Nhyira Mante

Professor Rocky Chang

CS-108

Final report

<u>**The Clicker Challenge: An interactive Reflex game**</u>

Nhyira Mante
13 December 2024

Novelty

This program ius an interactive reflex game that allows users who play reflex based games like call of duty, Valorant etc to train their skills. There are a few games that do this but most have their features locked behind paywalls and also have static and repetitive designs to combine a visual and interactive experience with customizable difficulty levels. This program configures game elements (e.g., target speed) based on user-selected levels to make it harder. The game also has an amazing soundtrack tuned to the levels. It integrates the music, responsive targets, and user feedback (e.g., scoring) to provide an engaging user experience. It uses Python's tkinter for interface design and pygame for sound. Its also coded so it can easily be changed for future features like leaderboards or multiplayer support. I also made it hyper-realistic to shooter games:). (sometimes when you click it may not register (happens very randomly))

Goals

The overarching goal of this project is to design a fun and interactive shooter game that is easy to use and adaptable. Specific objectives include:

1. Developing a visually appealing user interface with customizable difficulty levels.
2. Implementing smooth gameplay, including responsive target movements and score tracking.
3. Incorporating sound effects and music for an immersive gaming experience.
4. Providing an end-game screen with feedback on the user's performance.

Data and Python Data Types

This project involves various types of data to ensure functionality:

- **Target Positions**: Stored as integer coordinates (x, y) for precise placement.
-
- **Score and Timer**: Managed as integers, ensuring accuracy in calculations.
- **Flags (e.g., stop_moving, stop_countdown)**: I represented these as Boolean lists because through reddit and combining what you said about a the mutability of lists and how they are stored I realized that if I was to make a true or false variable in a list if I updated it it would reflect in other functions.
- **Button State**: Managed through tkinter buttons to enable/disable user interaction dynamically.

- **Canvas Elements**: These are all class objects that I employed several instances of like create_rectangle and create_image etc. I used them all over my code to handle tkinter widgets and enable visual updates for game objects and progress bars.

Algorithm Design

The game had a lot of interconnected functions, and it was very challenging to implement them logically and correctly. The final project operates through a series of interconnected 'algorithms':

1. **Countdown**: Decrements the time, updating the canvas every second. If interrupted (e.g., by pressing "End Game"), it halts immediately. This part was so so difficult for me because of the endgame button. The after calls proved to be extremely stubborn to stop when a button is pressed.
2. **Dynamic Target Movement**: This uses a random function to calculate new coordinates for targets, ensuring unpredictability. The speed adapts based on the difficulty level selected. I had to try many many different ways of removing the targets before I found one that worked. The problem was because I used after calls to move the targets, the targets kept on moving even after the game ended.
3. **Score Updating**: Each successful target click increments the score. Data is displayed dynamically on the canvas.
4. **End-Game Logic**: Triggers a transition to the game-over screen, displaying final scores and options for restarting or returning to the main menu.

Program Design

Function	**Inputs**	**Outputs**	**Description**
main()	None	None	Initializes the game window, canvas, and main menu.
open_app()	Canvas, Buttons	None	Transitions to the loading screen and displays a progress bar.
update_progress()	Canvas, Loading Bar Dimensions	None	Updates the progress bar and transitions to level selection.
choose_level()	Canvas, Root Window	None	Displays level selection buttons and handles user input.
start_game()	Selected Level, Canvas, Root	None	Starts the game, initializes targets, and activates countdown.
move_target()	Targets, Speed, Game Area	None	Moves targets dynamically within the game area.
count_down()	Time, Timer Text, Callback	None	Decrements time and handles end-of-game logic.

Function	Inputs	Outputs	Description
end_game()	Canvas, Score, Flags, Root	None	Displays the game-over screen and final score.
play_music()	Filename	None	Plays background music during gameplay.
target_clicked()	Target Index, Score Data	None	Updates the score and relocates the clicked target.

Help sources.
External sources for help:
----

1. Tkinter Course - Create Graphic User Interfaces in Python Tutorial on Youtube
   https://www.youtube.com/watch?v=YXPyB4XeYLA&pp=ygVNVGtpbnRlciBDb3Vyc2UgLSBDcmVhdGUgR3JhcGhpYyBVc2VyIEludGVyZmFjZXMgaW4gUHl0aG9uIFR1dG9yaWFsIG9uIFlvdXR1YmU%3D

2. The ultimate introduction to modern GUIs in Python [with tkinter] also on Youtube
   https://www.youtube.com/watch?v=mop6g-c5HEY&pp=ygVTVGhlIHVsdGltYXRlIGludHJvZHVjdGlvbiB0byBtb2Rlcm4gR1VJcyBpbiBQeXRob24gWyB3aXRoIHRraW50ZXIgXSBhbHNvIG9uIFlvdXR1YmU%3D

3. https://docs.python.org/3/library/tkinter.html
   This site for all the module information and how to use the attributes methods etc.

5. I also used r/learnpython subreddit to help find answers when I was lost.

6. I used this to get started it's a bit like the Guizero site you gave us.
   https://www.geeksforgeeks.org/python-gui-tkinter/

7. For the music I found out pygame could play it on reddit
   https://www.youtube.com/watch?v=xdkY6yhEccA
   this was an easy to understand for implementing the music

8. I used python tutor to follow the progress of the code to check the steps and what the code did. I also used the debugger tool a bit but that was only recently.

9. I used to learn a bit more about list mutability and that's what helped me to discover using lists as stop flags and to update score etc because of their mutability
   https://www.youtube.com/watch?v=ohCDWZgNIU0