Liam Neate 2020 (N0919032)

COMP10082

Python Project

Contents

[1). Brief Project Idea/Aims of Program 1](#_Toc58181267)

[2). Analysis of the requirements of the program 1](#_Toc58181268)

[3). Design 3](#_Toc58181269)

[4). Testing 6](#_Toc58181270)

[5). Critique 6](#_Toc58181271)

# 1). Brief Project Idea/Aims of Program

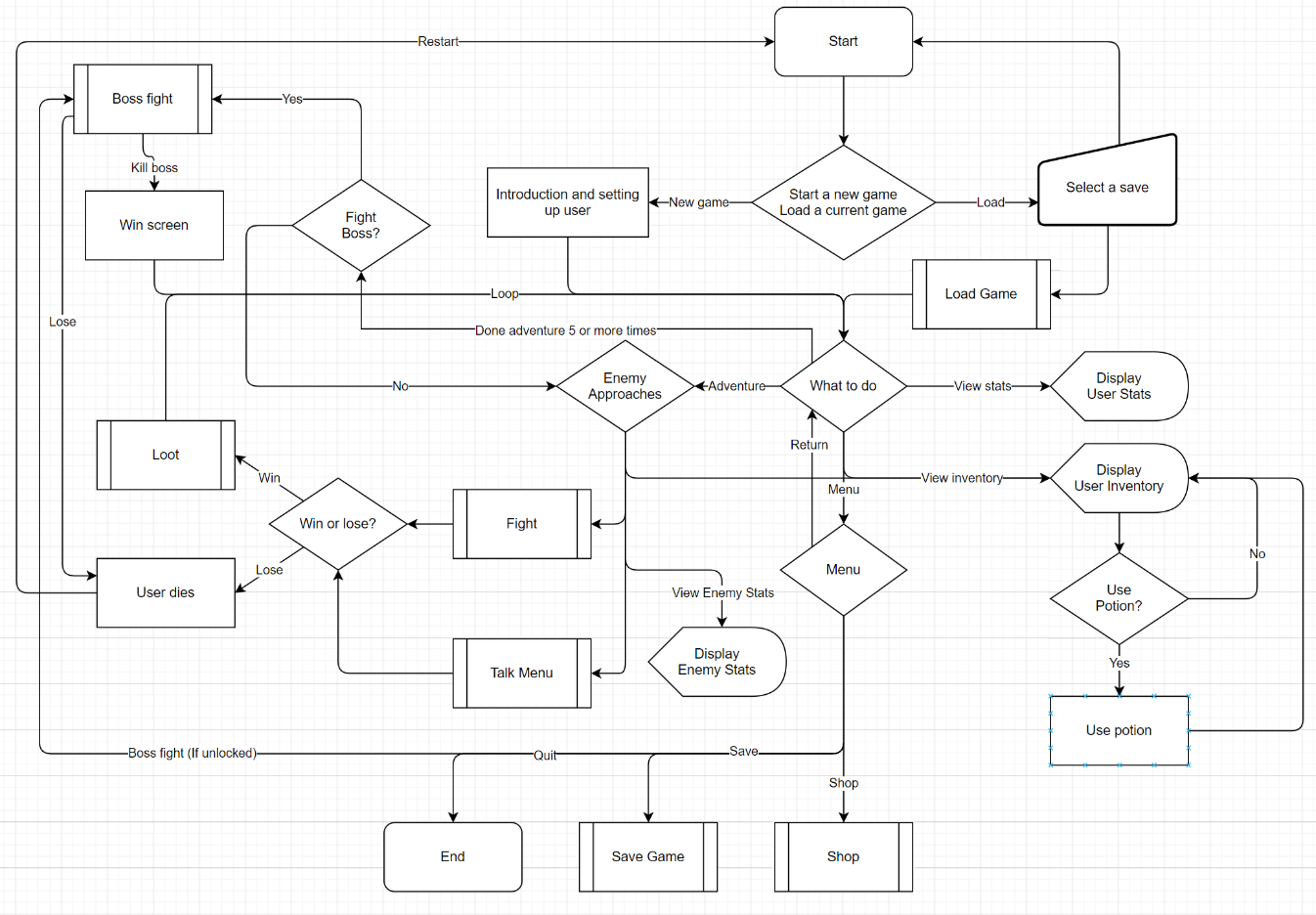
For my python project, I am hoping to do a text-based adventure game. In this game, the user will be able to fight waves of enemies before getting to fight the boss. Each enemy they fight is unique and has different strength and health. They also drop different loot which can then be used at shops to craft weapons which will increase damage. They also drop money which can be spent at shops to get potions. However, the player does not always have to fight the monsters, as they an also try talking to them to make them run away. This enables me to give the enemies more character as well as opening the game up to multiple endings. There will also be a levelling system where the user will get XP from each enemy they fight. The harder the enemy is to kill, the more XP they will get. Once the user has got 15 XP, the level up, which increase their health and damage. Also, as they level up, more enemy types appear that are stronger. The game will also have multiple endings, a normal ending, a neutral ending, and a hard ending. The pacifist ending will be where the user talks down all he enemies instead of fighting them, and the hard ending will be where the user fights the hidden boss before the main boss (as this makes the main boss harder). The user will also be able to save the game when not in battle, so that they can resume the game later. The save will be a text doc that python creates. This will also allow me to have a load menu where the user can load a certain save.

