CS 11 Machine Problem 2 - Your Own Graphical/Game Application

University of the Philippines Diliman

November 2018

Instructions

- You have to do this machine problem in groups of three. Groupings will be posted in UVLe.

Submission Guidelines

- Submission links will be posted in UVLe.

Deadline

- The deadline of all deliverables is on December 01, 2018.
- Late submissions will receive a 10% deduction per day late, with a maximum of 70% deduction.

Evaluation

1. The machine problem will be evaluated based on the following:

1	File Input/Output	10%
2	User Interface - Graphical Display	15%
3	User Interface - Event Handling	15%
4	Engine	15%
5	Programmer-defined objects	15%
6	Modularization	10%
7	Documentation	10%
8	Presentation/Demo	10%
Bonus	Virtual Environment	10%
Bonus	Application Website	10%
Bonus	Exhibition Score	10%
	TOTAL	100%
	TOTAL (WITH BONUS)	130%

   Each item is described in the succeeding sections.

2. Each member must rate other members by submitting a peer evaluation. Each individual student must also submit a self-evaluation. Peer evaluation and self evaluation forms will be available after the software demo.

3. The grade of each student will be

   (MP Grade) × (Average of Peer Evaluation and Self Evaluation)

   Where the average of peer evaluation and self evaluation is a number from 0 to 1.

Introduction

For this Machine Problem, you have to create an application that is subject to the following constraints:

1. Your program must use the `Pyglet`[1] programming framework.
2. The application must write some data to a file.
3. The application must read some data from a file.
4. The application must have a graphical user interface (i.e., the interface is not on the terminal).
5. The application's interface must have the facilities to accept input from at least 1 input device (mouse, keyboard, web cam, microphone, etc.).

[1] `https://bitbucket.org/pyglet/pyglet/wiki/Home`

6. The application must have a purpose (it could be a game for entertaining users, music/video player, mailing application, graphical editor, etc.).
    - A word (un)scrambler game is not allowed for this MP since you already did it in MP 1.
7. The application must utilize objects that are defined by the programmer.
8. The application's source code must be divided into two or more modules (similar to your MP1).

You have the freedom to think of any kind of application, as long as it satisfies the constraints mentioned above.

After submitting your application on or before the due date, you have to present your work to the CS 11 instructors.

Bonus points will be given if the following are satisfied:

1. Your program must be developed in a virtual environment.
2. Your program has Github/Gitlab page.
3. Your program is one of the top 50 programs during the exhibition.

Details on these are written at the end of the specifications.

1 File Input/Output (10%)

Think of any requirement that will necessitate your program to get data from a file (e.g. getting the game leaderboard, getting a user's profile, opening a media file, etc.). Also do the same thing for storing data to a file (e.g. saving a game leaderboard, saving/updating a user's profile, etc.). There is no upper limit to the number of files that your program has to read and write.

2 User Interface - Graphical Display (15%)

Using `pyglet`, you have to create windows and other graphical entities to show your program. The terminal must only output debug messages. Debug messages are messages that are not important to the user but are often used to debug the program.

The minimum requirement is for your program to display output at the screen. Output coming from speakers, printers, etc. are optional, and it may depend on the application that you will create.

3 User Interface - Event Handling (15%)

How the user interacts with the program must be done through handling events. Examples of events are the following:

1. Pressing a key on the keyboard.
2. Clicking/hovering/scrolling/holding the mouse.
3. Clicking the close window button.
4. Receiving video input from the web cam.
5. Receiving audio input from the microphone.

4 Engine (15%)

Part of your program will be dedicated to an engine, or the component that contains all the algorithms needed by your program to serve its purpose. This is similar to your game engine in MP1.

5 Programmer-defined objects (15%)

Your program must utilize objects that are defined by the user. User-defined objects could be geometric entities such as points, shapes, or any abstract entity such as user profile, game state, etc.

6 Modularization (10%)

Your program's source code must be divided into two or more modules. Your first MP shows an example of a modular source code, where the engine, interface, and main program are separate. Note that when done correctly, you can create a GUI-based program using the same engine module, and without touching the interface and the main program. This is because the interface and the main program is stored separate from the engine.

Since we're already talking about modules, please don't forget to encapsulate your processes/algorithms into functions. If you want to know why you have to do this, please read section 3.11 of *How to Think Like a Computer Scientist*. This is also found in the lectures.

You can add as many user-defined modules as you like, but the minimum is to separate the program into 2 or more components.

7 Documentation (10%)

Your program must be properly documented:

1. (4 pts) Create a programmer's guide for your program. It is like a user manual but instead of "how to play the game," it answers "how the program is designed" and "how the program works."
2. (2 pts) Create a properly documented code by using descriptive names for variables, functions, and modules.
3. (2 pts) Add comments to the code if the source code itself does not suffice in describing what the program does.
4. (2 pts) All resources (books, sample codes, online resource or person) used should be properly cited in the source code (comments) and/or the programmer's guide.

8 Software Demo (10%)

After submitting your code, you have to present your program in public. This means that your audience may not only be your instructors. Your audience may include other CS11 students, other students, faculty, etc. The demo is scheduled on lab hours, so please make sure that you and your group mates are available.

During the demo, you have to present the following:

1. Sample Run *(5 pts)*
2. Source Code *(2.5 pts)*
3. Documentation *(2.5 pts)*

Bonus: Virtual Environment (10%)

The `Pyglet` framework is a third-party Python package. Third-party packages are not usually included in the default Python installation, and they are installed separately. Most third-party Python packages (such as `Pyglet`) are installed using `pip`[2].

The problem with using external packages (such as `pyglet`) is that there could be programs in your computer that also depend on these packages (and their dependencies), and installing/updating a package might affect other programs that are using it. This means that the packages that your program use must be isolated from the packages that other programs use. One technique to isolate packages is by creating a *virtual environment* for your program.

[2] `https://pip.pypa.io/en/stable/installing/`

Please isolate the installation of third-party packages (pyglet, etc.) using the `venv` module `https://docs.python.org/3/library/venv.html`. Also, please save all the third-party Python packages used in a `requirements.txt` file. An easy way to generate this file is through `pip`.

```
# If there is only one Python in your computer (Python3),
# you can use this.
pip3 freeze > requirements.txt

# If there are 2 Python versions in your computer, the Python3
# counterpart of pip is pip3
pip3 freeze > requirements.txt
```

Bonus: Application Website (10%)

Introduce your program to a wider audience by publishing them online through a Github[3] or a Gitlab[4] page. What these pages are and how to create these pages is for you to find out. Several resources on how to create pages are available online, and the minimum requirement is to learn how to use Git, Github, and/or Gitlab.

You will only get full points on this part if you have completed all the non-bonus requirements of this MP.

Bonus: Exhibition (10%)

There will be exhibition of your games where in audience will be voting for the best work. Then MPs will be ranked based on the votes, and bonus points will be based on your ranking.

Rank	Score
1-10%	10
11-20%	8
21-30%	6
31-40%	4
41-50%	2
51% and below	0

[3] `https://pages.github.com/`
[4] `https://about.gitlab.com/product/pages/`

This means that if you didn't get in the top 10 of the exhibition, then you will never get a perfect score on this part. ***Specific details and mechanics of this part will be posted in the UVLE.***