**Python3 Programming Project Report**

**Peruke Game**

**Student:** Leen Ghazi

**Introduction:**

The "Peruke Game" is designed to enable the players to be creative and smart in order to win the game. The game requires two or more players, and each player has 2 rows of discs, "Primary row: safe (flipped) and Secondary row: target (shown)". To win the game, you have to get a high score in many ways. First, make sure most of your discs are safe. Second, take other players' discs by rolling the dice and matching the discs with the dice number. You can do that by flipping the target disc once, or the safe disc twice. The way you choose which disc to be flipped is by rolling the dice and matching the value of it to the disc.

**Requirements:**

1. The user will choose between logging in or registering.
2. The program will ask the user to fill in some information about login or registering. The user who does that will be player 1 “by default”.
3. The program will make modifications to the excel sheet if applicable.
4. Then, the players have to enter their names (players’ names) and press start.
5. For the game, the game frame will show the players’ names, primary and secondary discs for each player, the “roll dice” button, and the dice images. - If the player’s name box is green, it means it is his turn , if it is red, it is not his turn -.
6. The player will press the “roll dice” button and match every die to the discs shown in the game. He can:
   * make his disc safe
   * switch other player’s disc from safe to target
   * take other player target disc and add them to their score
7. The program will stop when the primary row of any player is all taken.
8. The program will record the score the same way as it shown on the website of the original peruke game.
9. The program will display the score, and the result of the game.

**Design and implementation:**

In my program, I used the GUI to visualize the game and get a better experience, even though I have zero experience in programming using the GUI. YouTube was my main source for learning to implement the GUI in Python. I implemented many libraries to make the code run. I chose tkinter library for the GUI. Then, I implemented the random library for the dice and got a random number every time the dice is rolled. And I used openpyxl to activate the excel sheet, register new players, check log in information, and save any modifications (using the excel sheet as a database).

The main window of the GUI has exactly five frames. The first 3 frames are for introducing the game, checking the registration/login information, and asking for the players' names. The fourth frame is for the main game, which includes the discs, dice, and players' names. And the last frame is for displaying the score and the winner of the game **(figure 1–6).**

I used images to design the interface of the program. (In this program, the user who register/log in 🡪 player 1, by default) I draw the discs and dice by myself to fit the design I created in my program. Moreover, I used several frames in my code to control the widgets (e.g., buttons, labels, entries, etc.). In addition, I made a lot of functions to be called and made some actions if the user pressed some buttons. Also, I define integer variables for counting the scores “**player1\_score** and **player2\_score**”, how many times a player pressed a disc (counting the movement of each player) “**player1\_disc\_press** and **player2\_disc\_press**”, and the available moves for each player. Finally, I use lists to assign the buttons (discs). Primary row buttons list which has text of their numbers (score) and images of a safe discs, and a secondary row buttons list which has text of its number as well as images of a target discs “**Player1\_primaryRow – player2\_primaryRow – player1\_secondaryRow – player2\_scoendaryRow**”.

Additionally, other than the frames, the widgets used in my program are buttons (discs + roll the dice + register + log in), Labels (dice + primary and secondary row discs + players names), Photos (used in the buttons and labels), and Entry (ask to register/ log in + ask for players’ names).



