Computational Thinking and Program Design

Class Project: Teaching computer to solve the MCGW problem (version 1.0)

(Due at 23:59 on 15 January 2023)

Rocky K. C. Chang

3 December 2022

Objective:
You will have experienced the entire problem-solving process through designing and implementing a program to solve the Man-Cabbage-Goat-Wolf (MCGW) problem.

The Project:
In this project you will design and implement a program to solve the MCGW problem. You must apply and document the four-step process: data abstraction, algorithm design, program design and implementation. In the program design, you must modularize your code by identifying the functions that you need and their signatures. Unlike what we have done in the classes, **you must use a new data abstraction for this project which models each state by the positions of the cabbage, goat and wolf, without the man's position.**

Your program will print on the screen exactly the output in Figure 1. As you know, there are two equally good solutions to this problem. But your solution must be the one in Figure 1.

```
The solution to the MCGW problem consists of 7 rides:
-----------------------------------------------------
   East: man, cabbage, goat, wolf          West:
1. The man takes the goat from the East to the West.
   -->
   East: cabbage, wolf                     West: man, goat
2. The man takes only himself from the West to the East.
   -->
   East: man, cabbage, wolf                West: goat
3. The man takes the cabbage from the East to the West.
   -->
   East: wolf                              West: man, cabbage, goat
4. The man takes the goat from the West to the East.
   -->
   East: man, goat, wolf                   West: cabbage
5. The man takes the wolf from the East to the West.
   -->
   East: goat                              West: man, cabbage, wolf
6. The man takes only himself from the West to the East.
   -->
   East: man, goat                         West: cabbage, wolf
7. The man takes the goat from the East to the West.
   -->
   East:                                   West: man, cabbage, goat, wolf

Congratulate yourselves for solving the MCGW problem!
```

Figure 1: The expected results from your program.

Deliverables:

1. A report documenting the process of solving the problem
   a. Two pages maximum. One inch on all margins and Times New Roman font with 11pt and single space.
   b. The report must contain the following information and sections. There is no need to include the code, because it is already in the .py file.
      i. The names and student ID of the two members, and the class A or B
      ii. A problem description
      iii. Data abstraction (including a description of the states, a graph as a result of the data abstraction, the data types required, …)
      iv. The algorithm needed to solve the graph problem
      v. A modular design of the program
      vi. A Python implementation of the data types

2. A well documented Python program in a single "G"+"your group number".py file, e.g., G01.py for group 1 and G11.py for group 11.
   a. You cannot import any module to your program.
   b. You must use docstring to describe each function.
   c. By using appropriate comments and variable names, your program must be easy to follow and understand.
   d. Give a proper reference to the source of the code that you adopt for your program.
   e. Do not hard code: You should rely on your code, your logic, your program, and you should *not* write things out by hand or do computation in your head – even if you can do so easily. (https://runestone.academy/ns/books/published/fopp/SimplePythonData/HardCoding.html)

Avoiding plagiarism at all costs:

1. It is perfectly ok for you to discuss ideas and approaches to tackling this project problem.
2. However, your group's code could not possibly be the same or very similar to other groups', because there are just so many ways a program can be written to solve this problem.
3. There were plagiarism cases in the last two years. Many of them ended up failing this course, even though they did ok in their final exams.
4. If plagiarism is found, all the students involved will receive –100 marks for this project, assuming the full mark is 100. That means they will most likely fail this course.

Rubric for grading:

	Expectations fully met (4)	**Expectations mostly met (3)**	**Expectations somewhat met (2)**	**Expectations largely not met (1)**	**Expectations not met at all (0)**
Program Correctness (30%)	The program runs correctly according to the specification for all cases.	The program runs correctly according to the specification almost for all cases.	The program runs correctly according to the specification only for the major cases.	The program runs correctly according to the specification only for a small number of cases.	The program does not run correctly almost for all cases.
Code readability and documentation (30%)	The code is very well organized and very easy to understand. The documentation is very clear and accurate.	The code is generally well organized and easy to understand. The documentation is generally clear and accurate.	The code is adequately organized and needs some effort to understand. The documentation is adequately prepared.	The code is poorly organized and is difficult to understand. The documentation lends very little help.	No effort to organize the code for understanding. There is no/very little documentation.
Content of the report (40%)	The report documents all the main steps of problem solving very clearly, logically and comprehensively.	The report documents most of the main steps of problem solving clearly, logically and comprehensively.	The report meets just the basic requirements in terms of clarity, logical presentation and comprehensiveness.	Lacking important details and clarity, the report offers limited help in understanding the problem-solving process.	The report is largely incomprehensible. It offers no help in understanding the problem-solving process.