De La Salle University- Manila
Gokongwei College of Engineering

LBYCPA1
Programming Logic and Design Laboratory

Project Proposal

Multi Pac-Man

Leigh Andrei M. Tabanao
Timothy Brian A. Tiu

Project Description

The project proposed by the students is a Python recreation of the classic hit game Pac-Man, but with some twists from other classic video games. The students also propose the usage of the Python library "Pygame" alongside the lessons of the course program, to improve the graphics for the game.

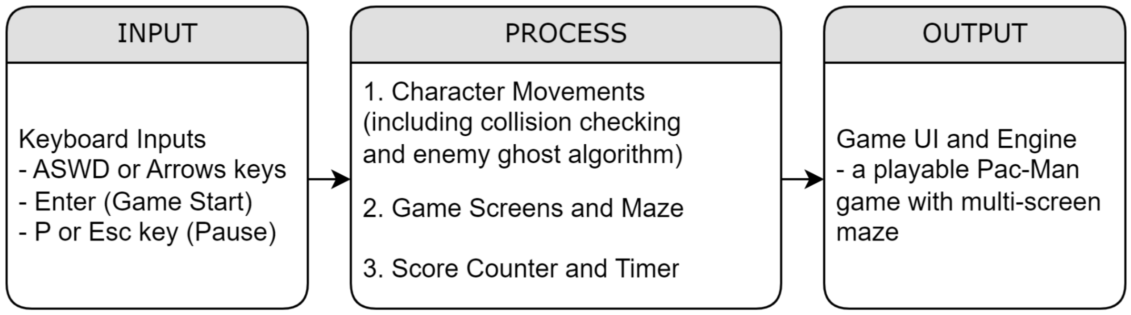
For a brief description of the original game: the player will be controlling Pac-Man, a yellow ball tasked to collect/eat all the cookies (yellow dots) in a maze as quickly as possible. However, in the game, there are ghosts, colored pink, orange, red, and cyan, roaming the maze. The player shall help Pac-Man evade, as touching these ghosts will result in losing a "life." The game is over when all of Pac-Man's lives are lost, and the score is based on the number of cookies that have been eaten.

The proponents have decided that the twist or unique mechanic of this game recreation would be the addition of multiple continuous pages and/or screen scrolling, akin to other classic video games such as Super Mario Bros and The Legend of Zelda. This is where the title of the project, "Multi Pac-Man," comes from. Furthermore, if time permits, the students might redesign the other "screens/pages" into the theme of Super Mario Bros. and/or Zelda, adding "multiversal travel" into the theme of the game, further fitting the title "Multi Pac-Man."

Thus, the technical objectives for the project are:

- To recreate the game Pac-Man using the Python programming language,
- To implement the multi-screen and/or screen scrolling feature seen in Platformer and JRPGs game genres into Pac-Man,
- (Optional) To incorporate a level/stage progression similar to the original games, and
- (Optional) To design a multiverse-like multi-screen maze, with different game universes serving as the other screens/"pages."

Table 1: IPO


INPUT
Keyboard Inputs
- ASWD or Arrows keys
- Enter (Game Start)
- P or Esc key (Pause)
PROCESS
1. Character Movements (including collision checking and enemy ghost algorithm)
2. Game Screens and Maze
3. Score Counter and Timer
OUTPUT
Game UI and Engine
- a playable Pac-Man game with multi-screen maze

This IPO diagram shows how the program will run. As the game progresses, the program will accept inputs from the user that will dictate which direction Pac-Man will walk. Additionally, other inputs will also be recognized to start or pause the game. The processes inside the program include the functions that let the characters move in different directions. This will also include the continuous collision check of Pac-Man to walls or the enemy ghosts. The other two processes include the level setup, which includes the multi-screen maze, and the score counter. Lastly, the output is the working game itself.

Methodology

The project will make use of the base concepts of the Python programming language combined with the library Pygame, which should aid the program in executing the necessary mechanics and displaying the game. Furthermore, online guides, like those for recreating the game (Jileček, 2022; Liu, 2019) and those for including multiple screens (GeeksforGeeks, n.d.), will aid the proponents in building the proposed program. The flow diagram that follows explains briefly how the program will function.

