**Python Project report**

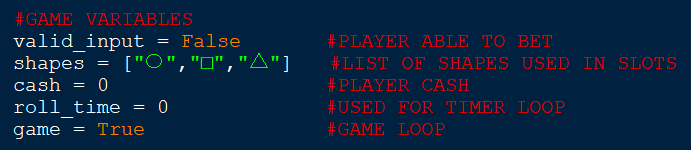
**Aim**

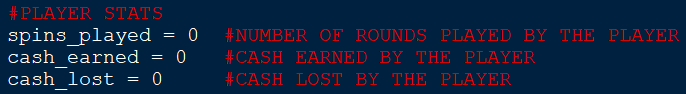
In this report I will be reviewing and analysing my python project. The goal of my project is to have a fully functional and working python application which can fulfil its goal, with zero to little obstacles in its process. What is my python Project about? For my python project, I have created a Text base Slot machine game. The user will receive, by default, a set amount of money and with that said money they can place a bet amount, decided by the user, the program will then randomize the output of the slots and depending on the circumstance the user will get double their money back or not get anything.

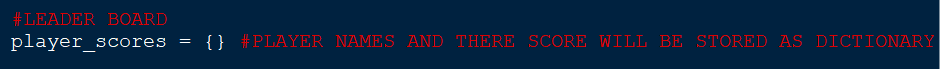
To accomplish the goal, it is essential that I make uses of different python skills. To store integers, strings and small functions, included but not limited to player cash, cash lost by player and more, for this I will need variables. For the program to run the game I will also need loops, this is crucial as it means I won't have to write repeating code, but I can instead have it loop over. As the program runs it need to examine if certain conditions are True or False and execute the corresponding outcome. Finally, my game will need to import, external text files, this will allow me to save game data externally oppose to on the program.

**Analysis**

Any basic program, on any programming language will need variables. For my program I grouped up my variables to make my code easier to read, for myself and others. I did this by commenting using the “#” feature, which ignores any lines of code which a “#” is typed before it, this is a useful feature to write comments on my work and to disable some lines of code for testing purposes.

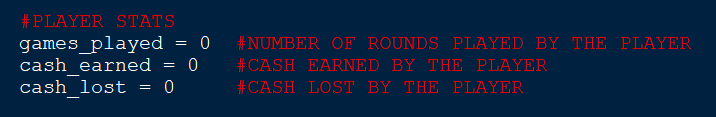
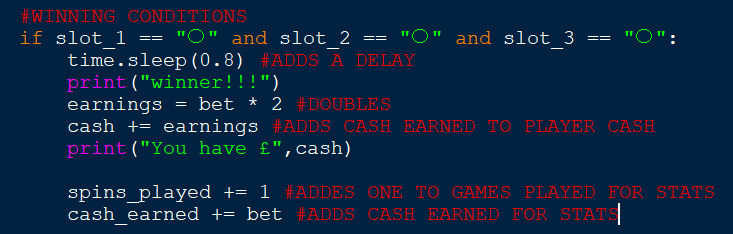
The First group of variables I have is called “GAME VARIABLES”, here I have grouped all the variables which have a role in the actual function of the game. The first variables I have in this group, In no specific order, is called “valid input” , this variable is set to False by default, unless changed. The variable “shapes” is a list, and contains 3 shapes, stored as strings they are the following: a circle, a square and a triangle. This is the list from where a shape will be picked and displayed to the user. Next a variable named “cash”, as suggested in the name, this varible which stores the players money, as an integer, which he/she uses in-game. “roll\_time” is a integer varible intergral for the “roll loop” , which will be discussed later in this essay, this variable keeps track of the amount of rolls. Finally, “game” is a Boolean variable, which is used in the game loop, the variable is set to True by default.

The second group of variables is player stats, here the variables record certain player actions that are later displayed as player statistics. These stats record for example the amount of times the player bet under the variable “games\_played”, the amount of money they player has made, as “cash\_earned “. finally cash lost, which records the amount of money the player has lost, “cash\_lost”. All these variables are set to 0 by default and are increase as the player does certain actions.