Figure 1

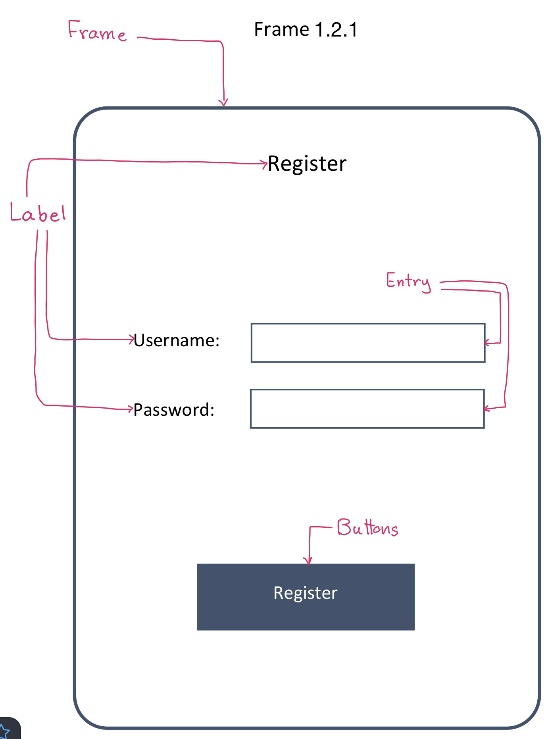


Figure 2

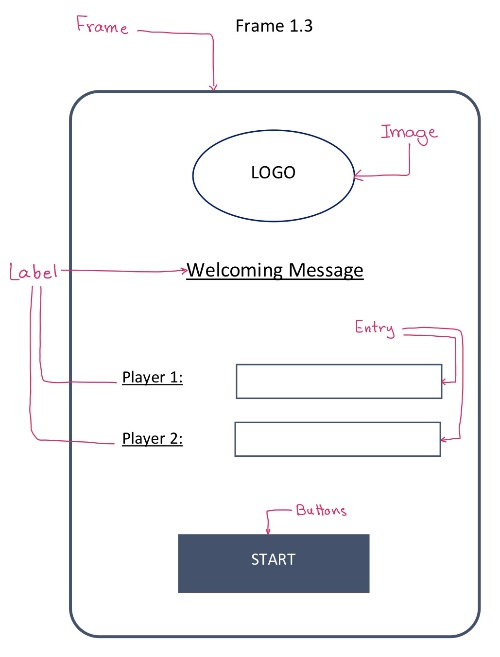


Figure 4

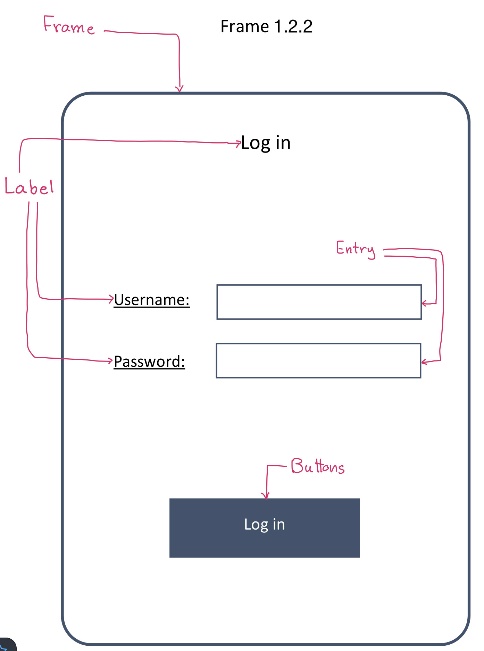


Figure 3

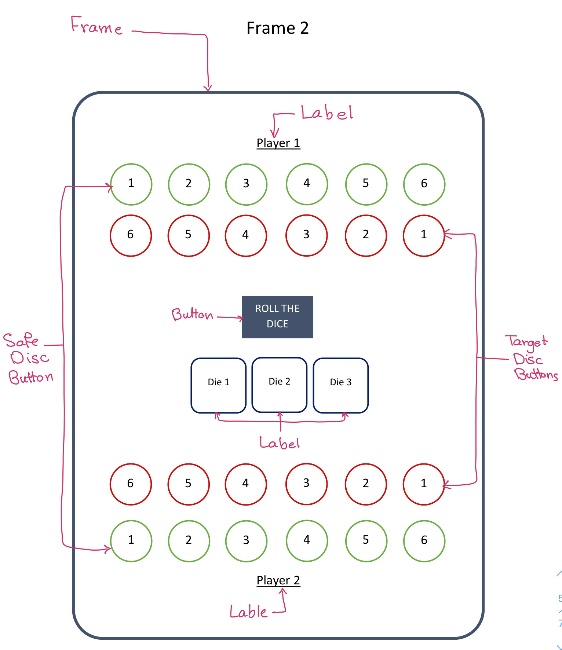


Figure 5

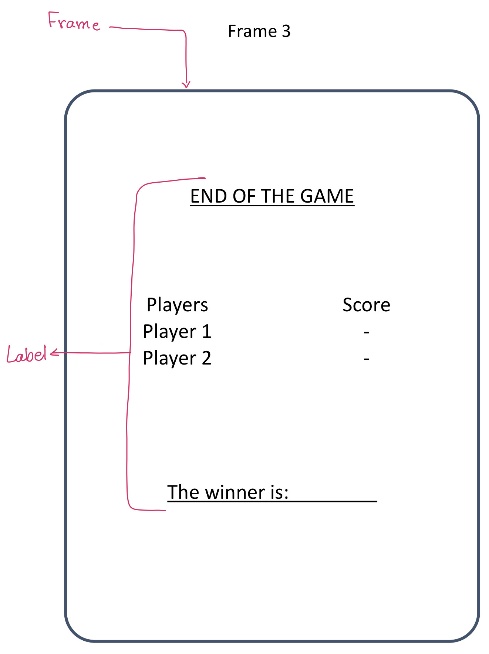


Figure 6

**Programming techniques used:**

Libraries/Packages/etc.:

* tkinter: for the GUI
* openpyxl: to active the excel sheet, register new players, check log in information, and save any modification.
* random: for the dice and get a random number every time the dice is rolled.

Database (filename= Peruke\_game\_records.xlsx):

* Usernames
* Passwords
* Number of games played
* Number of winnings
* Number of loses
* Highest score

The most important functions in my code are:

* clearFrame{frameNumebr}(): to switch between frames by removing the grid of the frame (hide the frame)
* check\_username\_and\_password(): check if the username or password in the database
* username\_and\_password\_registration(): to register new players in the database (worksheet)
* username\_and\_password\_login(): to check players’ information in the database (worksheet)
* roll\_dice(): to get random value every time the “roll the dice” button is pressed
* disc\_action\_for\_player1(): function for player 1 to make their rows safe and take other players row
* disc\_action\_for\_player2(): function for player 2 to make their rows safe and take other players row
* make\_disc\_safe(disc): to make the disc safe by changing the photo of the disc
* make\_disc\_target(disc): to make the disc target by changing the photo of the disc
* reset(): to reset the dice numbers, available move and disc press for each player, and the command for each disc (button)

I created many algorithms during playing the game. The most important algorithm is to assign the right function to the button command. I made several if statement and for loops to check the disc is:

1. match the dice number appear in the interface
