**Word Adventure Game**

## Documentation

Callum Wykes

# **Aims**

## Main aims and objectives

The program I am creating is a text based adventure game, the game will set in a fantasy setting and allows the user to choose how they want the story to progress. The story of the game will be written around the main character this is an unnamed character so any user of the program can put themselves directly into the story.

The main objective of this program is to print out the written story that accompanies it, it’s objective is also to give the user a story they can control which is both fun and challenging. The choices the user makes will affect the rest of the story, for example there will be one option where the user can choose to rest and eat or they could just skip these options. This would in turn mean that the characters would not have the energy to fight as well as they could leading to a game over. This will add a level of challenge to the game as well as making it more entertaining for the user.

## Programming aims

The story will be written in a text file which will then be imported into the program and saved into an array. By saving the text file into an array I can get the program to print out any specific line require for the story. The majority of the program will be in functions the reason for this is that the functions can then be called at different parts of the game allowing for the same scenarios to be played out at different times depending on the user’s choices saving time and mistakes during the programming process. The functions will also carry through the users choices saved in the form of variables, this would make changing the game based on the user’s choices easier and with less mistakes. The game will also use if statements for the user to enter their choices into which will allow for multiple inputs to result in multiple outcomes. While loops will be used so that a user can keep entering their choice just in case they enter an option that isn’t recognised by the program, this is especially useful in the program as multiple options will be required for the player to change the story to how they want it to be played out. For loops will also be used if there is a section that repeats itself and it can only be repeated a number of times until the story requires it to change.

# **Analysis of Requirements**

## Programming Requirements

1. The program should successfully open and read the given text file (providing that the file is in the same location as the program. If not then the program should inform the user and end.
2. Once opened the program is required to save the text file to an array so that individual lines can be called as the game requires them.
3. The program should split sections (introduction, opening/reading file, chapter 1 etc.) into different functions and call such functions in the correct order.
4. Some functions called will have to be called with variables that will be used later on in the program, these should be stated beforehand.
5. To display information to the user the program should use print statements, to display lines from the text file the program should access the array and state which line to print out.
6. For the program to receive the users input it is required to use input statements these will assign the users input to a variable.
7. The program will use a series of while loops and for loops which will allow sections to be repeated until an outcome is reached.
8. The program will use premade functions like time which will be used to break up segments of the story as well as sys which allows you to exit the program at will.

## Game requirements

1. This game should tell a story that makes sense no matter what options the user should pick.
2. As it is a game it should be fun to play and the user should want to play it again to find out all of the outcomes and combinations of choices.
3. It is required that the game should take place in a fantasy setting, this can be show by the presence of creatures and people typically associated with the genre.
4. The game will have choices for the user to pick which will change how the story is told, different outcomes will come from different choices.
5. A certain level of difficulty should be presented to the user so that the game will be challenging for the user to play, this level of difficulty can be presented in numerous ways such as hard choices in the game for the player to make.
6. Once the player has either won or lost the game should give them a chance to replay it so that they can retrace their steps easier, this will also prevent frustration from the player.

# **Design**

## Design brief

The design of the program is split into two parts: the program (which prints out the story based on the users input) and the story itself (what is displayed to the user).

The program will consist of functions, while loops, for loops, if statements, variables and print statements, these are aspects of the programming language, python, which I already know how use and implement in the game. When thinking of how to handle a file in the program I was faced with a problem, I originally thought about simply opening the file and using a function to skip lines and read only certain lines before skipping to the next, this would be be too complicated and time consuming to do each time I wanted to print out another line to the story. Since then I have been coming up with ways to do this more efficiently, one method that stood out is by saving the whole text file into an array, by using a print statement I can print out any line from the text file that the story requires at any time in the program. Another part of the program is the that sections of the story are broken up by a few seconds for the user to be able to catch up with reading, a way I could do this is by importing the time function and making the program sleep for the required amount of time I want until the next section of the story is ok to be printed out. This would make separating parts of the story significantly easier which would mean the user doesn’t get overwhelmed by all the text on the screen.

