|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2-m-logo-word-ppt_1420714889 | | | | **Macromedia University of Applied Sciences** | | | | | |  |
|  | | | | | | | | | |
|  | | | | Course Title: Basics of coding  Name of Examiner: Prof. Dr. René Brunner and Prof.Dr Ahmet yldiz | | | | | |
|  | | | | | | | | | |
| **To be completed by students:** | | | | | | | | | |  |
| 303496 | | | | | |  | | B-UBr DT ATI 6e 23W | |  |
| Student ID number | | | | | |  | | Matriculation | |  |
| RAZANAMAHOLY | | | | | |  | | Marvyn | |  |
| Last name | | | | | |  | | First name | |  |
| The student work will be submitted as:  (Please fill in the letter X in the appropriate box) | | | | | | | | | |  |
|  | | | | | | | | | |  |
|  | X | | Individual work | | | | | | |  |
|  | | | | | | | | | |  |
|  |  | | Group work | | | | | | |  |
|  | | | | | | | | | |  |
|  | | | | | | | | | |  |
| **Does only apply to group work:** (Complete only if it is a group work)  **If you are submitting a group work, please list the first and last names of all group members. The names must be entered electronically by the respective group members themselves. By entering the name, it is confirmed that the student agrees to submit the paper in the present form.** | | | | | | | | | |  |
| If you are submitting a group work, please list the first and last names of all group members. The names must be entered electronically by the respective group members themselves. By entering the name, it is confirmed that the student agrees to submit the paper in the present form. Furthermore, by entering their names it is declared by the individual groups members to have created the project paper (in case of a group work: the part which the respective student has contributed to the paper and has marked accordingly within the paper) on their own, without the help of others. In the process, the student has not used any aids other than those cited in the listing of sources and literature. All passages taken either verbatim or in adapted form from publications are indicated as such. The work has not been submitted to another examination office in the same or a similar form. | | | | | | | | | |  |
|  | | | | | | | | | |
| 1) | |  | | |  | | 5) | |  |
| 2) | |  | | |  | | 6) | |  |
| 3) | |  | | |  | | 7) | |  |
| 4) | |  | | |  | | 8) | |  |
|  | | | | | | | | | |
| Assessment of group work:  (Please fill in the letter X in the appropriate box) | | | | | | | | | |
|  | | | | | | | | | |
|  |  | | I apply for an individual evaluation (i.e. each member of the group will receive an individual mark) | | | | | | |
|  | | | | | | | | | |
|  |  | | I apply for a group evaluation (i.e. each member of the group receives an identical grade) | | | | | | |
|  | | | | | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Berlin/ 05.09.2024 |  | RAZANAMAHOLY Marvyn |  |
| Place/Date |  | Complete First Name and Last Name |  |
|  | | |  |
| **To be completed by the examiner:** (Text area for the second examiner) | | |  |
|  | | |  |
| *Title Page for Digital Submission, Macromedia version 1/2020* | | |  |

Table of content

**Overview1**

**Repository2**

**Structure3**

**UML4**

**Conclusion 5**

**Overview:**

This Tic Tac Toe game is designed to be simple and functional, so it's accessible and enjoyable for players of all skill levels. The game is built on a simple functional programming structure, which makes the game logic nice and clean, efficient and easy to follow. A graphical user interface (GUI) using the Python library 'tkinter' was added to make the game even easier to use. This GUI framework makes the game look great and provides an intuitive, user-friendly interface that makes the whole gaming experience better.

This version of Tic Tac Toe lets the player play against the computer, which adds a bit of challenge and unpredictability. This feature is really engaging because it provides a dynamic opponent that can adapt to different strategies. The game has been designed to suit players of all levels, with three different difficulty settings. These levels are designed to offer a tailored experience, from easy mode for beginners who are just getting to grips with the game, to a more challenging hard mode for seasoned players who are looking for a real test of their strategic thinking.

The easy level is great for beginners as it's more relaxed and the computer makes moves that are less optimal, so they can learn and enjoy the game without feeling overwhelmed. The medium level makes the game more challenging, with the computer using more sophisticated strategies. This makes players think more critically and plan their moves more carefully. Finally, the hard level is designed to be highly challenging. The computer uses advanced tactics to compete at a level that can rival even experienced players. This tiered difficulty system ensures that the game remains engaging and enjoyable, regardless of the player's skill level.

On top of that, using 'tkinter' for the GUI makes it easy to create a visually appealing interface and also allows for simple customisation and enhancement of the game. The game's interface is designed to be simple and easy to use, with buttons that respond quickly and a clean layout that lets players focus on the game without any distractions. The idea behind this project is to find a happy medium between what the games needs to do and how it feels to play it.

**Repository:**

https://github.com/MarvynWesley/TICTACTOE\_Game

**Structure:**

**Imports**

• Tkinter: It is used to create the game's graphical user interface.

• numpy: For handling the game board as a 2D array

• Random: For generating random moves for the computer to play in different levels

**Functions**

***create\_board()***

The function initializes and returns an empty 3x3 tic-tac-toe board using a NumPy array.

