**Project Report**

**Aims:**

The aim of this project is to create a text-based adventure game of a lumberjack going through a deadly forest in order to find their cabin. It is also aimed to make the game interactive in order to give the user a better experience when playing it, this means I want to add some features such as loops and options to repeat the code in case the player wants to attempt the game again as they were unhappy with the result (died). I will also add a feature that will display how many times the player attempted each of the paths in order to let them know how many times they played the game or how many times they attempted each path.

**Analysis:**

In order to make the project successful and in good working order, there would need to be things added. Firstly, the introduction would need to be added in order to make a scene/ setting for the player to imagine. I added a definition lumberjack in order to create and recognise this user-defined function when used at a later point, inside the lumberjack definition I have included some variables in which the user will input their own choice and this would allow them to create their own character and this would make them more invested on the game. The user would then have to choose from 3 paths within a list in which the user can pick one and it will allow them to resume the game based on their input.

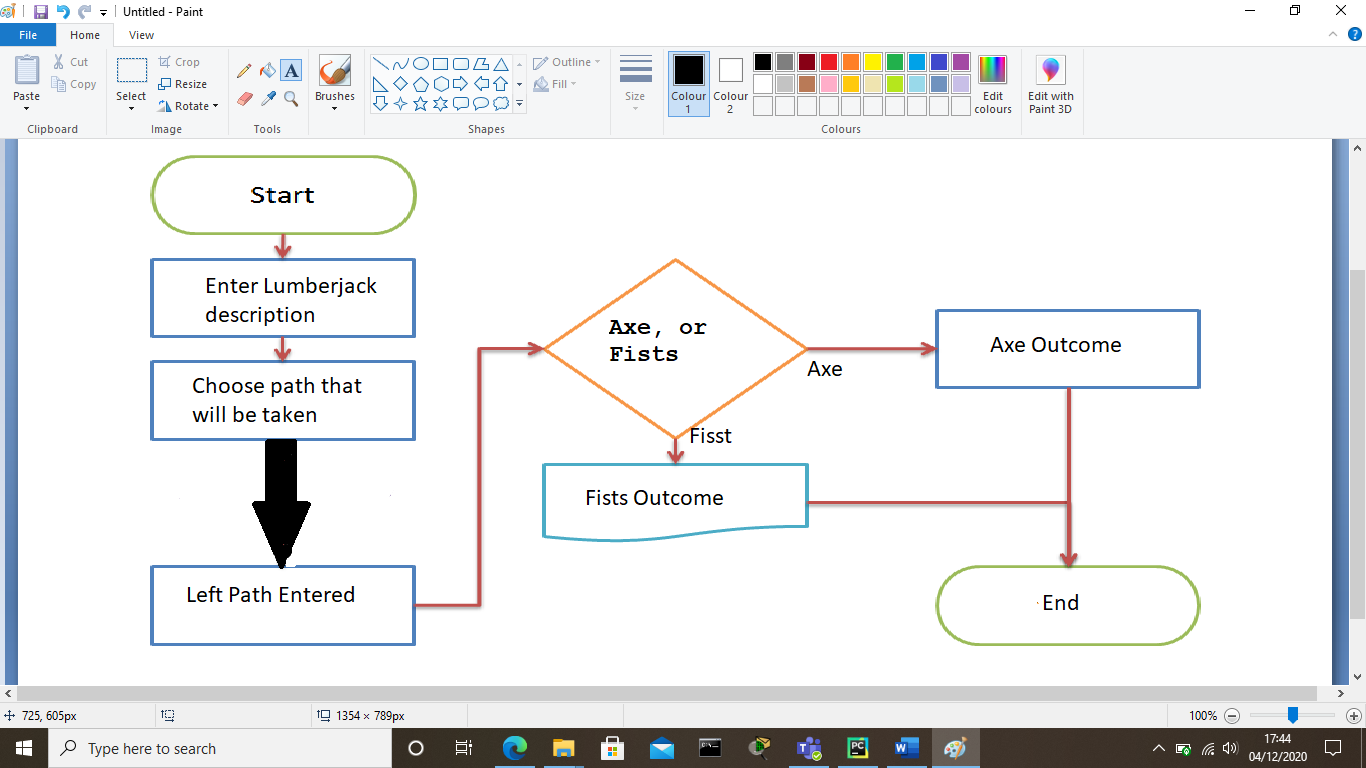
The left path is also defined to act as a whole function that goes within this path that was chosen. For example if the user chooses the left path they will be facing a bear in which they will have to fight with their chosen weapon in which they have to choose (The weapons include the axe they chose at the start or fists). If the player chooses the axe then there is an if statement that would run the code for the axe and will display the outcome of their choice and this is the same with the other two options which are fists and if they input a blank command. The left path is done within a different file and imported that file to the main fine of the game in order to separate the coding and make it clearer and more understandable (file handling used).