The story for the game will be set in a fantasy setting, it will have elements of danger and victory as well as some interesting places and creatures. The story must be written in multiple parts so that it can change to what decisions the player makes, this means that multiple scenarios will have to be thought out fully before starting to write the story. The game will revolve around the main character who will remain nameless so that whoever is playing the game can imagine themselves fully in the characters shoes this is important as they will think more about the decisions they make thought the story. The main character will also have a team who will be traveling with, these team members will help influence the main decisions of the story some scenarios will involve these teammates being in danger with the player having to choose to save one or the other.

## Design flow chart

Program Start

Start Game

How to play

Game instructions

exit

Display menu – gives choices: start game, how to play, exit

Open and read the text file that the story is written in.

File missing

Inform the user of missing file

File found

Save the contents of the file to an array.

Print out the introduction to the story

Print out section of the story

Decision that the user has to make

Death

Displays appropriate game over message to the user

Finished story?

No

Yes

Prints out a thank you message and the outro

End program

(This flowchart is not fully representative of the final program as there will be a lot of choices for the user will make that will not only have two outcomes)

# **Testing**

## Test Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. | Test description | input | Expected outcome | Actual outcome | Evidence of outcome | Changes implemented |
| 1 | This test will find out if the menu displays the instructions to the user when asked for. | Type “1” into the menu screen. | The program will display the how to play section. | The actual outcome was as expected. |  | None |
| 2 | If the program will end if the user selects the exit option from the menu. | Type “3” into the menu screen. | The program will display the outro and end. | The actual outcome was as expected. |  | None |
| 3 | Test to find out if the program will load the requested text file once the user has selected the start game option from the menu. | Type “2” into the menu screen. | The program will be able to open the file and print out the start of the story. | The actual outcome was as expected. |  | None |
| 4 | Test to find out what the program would do if the requested file is not present in the same location as the program. | Type “2” into the menu screen without the text file being present | The program will report an error in loading the file and inform the user before ending. | The actual outcome was as expected. |  | None |
| 5 | While the story is progressing the user will be able to make choices that will affect how the story is told and will result in different outcomes. | On the first user input type “rest”. | The program will play chapter 1 in which the player has the opportunity to rest and eat to conserve energy. | The actual outcome was as expected. |  | None |
| 6 | While the story is progressing the user will be able to make choices that will affect how the story is told and will result in different outcomes. | On the first user input type “carry on” | The program will skip chapter 1 and move straight into chapter 2 but the user will be at a disadvantage as they have not rested or ate which will mean the characters will be more prone to death in chapter 2. | The actual outcome was as expected. |  | None |
| 7 | When an option is presented to the user the user could type anything they want, the program should only allow the predetermined choices for the user to be able to continue. I will test this in three sections of the story just to be sure. | Input “word” into any user input section of the program. | The game will give the user a chance to repeat the input until they enter a valid word to continue on with the story. | The actual outcome was as expected. |  | None |
| 8 | When an option is presented to the user the user could type anything they want, the program should only allow the predetermined choices for the user to be able to continue. I will test this in three sections of the story just to be sure. | Input “word” into any user input section of the program. | The game will give the user a chance to repeat the input until they enter a valid word to continue on with the story. | The actual outcome was as expected. |  | None |
| 9 | When an option is presented to the user the user could type anything they want, the program should only allow the predetermined choices for the user to be able to continue. I will test this in three sections of the story just to be sure. | Input “word” into any user input section of the program. | The game will give the user a chance to repeat the input until they enter a valid word to continue on with the story. | The program did not perform as expected as the sentence displayed after the input was repeated until the program was forced to end. This was due to a problem with indentation of the while loop. |  | I located the part where the while loop and it now works as expected. |
| 10. | One part of the program is that the user’s decisions may come back into the story one such decision is involving rope and if you use it earlier on means that you won’t be able to save one of your team mates in the final section of the story. | The user should choose to sleep on the tree and “use” the rope then continue. | If the user has used the rope they will not be able to use it again therefore the teammate will die. | Actual outcome worked as expected. |  | None |
| 11. | One part of the program is that the user’s decisions may come back into the story one such decision is involving rope and if you use it earlier on means that you won’t be able to save one of your team mates in the final section of the story. | The user should choose to sleep in the cave or on the floor then continue. | If the user has not used the rope they then will be able to use it again therefore the teammate will be saved by the user. | Actual outcome worked as expected. |  | None |

## Evaluation of tests

Out of all of the test conducted the program passed the majority, those that weren’t passed were fixed after. There are too many different aspects to the story because of the decisions to test and document each and every one of them but a sample size was documented to show what the larger amount of code does.

