Any input text file(s) required by your program.
- Any output text file(s) produced when your program is run.

Plagiarism Disclaimer:

You are to work **alone** when writing your code. Students may not copy problem descriptions, and if two students independently come up with the same or similar problems, their programs must be sufficiently unique. You can discuss general ideas with your classmates, but you cannot copy code or develop code together nor take code from the web (this includes asking ChatGPT or its derivatives to solve your problem for you).

The Department of Computer Science takes the act of plagiarism very seriously. Those caught plagiarizing (both originators and copiers) will be sanctioned. Please see TMU's Policy 60 for possible penalties and consequences. If you are unsure what constitutes plagiarism, please see your instructor.