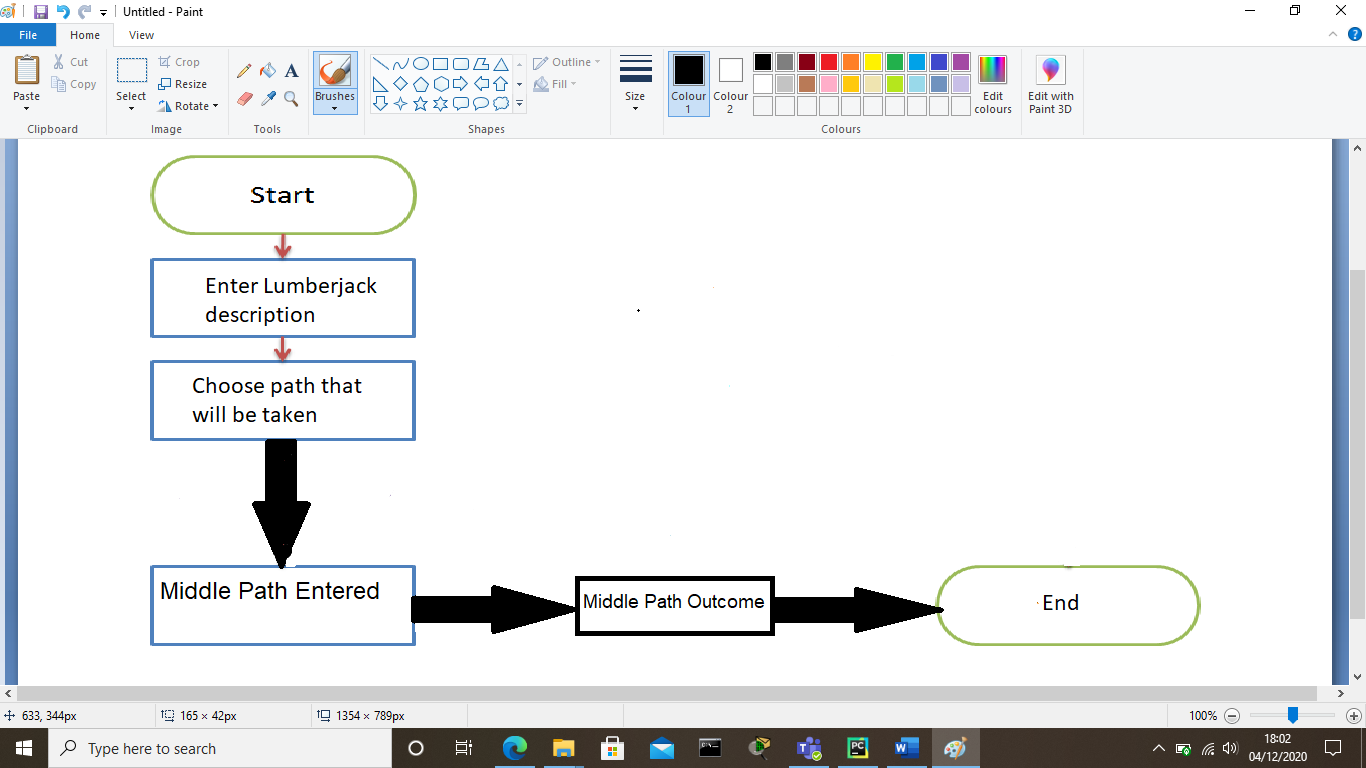
The Middle path is also defined to act as a whole function that goes within this path that was chosen. If the player chooses this path they are going to be introduced with the random function, this means that I have saved the 6 alternatives as variables and then saved each individual variable into a list in order for the randomise function to work properly and then the outcome would be randomised so every time the user chooses the middle path, it should have a different outcome within the six that I have included. This would give the user a sense of mystery and adventure as they wont know the outcome of the middle path if they choose it again and again as the outcome would be randomised and there is a chance of one out of six for the same choice to be displayed again.