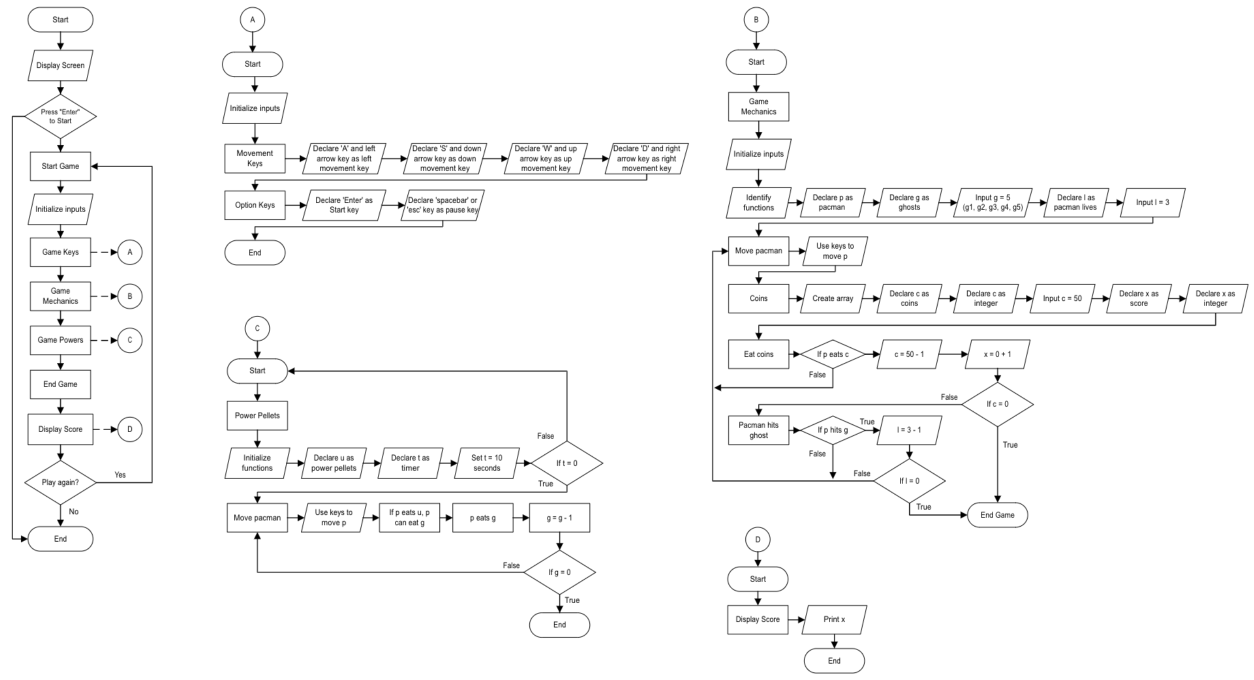


Figure 1: Program/System Flowchart

The leftmost flowchart represents the overall system of the game, where some variables named as letters, A, B, C, and D are expanded and explained further by the corresponding flowcharts on the right. The 'A' flowchart represents the game keys that are initialized to play the game, this includes the movement and option keys. The 'B' flowchart showcases the game mechanics, and determines how to win or lose the game, while the 'C' flowchart incorporates the extra options or the game powers that help the player win the game. The 'D' flowchart displays the player's game score.

Python Concepts:

- User Inputs and Interface Output: These will be used to get the keypress from the player and display the game itself.
- Loops and Conditional Statements: Both of these concepts will be used to continue the execution of functions, particularly those of character movement, collision checking, and that of the program itself.
- Arrays / Dictionaries: One of these will be used to keep track of player records/scores.
- File Operations (Optional): The player information and high scores will be stored in a separate file to serve as a database of previous program executions.

Schedule of Activities

Table 1: Gantt Chart

Activity	Team Member Assigned	Month	March				April		
		Week	1	2	3	4	1	2	3
		Day	26-4	5-11	12-18	19-25	26-1	2-8	9-15
Initial Project Proposal	Tabanao, Tiu		xxxx	xxxx					
Final Project Proposal	Tabanao, Tiu				xxxx	xx			
Game Mechanics Development	Tabanao, Tiu					xx	xxxx	xxxx	
Game Coding	Tabanao, Tiu						xxxx	xxxx	
Game Design	Tabanao, Tiu						xxxx	xxxx	
Project/ Document Review	Tabanao, Tiu								xx
Demonstration	Tabanao, Tiu								xx

References

The following articles/resources may be of help in accomplishing the project/program:

GeeksforGeeks (n.d.). How to Use Multiple Screens on PyGame? Retrieved March 17, 2023 from https://www.geeksforgeeks.org/how-to-use-multiple-screens-on-pygame/

Jileček, I. J. (2022). Creating a Pac-Man clone in Python in 300 lines of code or less. *ITNEXT - Medium.* https://itnext.io/how-to-create-pac-man-in-python-in-300-lines-of-code-or-less-part-1-288d54baf939

Liu, A. T. (2019). *Pacman With AI Python*. [GitHub Documentation/ Repository] https://github.com/andi611/Pacman-With-AI-Python/blob/master/task1_Search/pacman.py