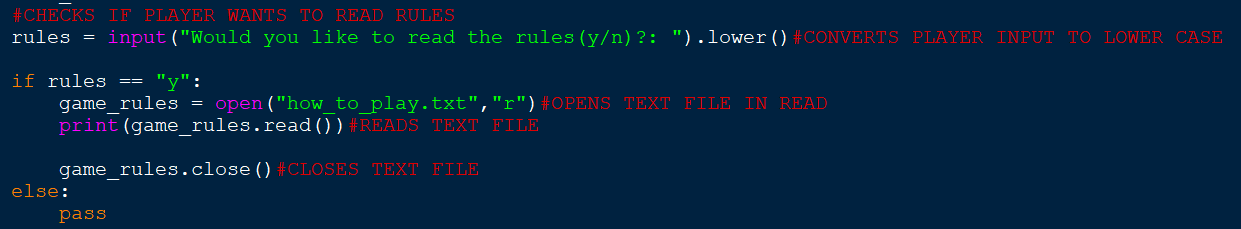
The final and smallest group of variables is the “player\_score” variable under its own group. This is a dictionary, which stores player names are strings and their associated total cash earned as integers. The values are displayed at the end of the program at the request of the player.

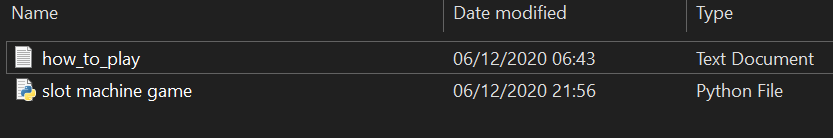
**Program requirements**

* Program should be able to keep track of player game stats:

To keep track of the players game stats I used variables for the statistics I want to keep track, having them set at Zero and increase or decrease depending on program outcomes. For example, one game statistic is money lost, so every time the player loses, their bet will decrease the game stat. This is done so that the player can see how they performed at the end of the game.

* Program should be able to print text file:

At the beginning of the game, I made the program ask the player if they want to read a guide on how to play the game, however I thought it would take too much space and look too messy having the guide stored in the code, I wrote the guide on a sperate text file and have the program print the .txt at the player’s request. Asking the player first would be more user friendly as it means printing the guide is optional and would make the terminal look less cluttered if the players says “no” because maybe they have already seen it.



* Program should be able to receive player’s name:

The program asks the player for their name and store it in a variable called “name”, this is done to keep track of the person associated with the score. However, I had an issue with the dictionary being case sensitive when storing name and the person associated score. Meaning that someone who entered their name twice one with capital and one with out would be treated as two different people and get two different score instead of updating their old score. To get around this I included a “.title()” this would automatically make the players inputted name start with a capital letter.



* Program should check if player has enough money to play:

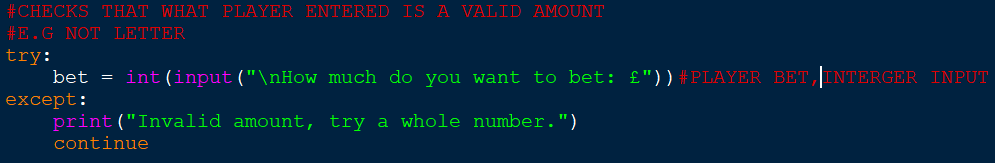
To make sure the player had enough money to bet, which in this case is, not zero, I added a while loop to counter this issue. This while loop was to loop as long as the bet was no equal or less than zero, in other words, if the player ran out of money the loop would break.

* Program should be able to pick a random shape from a list:

The way I got the program to randomly pick a shape for the slot machine was to first create a list called “shapes” which contained 3 shapes: square, circle and triangle. I then imported random and used the random.choice() function to have it select one of the shapes from the list.

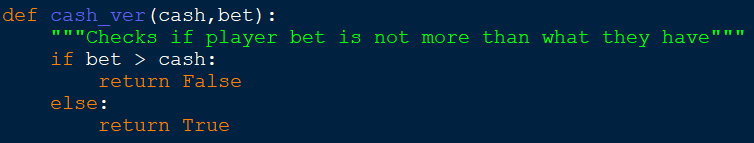
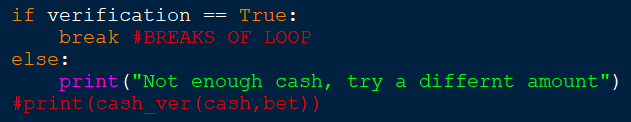


* Program should be able to receive player input in the form of a bet:

For the program to receive the players bet, I created an input variable called “bet”, to ensure what was inputted was a whole number I placed the input inside a int(). Once the program loops the player will have to enter a new bet and the one will be over written.

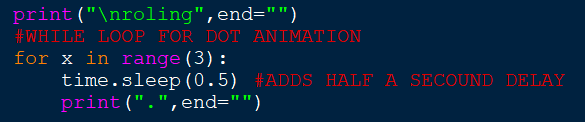
However since the players input had to be a integer due to the “int()” anything else would return with a error and end the game. To get around this I included the “try” and “except”, this would “try” the bet input and if any errors came up, the “except” would tell the user the input is invalid and to try again.