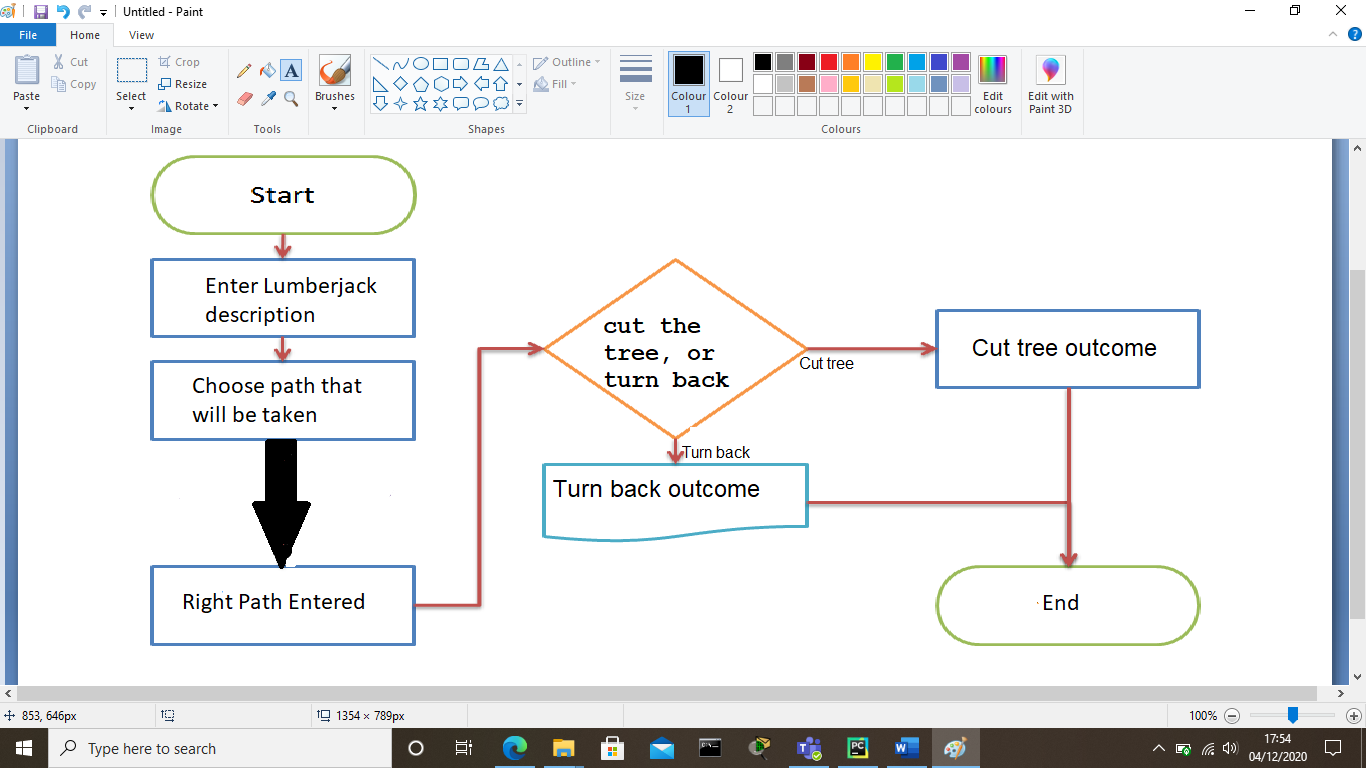
The Right path is also defined to act as a whole function that goes within this path that was chosen. On this path I decided to do the same as the left path, however, extend it and have a different scenario for the path. I used a variable to allow the player to chose what action they would take based on the scenario that was displayed on the screen, then I used an if statement to separate the actions into two different parts so depending on what the player chose the outcome would be based on the action they chose. If the player decided to turn back then they would have a randomiser function in which would give them a random alternative from which I have inserted. If the player chose to cut the tree, they would be met with a random action in which would allow them to have a different experience every time they try it again which would make it

At the end I added a while loop in which would allow the player to repeat the game in case, they were not happy with the result they have go from their first attempt and before it does it will ask the player if they want to replay and if they say no the loop will break. I also added a dictionary in order to save the character.