This function returns: A 3x3 NumPy array filled with zeros.

***possibilities(board)***

The function identifies all the available positions on the board and returns the list.

Board: A 3x3 NumPy array showing the state of the game.

This function returns: a list of tuples, with each tuple showing the row and column status of an empty cell.

***computer\_move(board, player, level)***

This function runs the computer's move based on the selected difficulty level.

Player: it is an integer value (1 for human, 2 for computer) that shows who's playing.

Level: a string that indicates the difficulty level, which can be "easy", "normal" or "hard".

***random\_place(board, player)***

The function randomly selects an empty cell and places the computer's mark on the board.

***row\_win(board, player)***

The function checks if the player has won by filling any row with their mark.

• Returns:

1 if the human player wins.

2 if the computer wins.

1 if the game is a tie.

0 if the game is still ongoing.

***player\_move(row, col)***

This function handles the player's move by placing their mark on the board and checking for a winner. If the game continues, the computer makes its move.

Here are the parameters:

***update\_buttons()***

This updates the button labels on the GUI to show what the board looks like after each move.

***end\_game(winner)***

The function ends the game and shows the result (win, lose, or tie). It also disables all the buttons.

***restart\_game()***

This function resets the game board, enables all buttons, and clears the result label.

***set\_difficulty(level)***

The function sets the difficulty level for the computer's moves and restarts the game.

**GUI Setup**

The graphical user interface was built, and it includes the following components:

The main window is where you'll find all the main features. The root window is set up with a specific size and centered on the screen. I have ve set the background color to purple.

Title Label: **"Tic-Tac-Toe Game"** at the top of the window.

Now, let's look at the buttons. The 3x3 grid of buttons represents the Tic-Tac-Toe board. You can style the buttons with different fonts and colors.

The different labels: Once the game is over, it shows y the result ("You Win!", "Computer Wins!", "It's a Tie!").

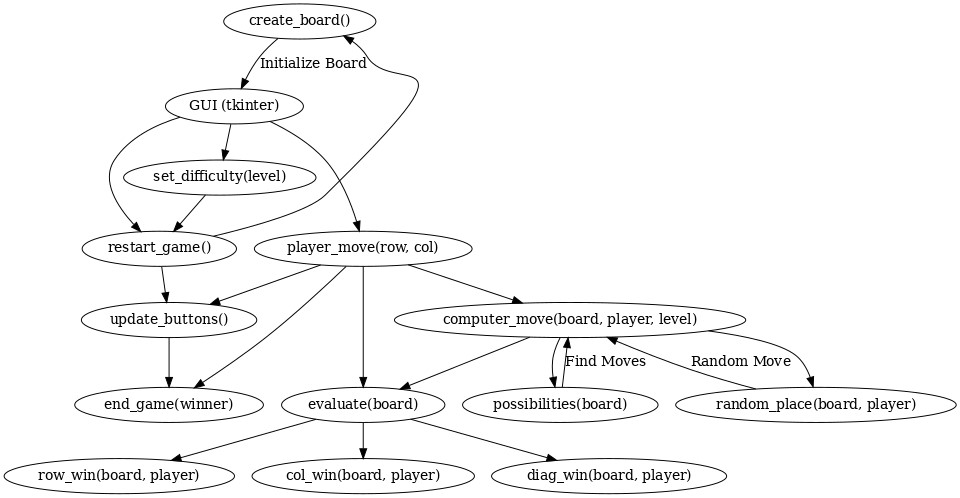
You can choose the difficulty level you want to play at. You can use the radio buttons to choose the difficulty level for the computer.

Restart Button: You can also restart the game at the same difficulty level.

**Main Loop**

***root.mainloop()***

It starts the event loop, which looks out for user interactions and updates the GUI as needed.

**UML: **

**Conclusion:**

In conclusion, it can be stated that…

The Tic-Tac-Toe game is a well-structured Python program that effectively combines simple functional programming with a graphical user interface (GUI) constructed using the tkinter library. The codebase is structured into discrete functions, each of which addresses a distinct aspect of the game logic. These include the creation of the board, the processing of player and computer moves, and the verification of win conditions. The UML diagram illustrates the flow and interaction between these functions, demonstrating how they collectively contribute to the game's functionality.

The program offers three difficulty levels (easy, normal, and hard), each providing a distinct challenge to the player by modifying the computer's strategy. This tiered difficulty structure ensures that the game maintains engagement for both novice and experienced players. The tkinter GUI enhances the user experience by providing an intuitive and visually appealing interface, thereby ensuring accessibility and enjoyment for players of all skill levels.



Sworn  
Statement

I, **Razanamaholy Marvyn**

Born on **24/10/2005**

Hereby declare that I have prepared this thesis independently and without external assistance. In doing so, I have not used any aids other than those mentioned in the enclosed list of sources.

All points that have been taken from publications literally or adapted form have been identified as such by me.

**.Berlin......**......................., ……**05/09/2024…**…………… ……….. …Dessins manuscrits
Dessins manuscrits


[Place] [Date] [Signature]