2. Choose the right function to implement
   1. Make disc safe: if the disc is belonged to the player
   2. Make disc target/take: if the disc belonged to other players. If the disc is already a target, the disc will be disable (has been taken by the enemy)
   3. Do nothing: if the disc didn’t match the dice number
3. 2 different algorithms to eliminate extra dice (the one that does not match any available disc)

Also, I make the available disc to be played have some special attributes (shapes) to guide the player which disc could be used to play, such as the border and background color of the disc. In addition, I used the openpyxl library to edit the database (excel sheet) .

For the variable, as I mentioned earlier, I used integers to count players score, their available move according to the dice number, and how many time a disc is pressed by players. I made an algorithm to check:

* if dice button press % 2 == 1 🡪 player 1 turn
* if dice button press % 2 == 0 🡪 player 2 turn
* if len(dice\_number list) == 0 or available move == 0 🡪 switch players

I also made most of my functions have some sort of recursion. Because most functions end with a function to be called or an action to be performed.

**Instructions:**

First, the programmer has to download a folder to the Python program file. This folder has one python program (PERUKE GAME), which has the python code for the game. The folder also has an excel sheet (Peruke\_game\_records) which has the database for log in/registration information, number of games played, number of wins, number of losses, and the highest score for the use. Finally, the folder has images. 6 images for the safe discs, 6 images for the target discs, 6 images for the die, and the game logo. All you have to do is to open the PERUKE GAME python file, run the program, and enjoy the game. The figures below show in order the interface of the program. **(Figure 7 – 11)**