**Design:**

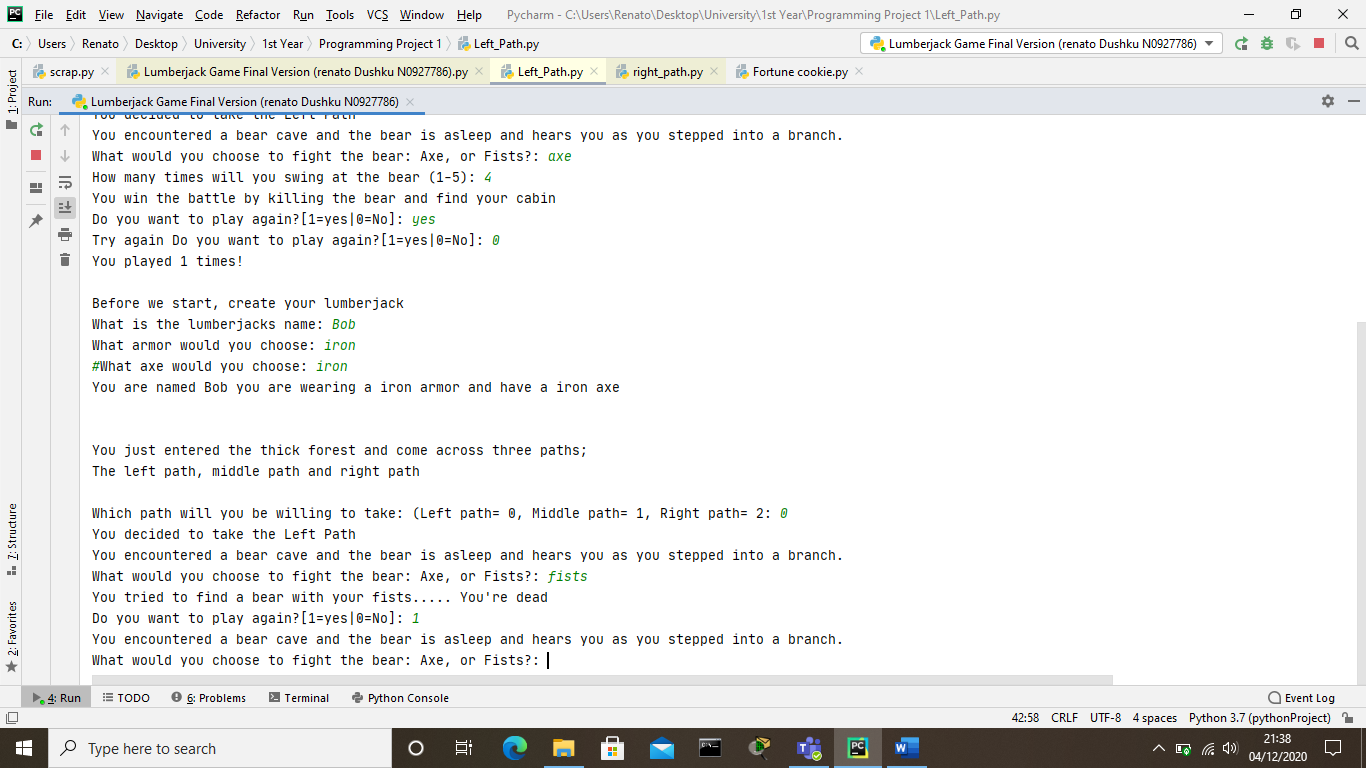


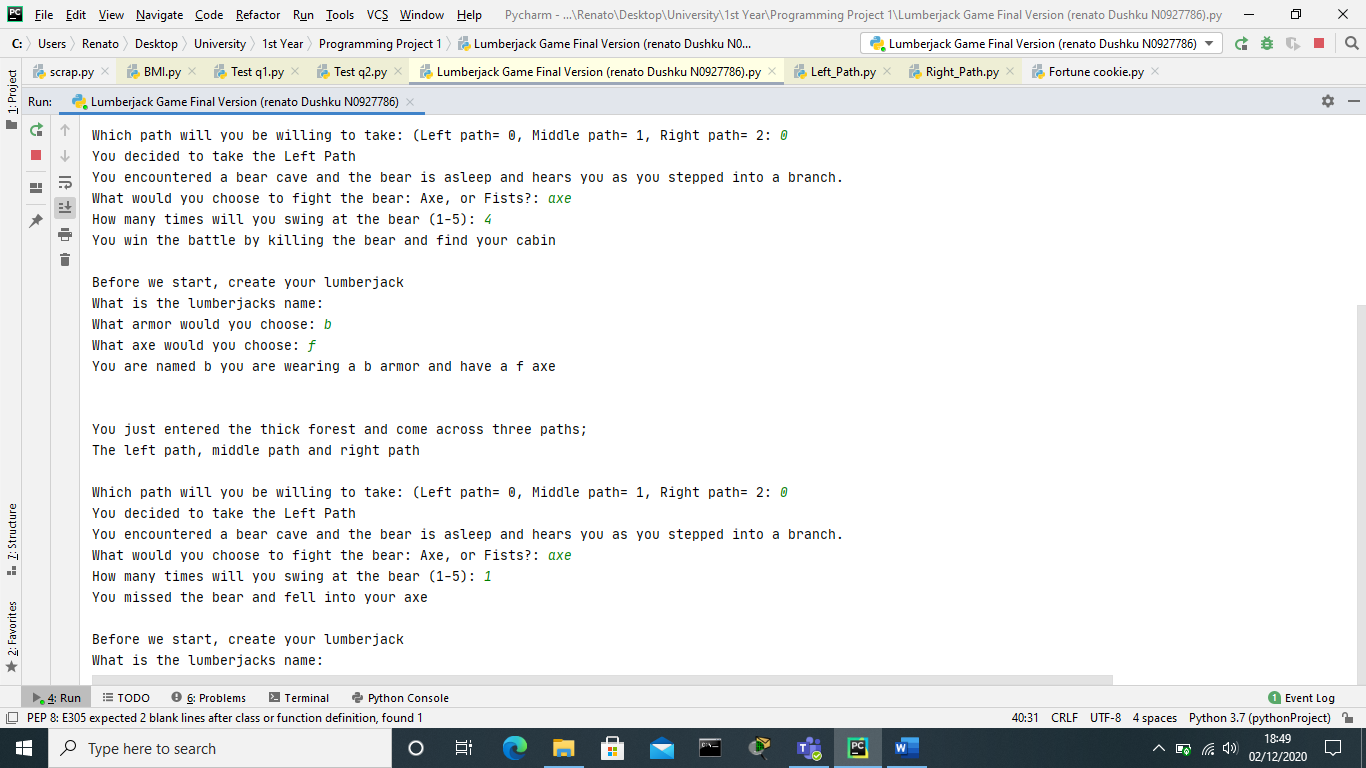


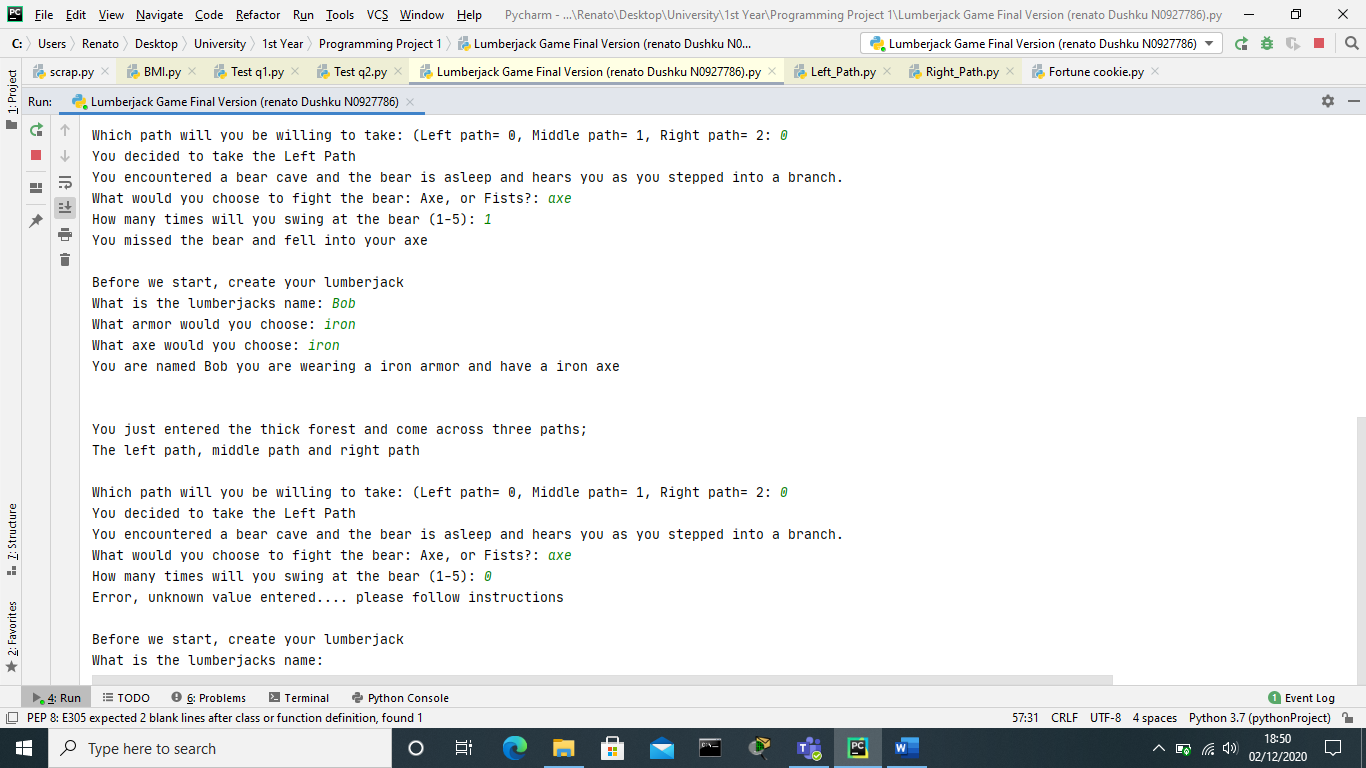


**Testing:** table of tests to be undertaken to prove that the program achieves its goals (1-2 pages)

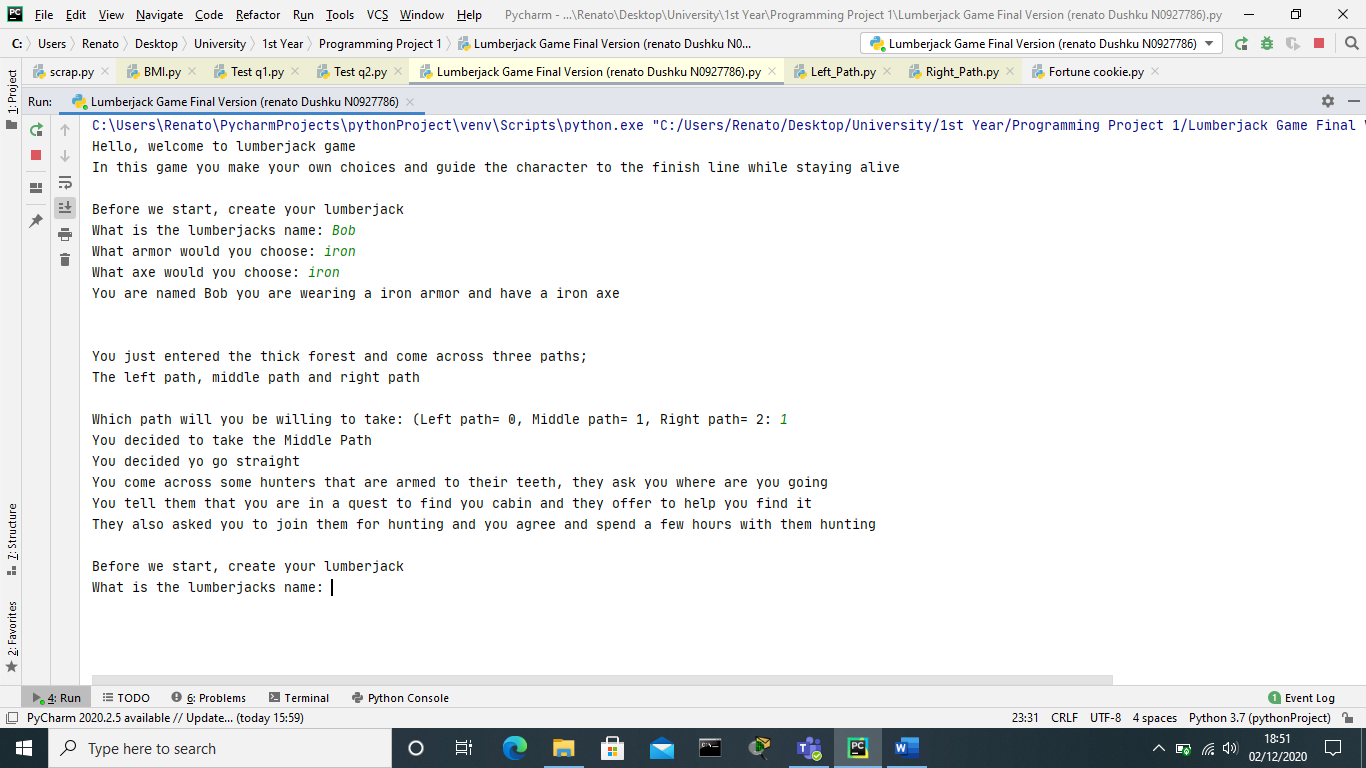
**Left Path:**

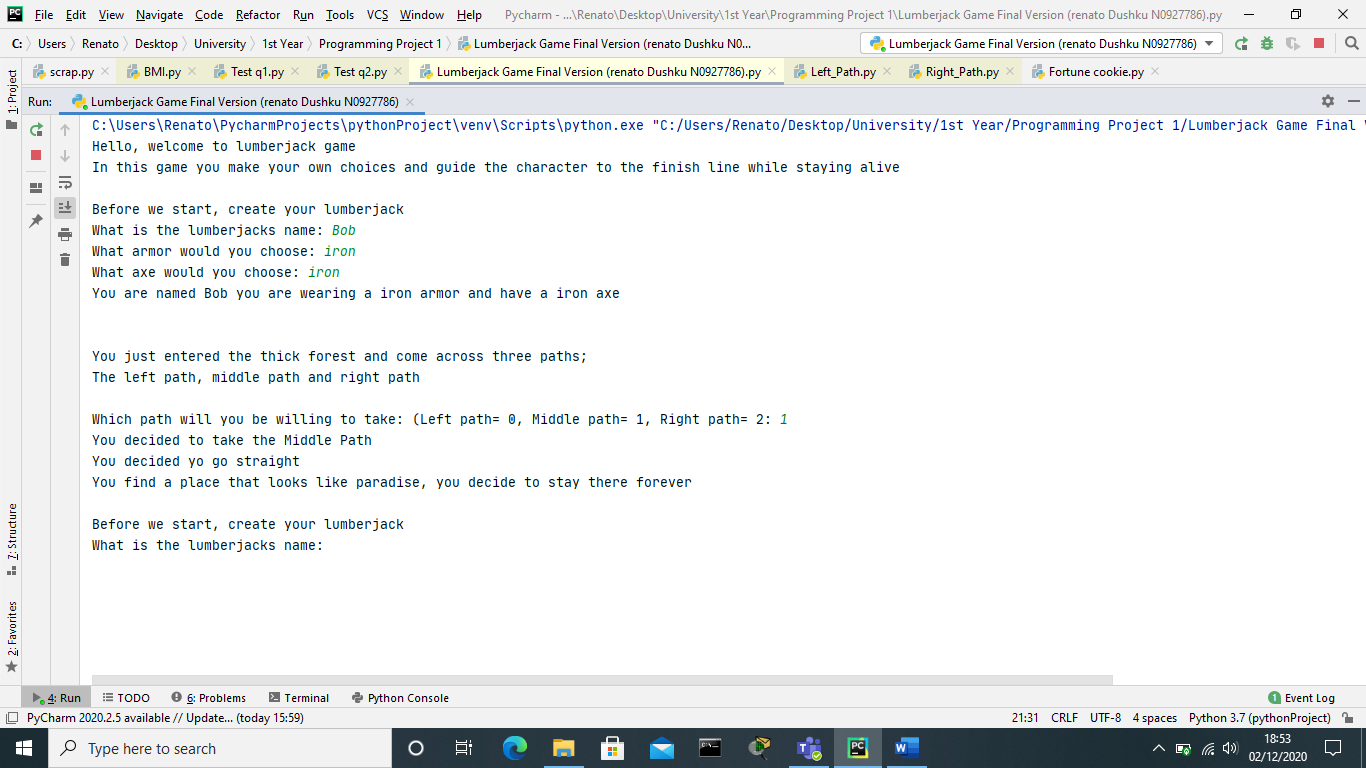




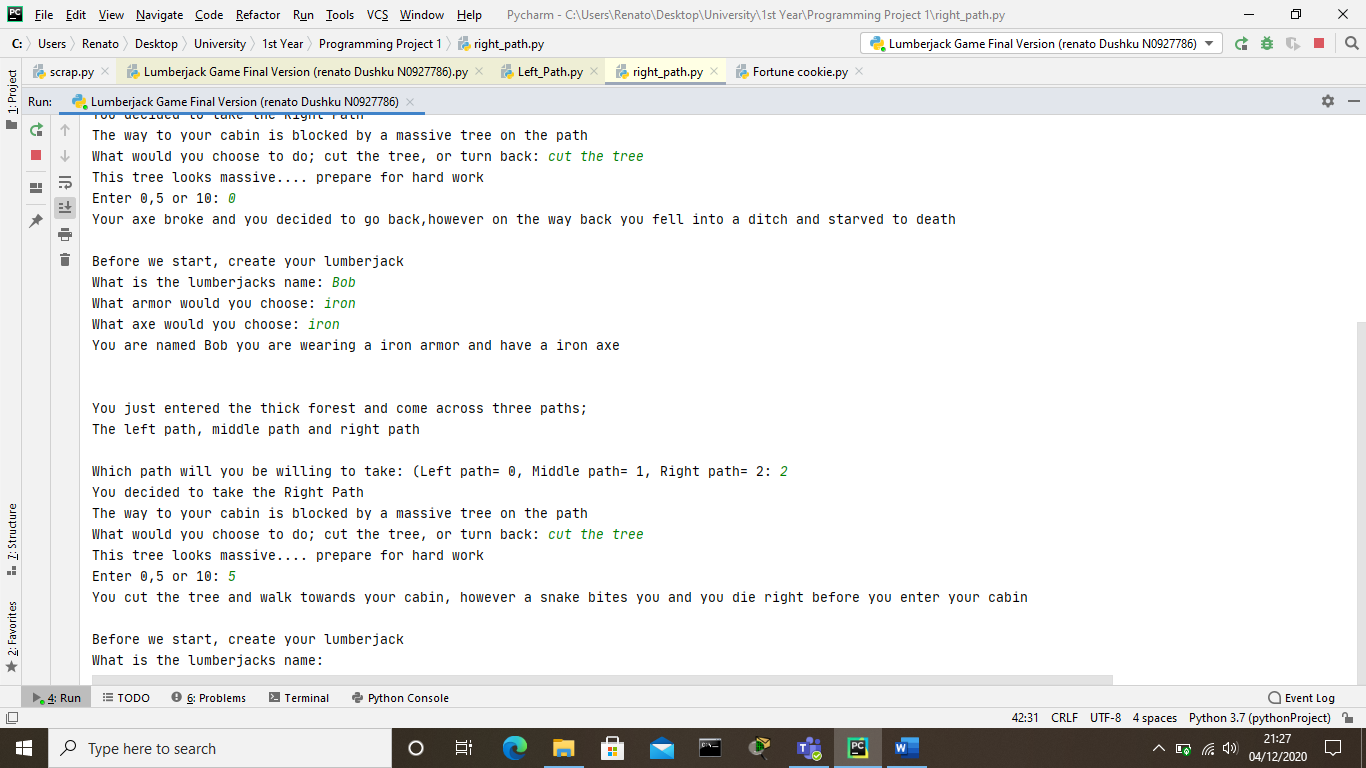


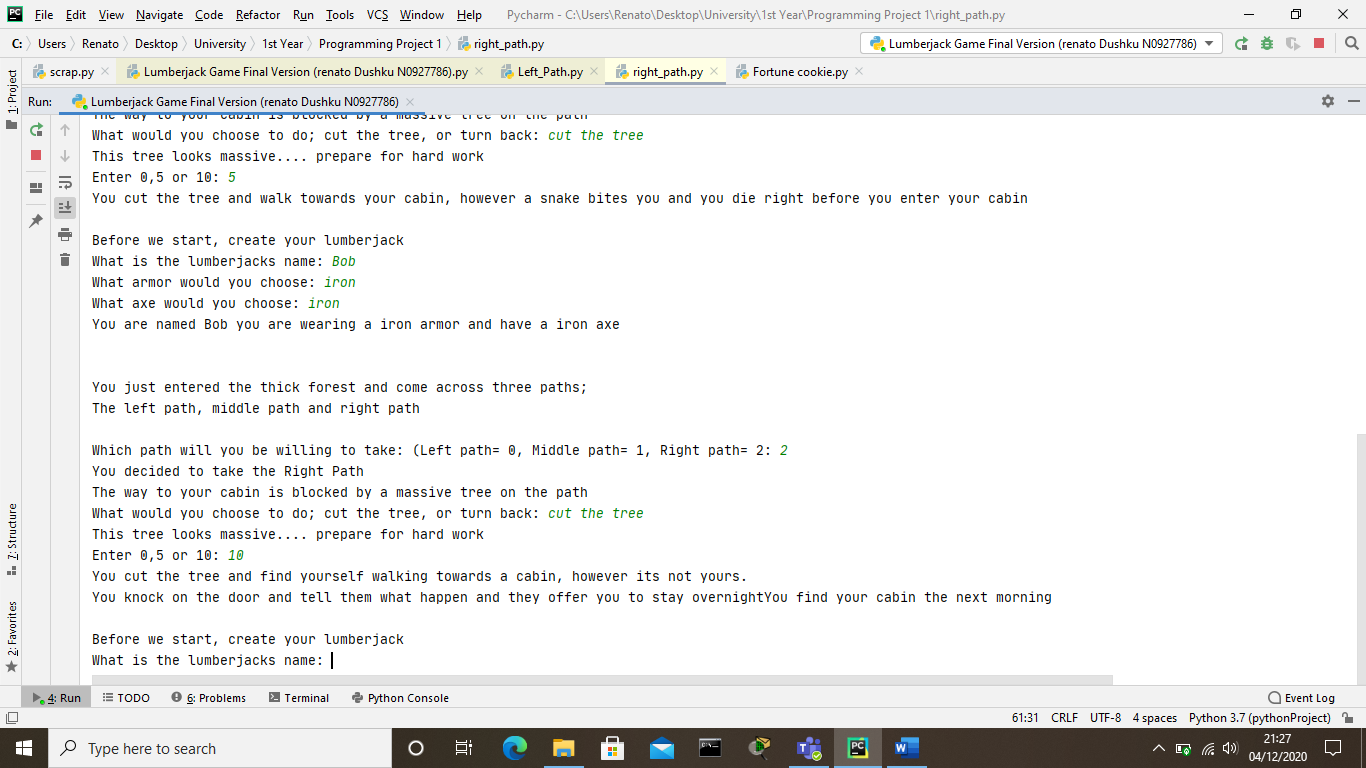
**Middle path:**





**Right Path:**





**Critique:**

On this project I had many obstacles that I overcame by doing my own independent study and I also used the resources that the university provide within the online workplace.