* Program should be able to check if player’s input is eligible to proceed to next step:

In an earlier version of my program the player’s bet was not bound by the total cash they had, meaning that as long as they had money, they could input a bet which would far exceed their total cash. To combat this amusing problem, I created a function which would first check that the bet is not more than the players cash and return either True or False and depending on the outcome let them proceed or re-enter the bet.

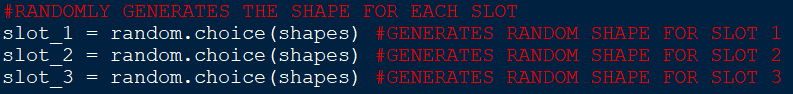
* Program should be able to include a delay and loading animation between bets:

To include a delay I first had to import time into my program, this would then allow me to use the time.sleep(x) function, which add a delay in seconds, depending on the number between the brackets.

To make a loading animation I first printed “loading”, next added a timer of 0.5 seconds and then printed “.”. instead of re-writing the code 3 times to get dot to appear 3 times, I place the code in a for loop with a range of 3, repeating the code 3 times. I placed the loading text outside the for loop, because during testing the word “loading” also got repeated. Nevertheless, I still wasn’t happy with the outcome, as I would prefer to see the dotes appear on the same line of text instead of each dote having new line. To overcome this I include “ end=”” “after the loading, this made it so that the “.” Would appear next to each other instead of on a new line.

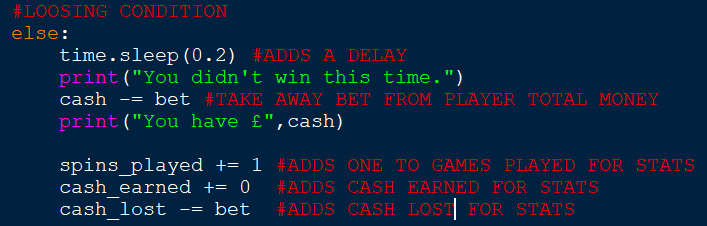
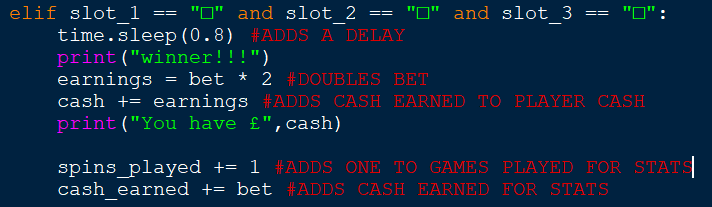
* Program should be able display to indepenadtly randomized shapes in each slot :

After importing random I was able to get my program to randomly select a shape from the list shapes, using the command random.choice(), this would be the base of the game. However this command would only give me one random shape and I need three, to get around this I created three variables called “slot\_1”, “slot\_2” and “slot\_3”. Each of these variables would independently pick a shape from the shapes list, which essentially gave me three random shapes.

Initially, I only had one variable called “slot” which used the random command and I just printed it three time, I quickly realised that the random shape it picked would also be displayed three times.

* Program should be able determin if the player has lost or won and take away or give the due amount:

To achieve this, I set created “if” and “elif” statements for all the victory conditions to determine if the player has won or lost. A example of a win condition would if the player get three or any shape e.g: “□”, “□”, “□” and “else” would be considered a loose condition as the shapes which were generated do not meet any of the win conditions.

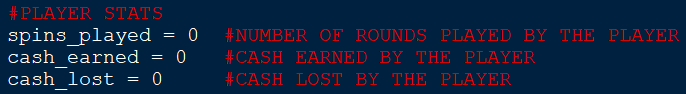
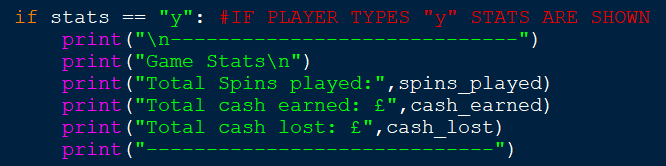
Once the program has determined that the player has won multiple things happen. First there is an artificial delay, then the program prints that the player has won, next the player’s bet is doubles and added to his current cash, which is displayed finally the round played is recorded under the variable “games\_played” along with the total cash earned under the variable “cash\_earned”.