Figure 7

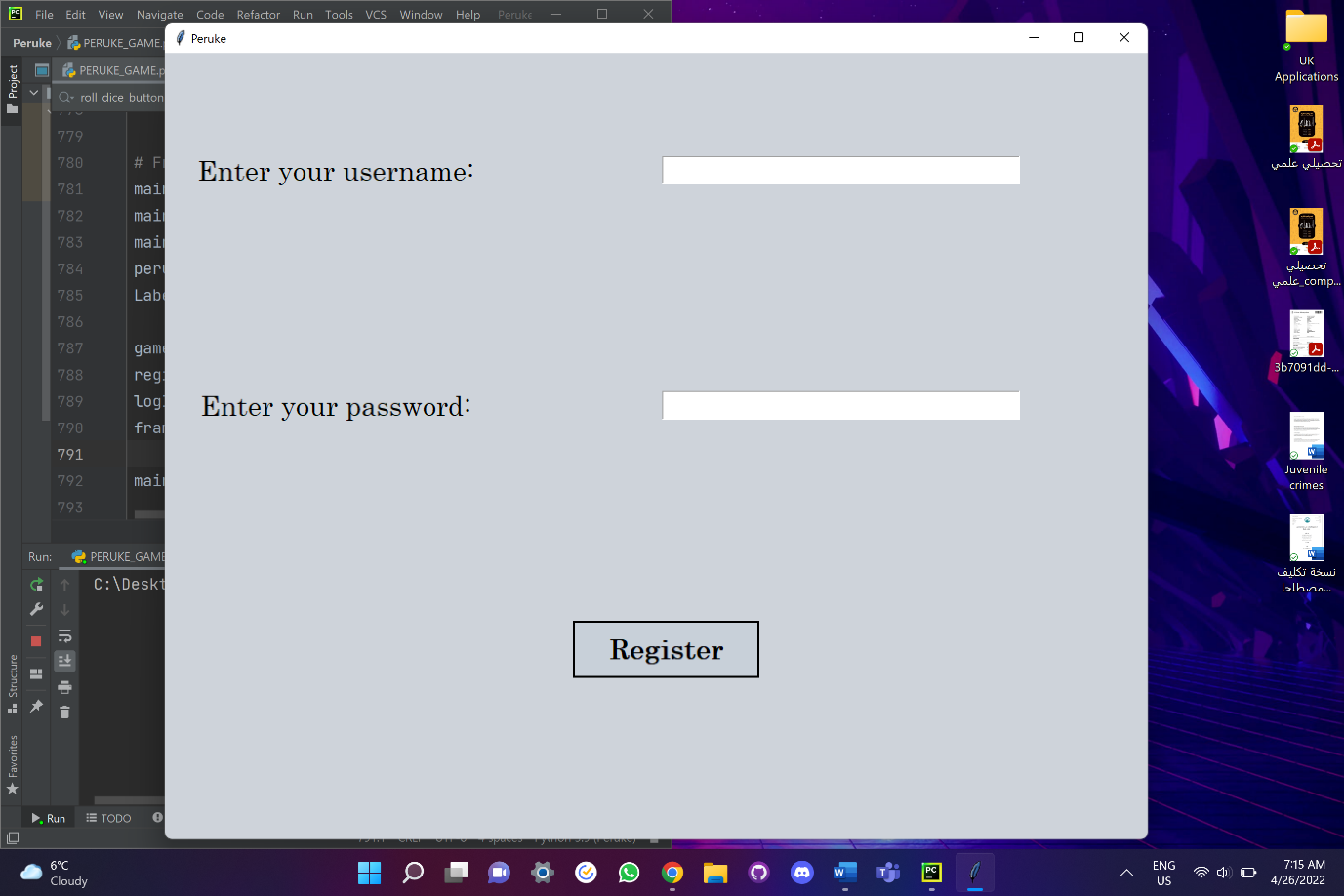


Figure 8.1

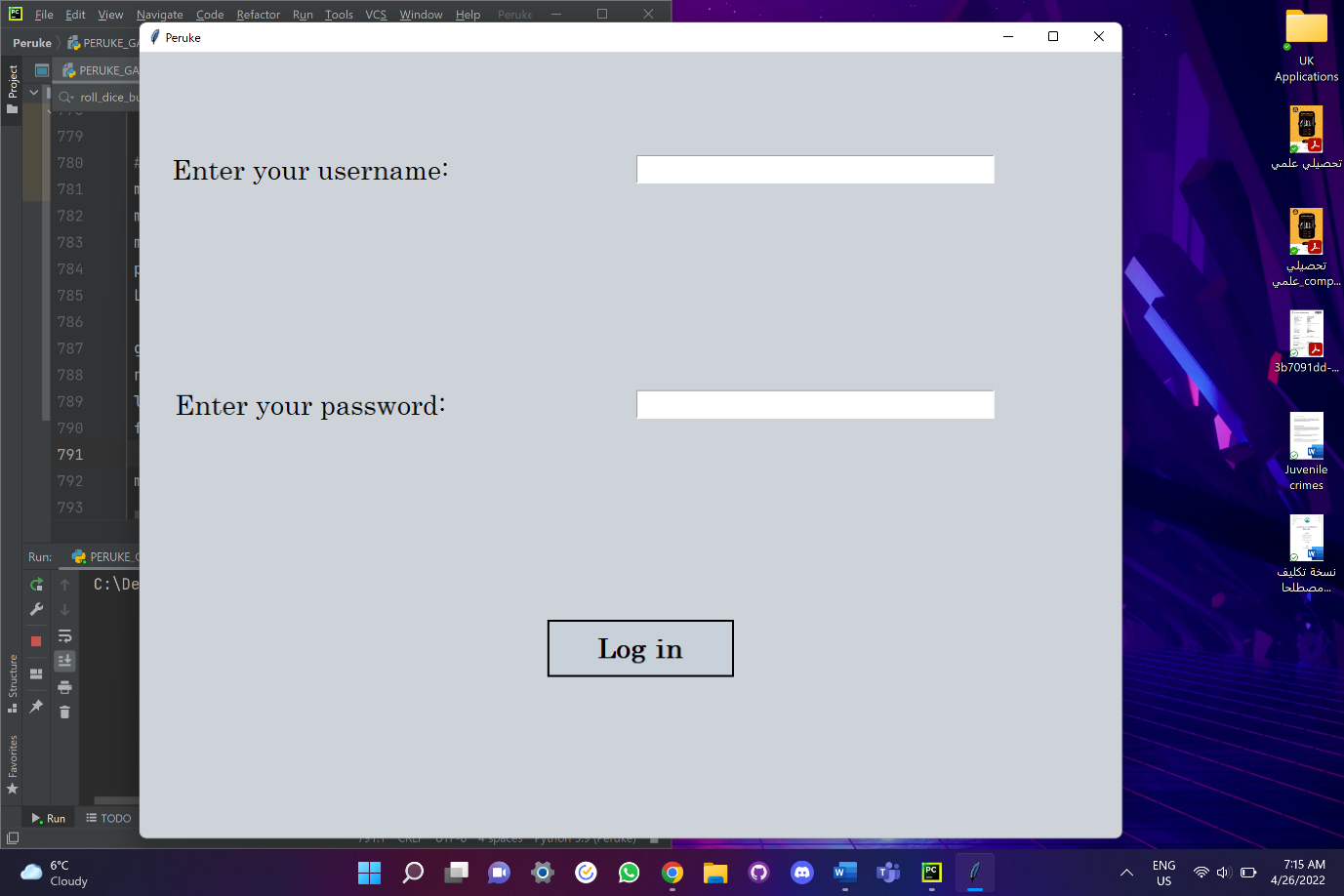


Figure 8.2

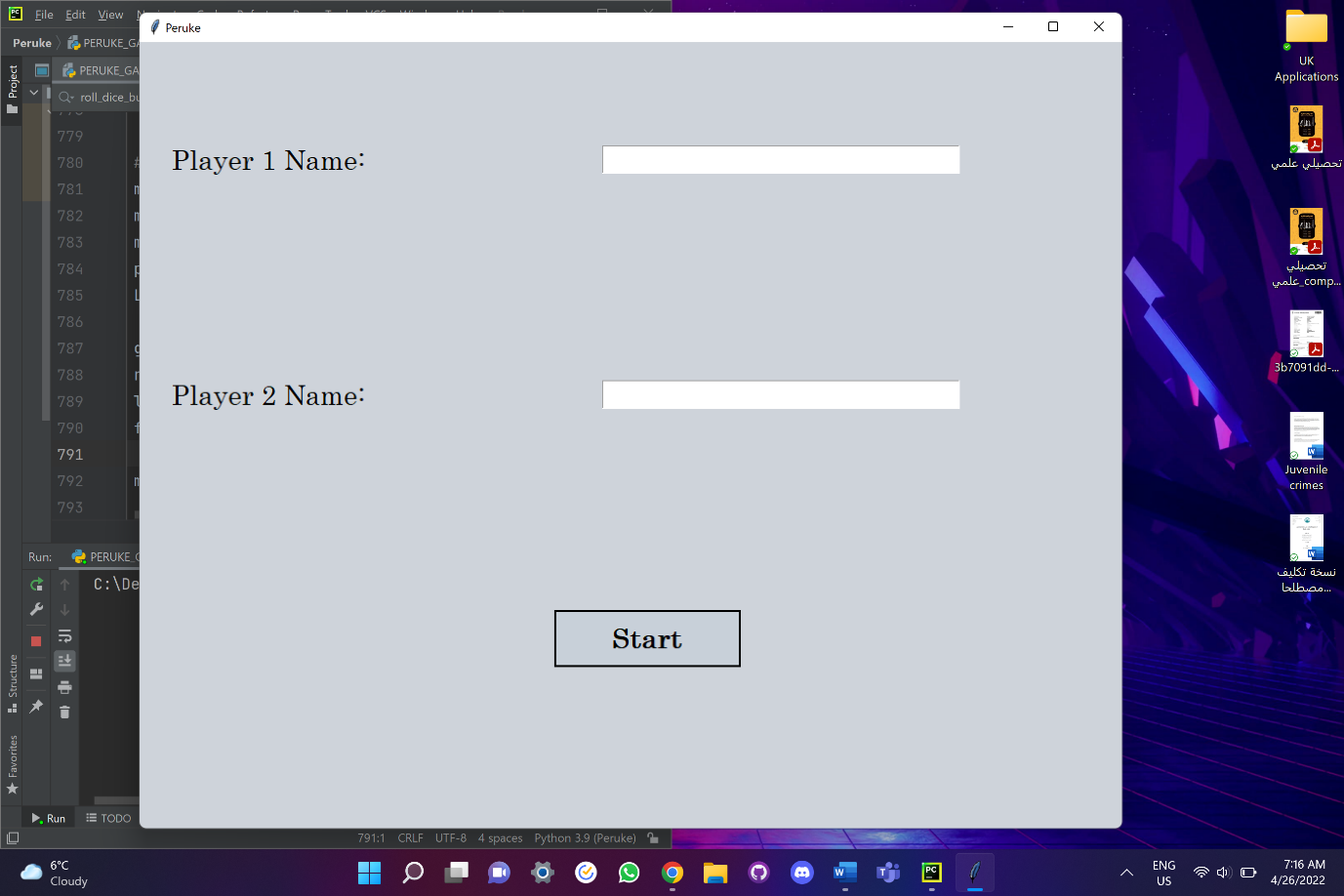


Figure 9

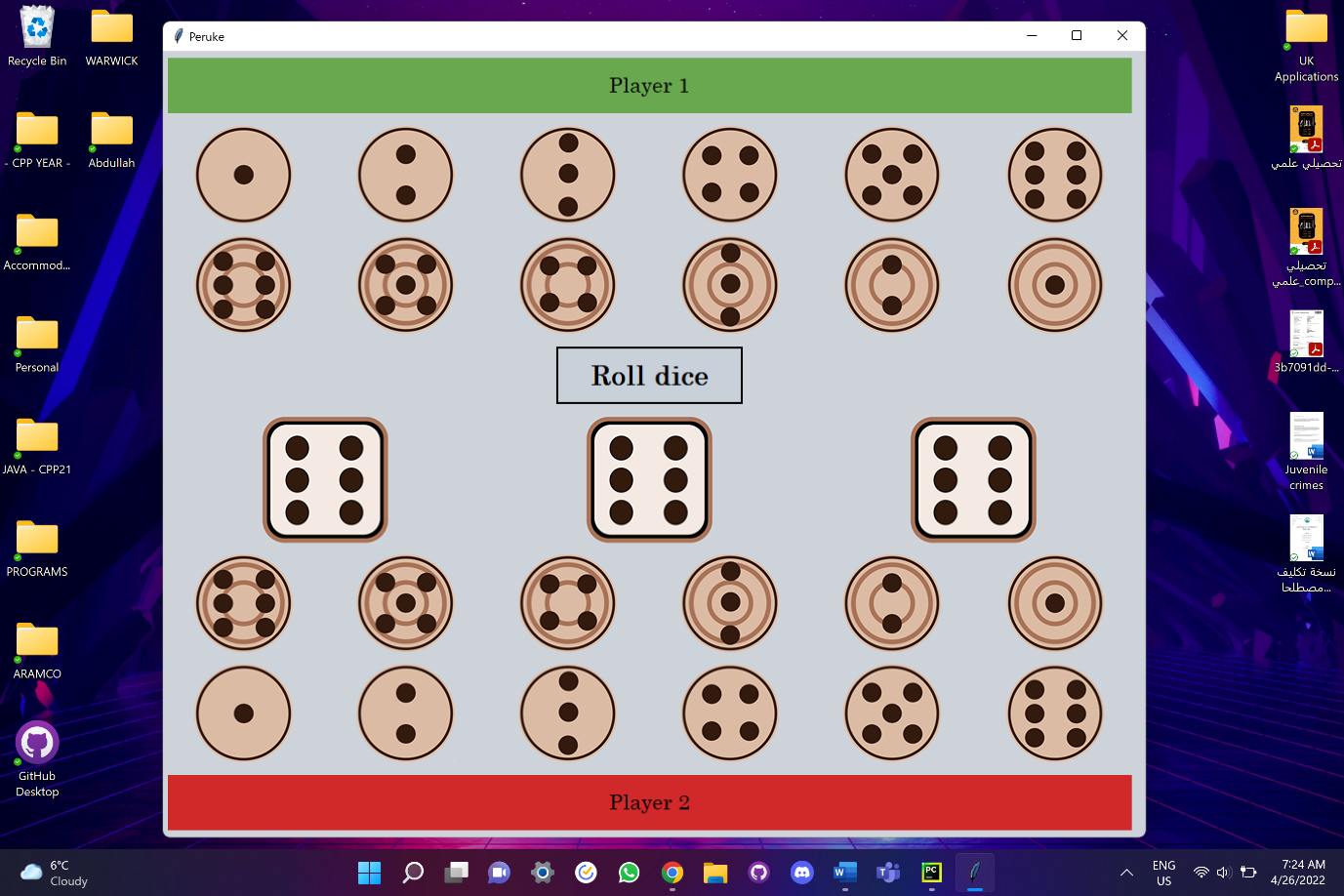


Figure 10

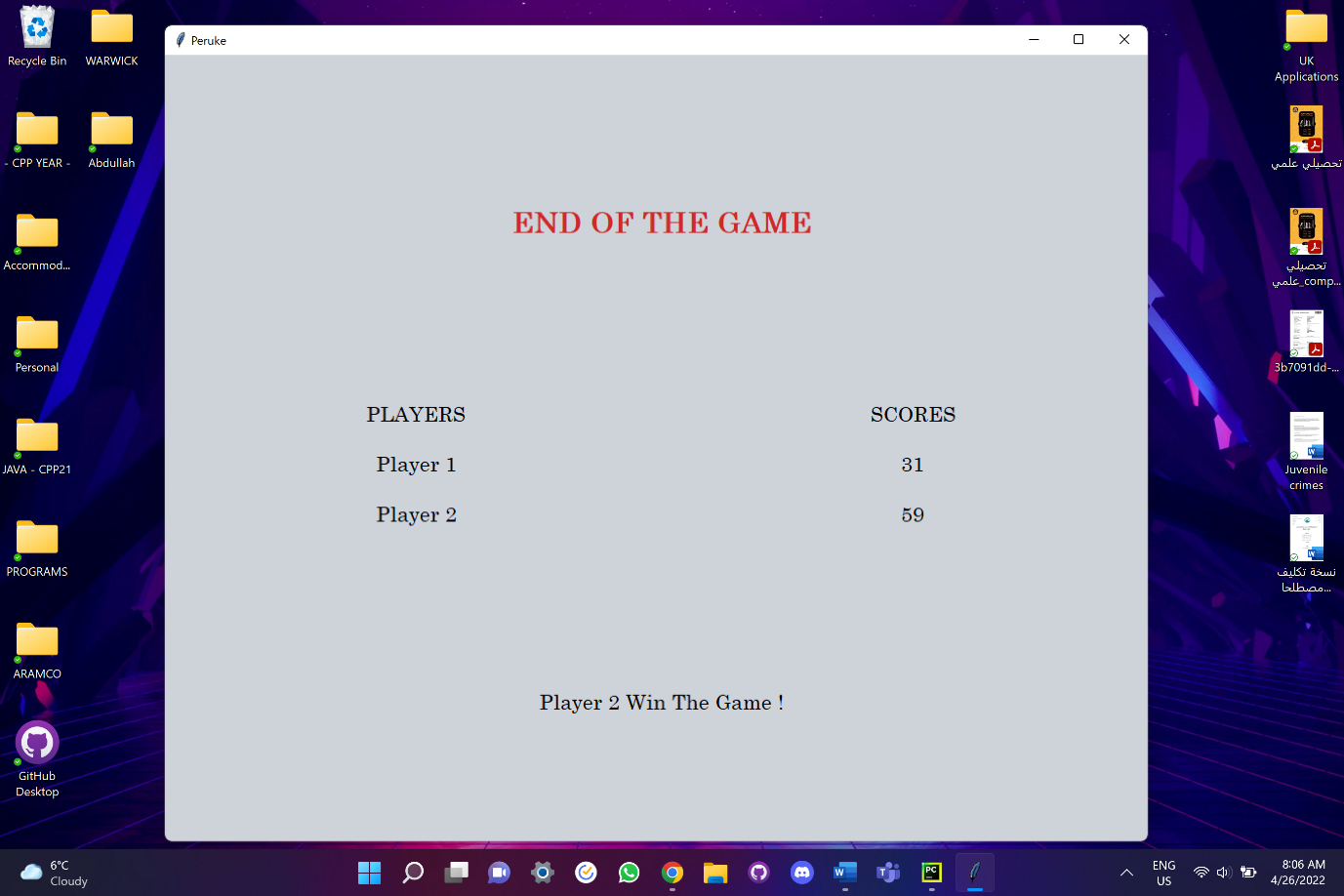


Figure 11

**Testing and maintenance:**

To test the program, I used another Python file and copied the code to it to test the effectiveness of the function used in the main program. I used only 2 buttons, one is safe and the other is a target. By using this Python file, I was able to run the program, fix all the gaps, and cover all the possibilities while interacting with discs, such as making a