In my first attempt I used the print function to display the intro of the game and some instructions on how the user would be able to partake within that game so that they could play it. I used the input function to create the illusion that the user is choosing the path they wanted to take, however I inserted the random function so that no matter what path they chose to take the outcome would be randomised from a selection of “alternatives” that I inserted as variables and then saved them as a tuple in order for the randomise function to work properly and chose an outcome randomly for the user. After the outcome was displayed via the print function, the game ended. Overall, my first attempt worked without any errors, however it was too simple to be graded higher than a pass or even a pass as it mainly used the print statements and variables.

In my second attempt I did the same things as my first attempt, however in this attempt I tried to include loops, if statements etc. I started this of my setting some variables and giving them a value’s for their correct purpose in order to function properly. However, I had some difficulties making it work as I found out that a value of “left path” could not be set to a variable that would be then used in a loop, then I tried to re-do the variables by setting numbers as their values and it didn’t work as well. This obstacle stopped me from creating a fully functional loop that could allow the user to revisit the start if they die and also stopped me from adding a section at the start that the user could choose if they wanted to continue to the game or not. I did a third attempt and figured out the problem in which did not allow me to create the game in the way I wanted and then I finished the project. I also proceeded to add a while loop so it can repeat the game once the user dies or wins. The second attempt of improving my project also worked even with the file handling.

If I recreated this game I would make sure I draw an idea board before starting the design of the project as I added other features at the game and most of them were hard to implement as the code would needed to be changed in order for it to work perfectly and minimize errors.

**Credit/ Bibliography:**

W3schools.com. n.d. *Python Tutorial*. [online] Available at: <https://www.w3schools.com/python/default.asp> [Accessed 27 November 2020].

(Python Tutorial, n.d.)

Letstalkdata.com. n.d. *How To Write A Text Adventure In Python Part 1: Items And Enemies – Let's Talk Data*. [online] Available at: <https://letstalkdata.com/2014/08/how-to-write-a-text-adventure-in-python-part-1-items-and-enemies/> [Accessed 2 December 2020].

(How to Write a Text Adventure in Python Part 1: Items and Enemies – Let's Talk Data, n.d.)

Anderson, M., n.d. *Text-Based Fighting Game In Python 3.0*. [online] Code Review Stack Exchange. Available at: <https://codereview.stackexchange.com/questions/223917/text-based-fighting-game-in-python-3-0> [Accessed 2 December 2020].

(Anderson, n.d.)