* Game should end if player runs out of money:

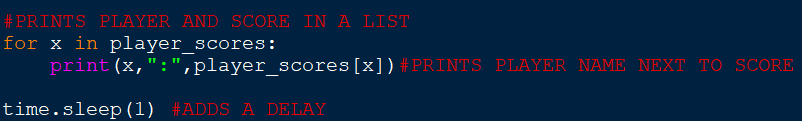
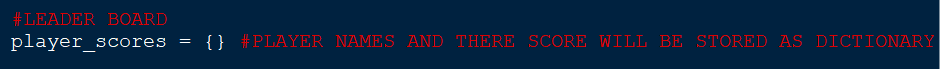
To ensure that the game ends when the player has run out of money, I placed the slot game in while loop, with the parameters that the loop will continue as long as the players cash is not equal or less than zero (cash >= 0).

Originally the parameter of the while loop was that it would break out of the loop if cash was equal to zero (cash = 0), however this caused problems when there was a glitch which causes the players balance to go into minus and the game would only end if the players cash was 0. The glitch led to a funny situation that if the player managed to get into a negative balance the game would never end. This problem was fix by making cash >= 0.

* Game should be able to show player stats :

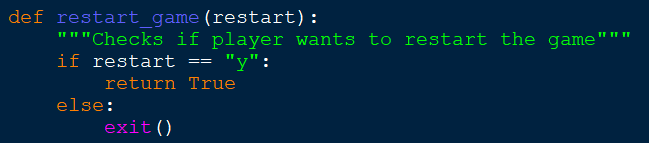
Once the player has ran out of money the game will effectivly end and the user will be promted if they want to see their game stats of their recent game. When the players agrees the program will print out the values of the variables associated with them, this includes: total spins, cash earned and cash lost.

* Game should be able to show player score boards:

A score board will essentially show the past players and their associated score. Initially wanted to add the player names and their scores to a list as a tuple, however this proved too long and trick to carry out, so instead I decide to store this data in a dictionary, so the player name is automatically associated with a score. The player name and their score would be added to the dictionary once the player ran out of money, with the players score being determined by the amount of cash he won.

* Game should be able to restart with fresh player stats and money:

The final question the player will be asked if they want to play a new, ther players response to this will be saved in variable called “restart”. To makes it so that the program restarts, I decide to make a function which takes the input of the “would like to play again?” and returns either “True” or “exit()”. If they player want to restart the function will return “True”, this means that the original game loop will reactivate and if the player dosent want to restart, the exit function will be run, effectivly killing the program



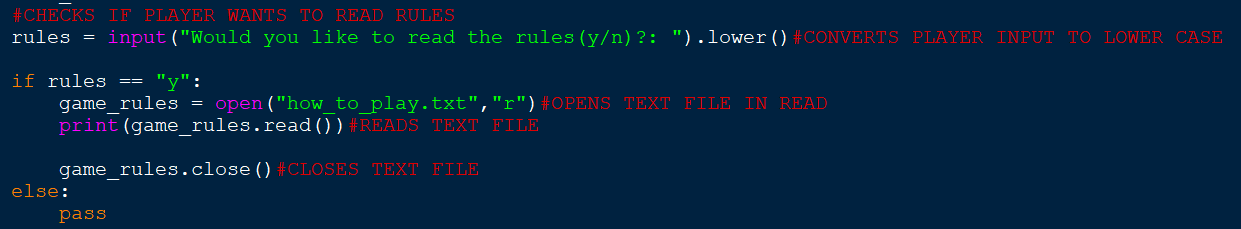
In a previous iteration of this program if the player wanted to restart the restart function I made for the game would return main(), which is the function which encapsulates the how program, the issue with this was that the program couldn’t save previuos player names and scores to the dictionary and disaplay them .

**Design**

My program is very long and has many complicated processes which work together along with other processes. However, for simplicity purposes I have divided my code into chunks from which I will have an easier time explaining and they are the following:

* Basic questions
* Cash loop
* Try and except
* Verification (Function)
* Winning condition
* See game stats
* Play game (Function)

Basic question design

Since my program is a text-based slot machine game, many of the ways to proceed is via player text input, this means that there are a few instances where the program needs to get the players with a question.

Start

Read rules?

Print rules

pass

Cash loop

This is loop is in place to check that the player still has money to continue after every spin.

end

Not Cash <= 0

Print player cash

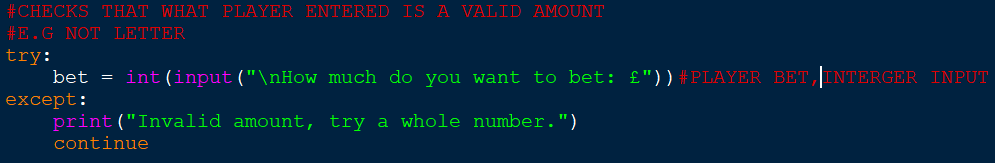
Game

Start

yes

no

Try and except

This process it to check that what the player has inputted a valid number.