# **Critique**

## Programming requirements table

|  |  |
| --- | --- |
| Requirement number | Requirement met? How? |
| 1 | This requirement has been met. As shown in the testing section of the documentation the program opens the relevant text file. If the text file is not located it informs the user before ending the program, this is also shown in the test table. |
| 2 | This requirement has been met. The program save the entire text file to an array known as story\_array. Each individual line can be printed out using a print statement such as print(story\_array[26]). |
| 3 | This requirement has been met. The final program is made up of different functions that all call up one another as the story progresses. |
| 4 | This requirement has been met. As some aspects of the earlier functions are required in later parts of the story some of the variables saved are passed through to other functions when they are called. An example of this would be if the rope has been used or if the characters have rested or eaten. |
| 5 | This requirement has been met. As discussed in requirement 2 the program uses print statements to print out required lines of the story. |
| 6 | This requirement has been met. When the user is required to make a decision their input is assigned to a variable using input statements, the inputted data in the variables then can be used to change the story. |
| 7 | This requirement has been met. As stated in the requirement the program uses a series of while loops and for loops to repeat necessary sections of the story as well as making sure a correct value is entered when asking for a decision. |
| 8 | This requirement has been met. The pre made functions stated in the requirement description are used in the program, time is used to make the program sleep to break up the text whereas the sys.exit function allows the program to end at any required time. |

## Game requirements table

|  |  |
| --- | --- |
| Requirement number | Requirement met? How? |
| 1 | This requirement has been met. This game tells the story of a hunter trying to avenge his family and village from a pair of monsters who attacked them, all of the possible combinations of options work and tell a story that makes sense for the character the user tries to portray. |
| 2 | This requirement has been met. I have had fun while writing and playing through this story, there are different ways to win which makes it interesting an example is that some teammates could die or none of them can. There are a lot of ways to die in this story which makes it challenging and engaging. |
| 3 | This requirement has been met. This game is set in a fantasy world called fezzintia, it feature humans living alongside magical beasts and monsters, in this story there are monsters that hurt the main character and their quest is for vengeance on these monsters. There are several beast that you can fight with your sword, this makes the action more intense as certain monsters have special features, attacks and abilities. |
| 4 | This requirement has been met. The story has many parts where the user can decide how they want the story to progress in the form of the main characters actions, these choices will affect the overall story. For example the user can decide to rest or not to rest this will as a result mean time will pass more quickly but the characters have less energy to fight and will be overpowered and beaten easily |
| 5 | This requirement has been met. The game is very difficult, there are a lot of ways to die which means that the player will really have to think about what they are doing before they decide on their choice. |
| 6 | This requirement has been met. Once the player has finished the story (weather they have died or they have won) the game gives them an option to replay. This is especially useful when the user has died as they would want to retrace their steps and find out where they went wrong. |

## Final evaluation

This game has been very fun to make as it has required me to think of a story, I have tried to think of unique ideas for the story as well as an interesting setting for it all to take place.

The main objective of the program is to print out the story in the correct order while printing out the corresponding sections to what choices that the user makes, the program achieves this as the user can run thought the entirety of the story using only the program. As well as this it effectively processes all of the user’s choices and changes the direction the story goes in based on these decisions. I am particularly proud of the arrays which I had to do some outside reading and practicing to implement fully but I managed to get them working properly, another part of the program I had to do some research on was the time function as I had not used this function in before. One part of the program which I enjoyed testing the long lasting decisions that run thought the program as I enjoy games where your choices catch up to you.

Some parts of the program I had trouble with was making sure that all of the possible choices worked. Sometimes when programming one mistake could mess up the game, timing was another barrier to my work as I spent too much time writing the story then working on the program, this meant that I could not print as much of the story out as I wanted to reducing the length of it by quite a bit. At first finding out an efficient method of accessing the lines of the text file proved difficult as my original plan (talked about in a previous section) would have taken a lot longer and proved to be less effective than using the arrays which I thought of later.

Some things I could have improved on in this program are down to the layout of the program, after finishing it I wanted to find out a way to deduce the number of while loops and if statements but I couldn’t find anything in time that would make the program not result in an error every time.

Overall, I am very happy with how the word adventure game turned out.