function called do\_nothing() and assigning it to each button (disc) that didn’t match the dice number.

Unfortunately, this program is limited to two players only. Also, I could not be able to restart the program after the result was shown (the programmer "tester" had to close the python file and rerun it). Moreover, the program does not have a menu or back button, when registering the information. Another thing I would do is to check the border color of a disc when the dice are rolled. The green border means the disc is available to be pressed, and the red means it is not available. Although the code is not able to check the border color every time a disc is pressed yet, the code is still working well (just needs to work on the layout). For example, if you roll the dice and have the dice numbers (3, 4, 5), all the discs with those numbers would be bordered by green. If you press "disc 3", you will probably see that the other "disc 3" is still green (which is not possible since you only have one 3)... But anyway, if you press another "disc 3" again, nothing will change.

**Reflection and next step:**

It was not that easy to implement the GUI for the first time because it had the wild possibility of designing the program. I believe the biggest problem I faced is assigning the right functions to the button command, especially if you want to call a function with arguments. Also, it was hard to use the openpyxl library and edit a specific element.

The program was easy to follow and design since I already had experience coding using Python. If I have more time, I will make the game able to accept more than 2 players, improve the design and decoration of the interface, and give the user the ability to choose how hard the game (Mode) is, such as limiting their time when thinking of what disc to choose.

**References:**

* www.youtube.com. (n.d.). *Python GUI | How To Make A GUI In Python | Best GUI Framework In Python | Edureka*. [online] Available at: https://www.youtube.com/watch?v=\_PHJvjQJa3w [Accessed 26 Apr. 2022].
* www.youtube.com. (n.d.). *Python GUI’s With TKinter - YouTube*. [online] Available at: https://www.youtube.com/playlist?list=PLCC34OHNcOtoC6GglhF3ncJ5rLwQrLGnV.
* tutorialspoint.com (2019). *Python GUI Programming (Tkinter)*. [online] www.tutorialspoint.com. Available at: https://www.tutorialspoint.com/python/python\_gui\_programming.htm.
* JetBrains. (n.d.). *Download PyCharm: Python IDE for Professional Developers by JetBrains*. [online] Available at: https://www.jetbrains.com/pycharm/download/#section=windows.
* Peruke. (n.d.). *How to play*. [online] Available at: https://www.perukegames.co.uk/how-to-play [Accessed 26 Apr. 2022].