Start

No

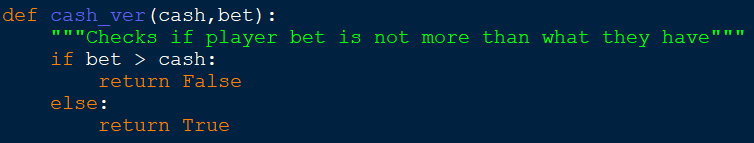
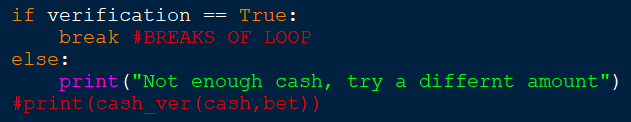
yes

pass

Is whole number

How much do you want to bet?

Verification (Function)

I created this function to check if the players bet is not more than what they currently have

Bet > cash

True or False

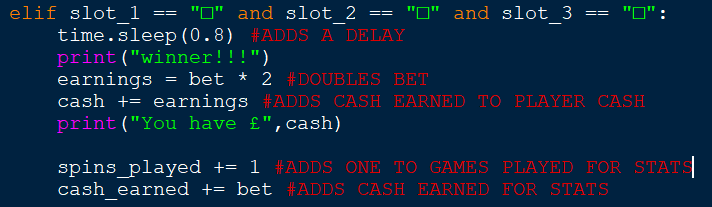
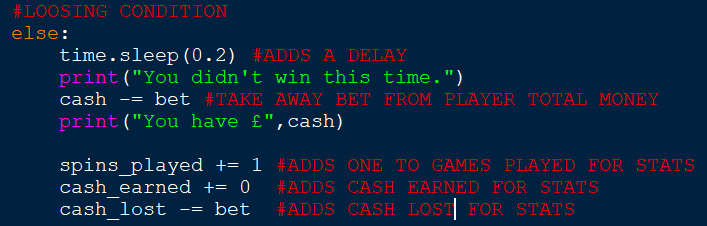
pass

Print error message

bet

Start

Winning conditions

These are a set of conditions that the players randomised slot need to follow so that the play can win.

bet

3 of the same shape

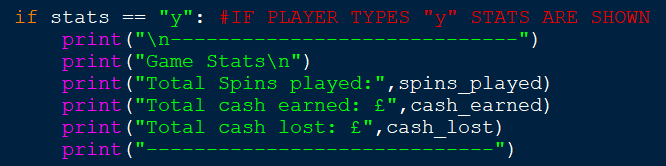
Start

Win bet

Loose bet

Game

See game stats



The player will again have the option to see his game statistics at the end of the game

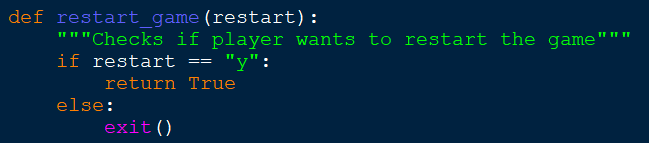
pass

Print rules

Start

See game stats

Reset game (function)

This is a function I created to determine if the player wants to play again. 

Yes

No

pass

Want to restart

Game

Start

slot machine game full flow chart

**Is bet more than player cash**

**Slot machine results**

**lose**

**Win**

**Cash 0 or less**

**Print player cash**

**Want to read rules?**

**Is bet a whole number**

**Player bet**

**Game loop**

**Start**

**If player wants to play again**

**Print game stats**

**Does player want to see game stats**

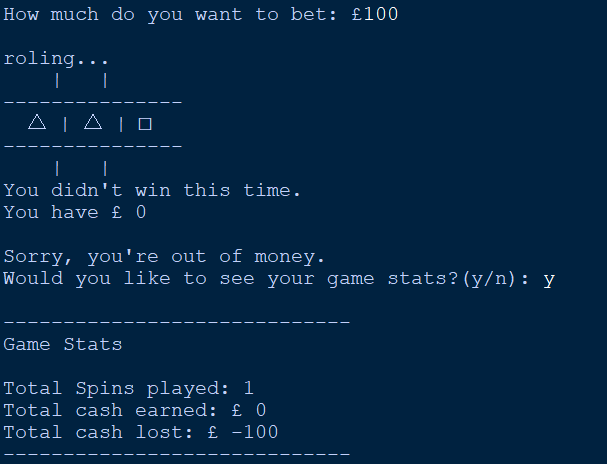
**Ask for name**

**Randomize slots x3**

**Print rules**

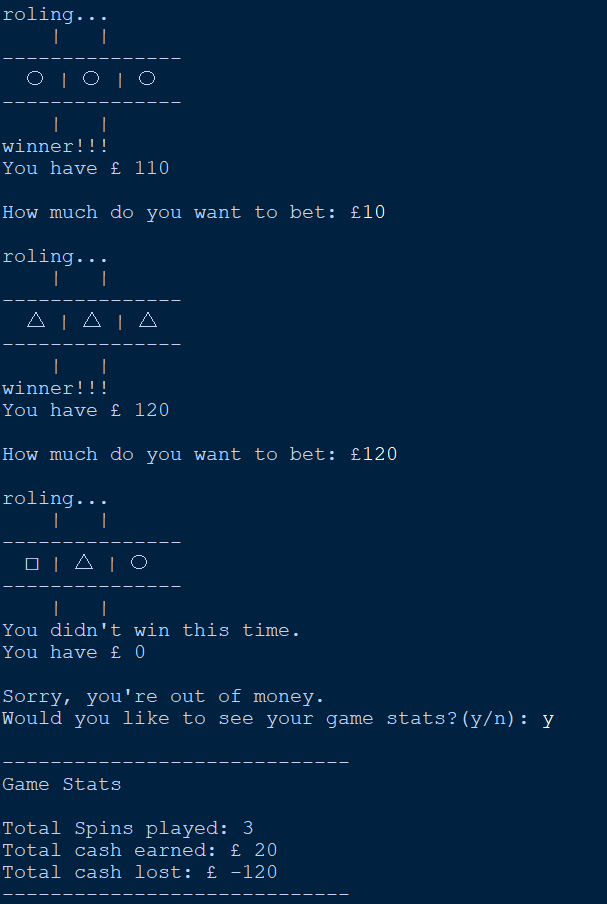
**Testing**

* Program should be able to keep track of player game stats:

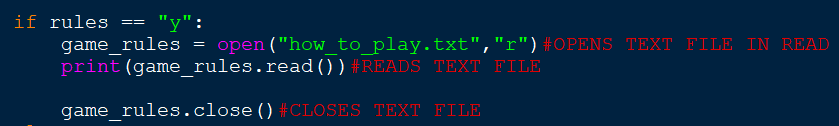


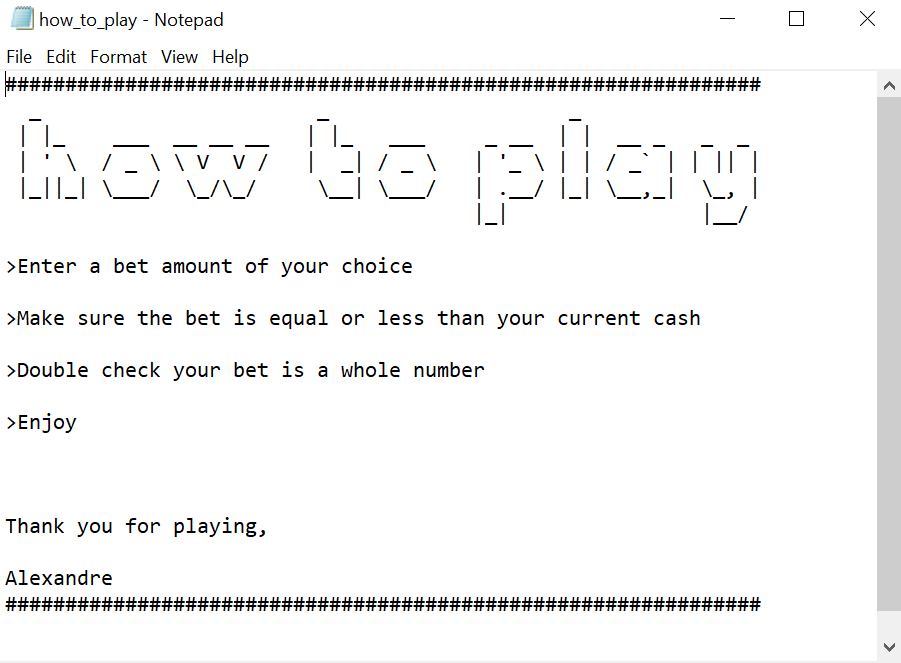
* £100 bet
* 1 spin

As we can here all the stats from the game where successfully recorded



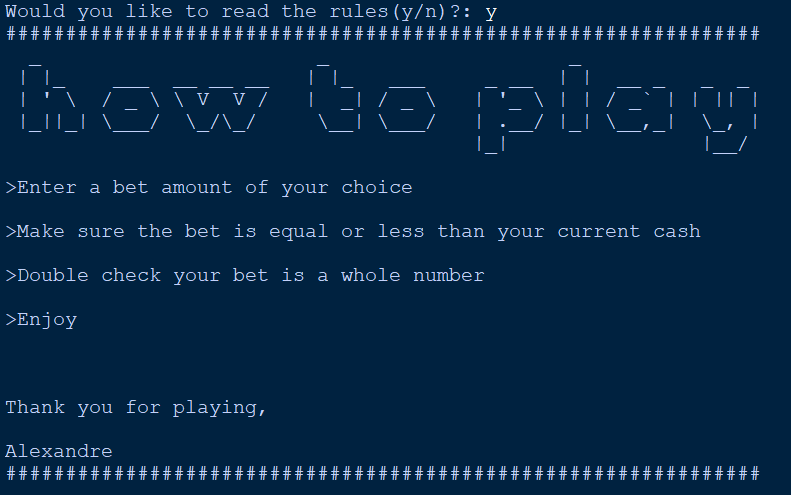
* 3 spins
* Won £20
* Program should be able to print text file:



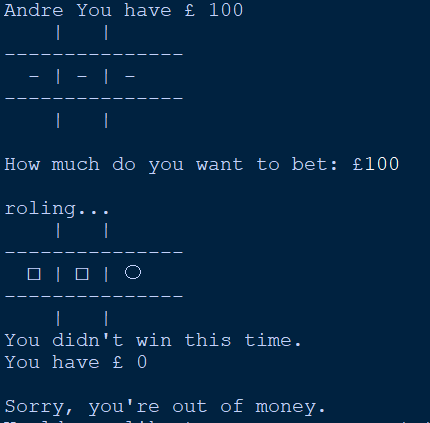


File name

Here we can see that the program is able to retrieve an external source of data, such as .txt and print it on the screen



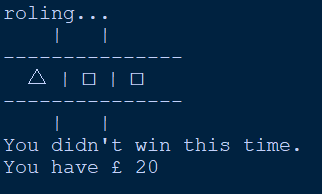
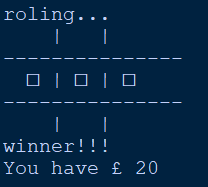
* Program should check if player has enough money to play:

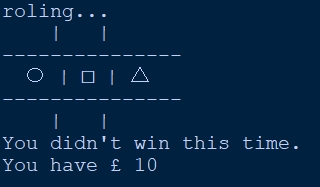


Here the player has spent all his starting money and once the player has spent all his money the game will Finnish

Here can also see that the player has bet all his money and lost they have zero money remaining.

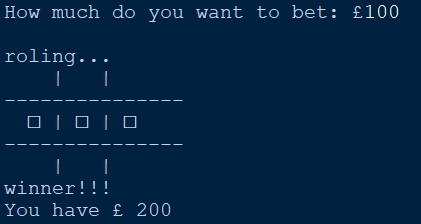
* Program should be able to pick a random shape from a list:





With each roll the slots display a different randomized shape.

* Program should be able to receive player input in the form of a bet:

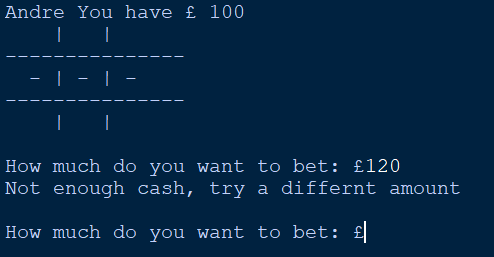


Player input

Game result

prize

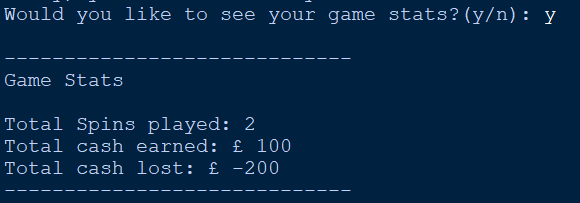
Here can also see that the player has bet all his money and won and doubled their money



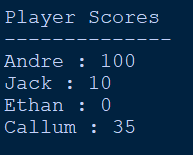
If the player tries to bet more money than he has the game will not let them do than and give the person a prompt, here we can see this feature works.

Here we can see that the try and except feature work, because when I try entering a letter or decimal, I get a prompt instead of a program crash.

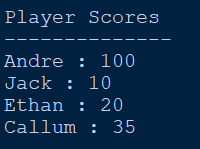
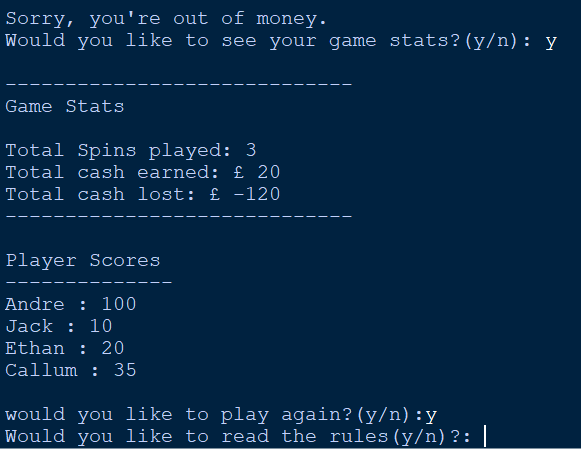
* Game should be able to show player stats :



Here we can see that when the player says “y” that plays game statistics are successfully shown.



Along with the game statistics the player score board is also shown, with the score of every player who has played during the session



If a previous player enters his name again and gets a different score the score will automatically be updated.

Here we can see that after shown the player statistics and they player accepting to play another game. The player will be bought right back to the start

**Critique**

Fortunately, most of the features I had originally planned where implemented in the final version of my game, however a few did not make it. Originally, I first thought of the idea of this game, I wanted my game to have visual interface with buttons and a proper spinning animation when each slot span to reveal the shape. Regrettably including these features was too complicated, I tried including a visual interface when I had my first draft of code for my program. Whilst trying to implement this, I quickly found myself overwhelmed and stuck on many aspects. It got to the point where I could not fully explain what most the code for the visual aspect did, this is when I decided to make my game text based and removed any reminisce of the visual interface, as including it would take me too long to produce a final project and would not be to the best standards.

Another issue I encounter came in the form of a glitch. Whilst testing the betting aspect of my game I found that when I would go “all in”, meaning betting all the money I had, the game would triple my reward instead of doubling it, if I were to bet normally. I quickly found the cause of the problem being that, once the bet was made the game would only take way that bet if you lost, but if you won by going “all in” the game would double your bet, all whilst you never actually gave away that bet, this essential tripled your money. I really liked the idea of risking it all and being better rewarded, but there were two problems. First, when the game gave you triple your money, one third of the earnings would not be recorded in the “score board” at the end of the program, which kind of defeated the purpose of the risk. And secondly, I had little time to fully implement this glitch as a feature with the deadline drawing closer and still having other things to test.

One aspect I would change in my program if I could do it again, is the way the program randomises the slots. As of writing this the program will pick a shape from a list for each individual slot and if all three slots have the same shape hurray you win! However, the problem with this is that the probability of me winning is at a set amount, around 3%, you may think that my program does its job and everything is good, and you’d be right. But this low percentage makes it a pain to carry out testing for win conditions, as I would have to sit here and actually play my own game waiting for a win to record my results to see, for example, if the game was rewarding the player the right amount. Next time I would want to have a set percent value of winning, probably around 8% double what it currently is, so it’s more enjoyable, and the shapes you see in the slots are shown if the value happened to fall in the 8%. This would mean I could change the win of the program on the fly if I need to for testing purposes, instead of doing what I had to do of adding, for example, more “squares” to my list of shapes, so the program is more likely to pick three squares and in turn I am more likely to win.

Finally, one last change I would make, is to make better use of functions. Whilst reviewing my code for the final time, I quickly realised that I could have save some time re-writing some lines of code by creating a function and running the data though it. One place this would have worked well, would have been the win conditions where at the end of these conditions the player would get plus one, for example, to spins which would then be shown at the end of the program in the form of a score board. I could have made a function would automatically add these points without me having to re write them for every “if”, “elif” and “else”.

|  |  |  |
| --- | --- | --- |
| number | uses | Referencing |
| 1. | I used this website to educate myself on how to use try and except in python for my project | W3schools.com. 2020. *Python Try Except*. [online] Available at: <https://www.w3schools.com/python/python\_try\_except.asp> [Accessed 29 November 2020]. |
| 2. | This is the website where I found out of the existence of the end=”” command in python. | utekar, s., 2020. *Python End Parameter In Print() - Geeksforgeeks*. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/gfact-50-python-end-parameter-in-print/#:~:text=Python%E2%80%99s%20print%28%29%20function%20comes%20with%20a%20parameter%20called,print%20statement%20with%20any%20character%2Fstring%20using%20this%20parameter.> [Accessed 1 December 2020]. |
| 3. | I watch this video to familiarise my self on dictionaries in python | *PYTHON DICTIONARIES & FOR LOOPS (Beginner's Guide to Python Lesson 8)*. 2020. [video] https://www.youtube.com/watch?v=IpbYrguvai0&t=561s: London App Developer. |