# 2). Analysis of the requirements of the program

* Fighting waves of enemies
  + To make this, I will use a function that will run every time the user moves forward. The function will also contain a random enemy generator to decide what enemy the user will fight. It will then call then enemy and its stats from a different class, so that the user can fight it.
* Enemies are unique
  + I will achieve this by having a class for enemies where I can define their variables, e.g., how much health each one has. I will also use the talk function to make it seem like each enemy has their own personality, which will be achieved by printing different lines depending on the enemy.
* Drop different loot
  + To do this, I will have another function for creating loot. The function will receive the enemy name and then reference that against a library to get stats of said enemy. It will then randomly generate the amount of each item using the random module in python.
* Shops
  + Shops will be a function that the user can call while in the program. It will have weapons shown as stock but requiring loot, while it will show the potions needing money. The shop will take in the user’s inventory, so that when a user choses an item, the program can check to see if the user can get the item. It will then also be able to check which items they already have. Once the user buys the item from the shop, it will be appended to their inventory or (in the case of potions) the amount will be changed.
* Crafting weapons
  + For crafting weapons, I will store the loot the user gets along with how much the user has, into a 2D list. The program will then check each item in the list for the material, using a for loop, then get the amount associated with it. It will then deduct the amount used from the 2D list.
* Talking to monsters
  + To allow the user to have unique interactions with the enemies, I will have a separate menu to the fight menu, for talking to the enemy. It will have different options on, and each option will be linked to one of the types of enemies. The user can then use the correct one on the enemy to make them leave or give the enemy an extra attack if the user picks the wrong one. This will be achieved by a simple menu with an if condition to check to see if it is the right one for their enemy
* Multiple Endings
  + Normal
    - This will be achieved by the user killing at least one enemy, and then killing the boss. I will track this in the program by having a Boolean which states if the user is on track for each ending. For this ending, the track will detect they have killed at least one enemy (turning the Boolean for pacifist ending to False) and if they have fought the secret boss.
  + Pacifist
    - This ending is got by not killing a single enemy. The Boolean for pacifist will be set to True when they enter the boss fight, so the program knows that is the ending they can get.
  + Hard
    - This ending is got by killing the secret boss and making the sword. I will not use a Boolean for this, instead I will just have a check to see if they have the sword or not.
* Levelling system
  + I will implement a levelling system by creating 2 variables: XP and level. The user will start with a bit of XP so that they can get their first level quick. After every fight, I will take the users XP and run it through a level up function that will loop until no levels are left. It will do this by taking away 15 XP each time and then adding a level. It needs to loop as some enemies give off more then 15 XP so it is possible to level up multiple times after 1 enemy.
* Level up increases damage and health
  + When the program levels up the user, it will also increase the damage and health by a fixed amount. This will be done by a basic function that gets the current stats and changes them. It will also by less health and damage gain once you go pass a certain level to try and slow down the user getting too powerful too quick. This will be a simple if statement that will look at the current level.
* More enemy types appear with level up
  + I will make the harder enemies appear as the user levels up. I will do this in a function that uses a simple 2D list that will store enemies and how common they are. Once the user reaches a certain level, then extra enemies’ value will be changed from 0, to however rare they are. Then, each time a battle is done, it will go through that function to get the rarity, so it will always be up to date
* Saving
  + The user will be able to save the game. The program will create a new save for each new user where the name of the save, will be the name of the user to make loading nicer. The file will then store all the user’s data along with the date it was saved. It saves the date so that the user can tell which save is there’s. Also, the program will do a try statement to try and open a file with the same name as the user, so that a save will not be overwritten. Instead, the user will be given a save that is their name, but with a number at the end. If this happens, the user is told what number they have at the end, just in case there are multiples with numbers at the end.
* Loading
  + The program will also have a load option so the user can resume their game. The program will show a list of all the available saves so that the user can pick theirs. I will be able to show all the files by using the glob module, which searches for all the files with a given field. Once the user choses a file, the program will extract all the data and assign it to the correct variables and put the user back where they saved.

# 3). Design

Flow chart to show how the main program works.



Flow chart showing how the shop works.

Diagram

Description automatically generated

Flow chart showing how saving works.

Diagram

Description automatically generated

Flow chart showing how the attack phase works.

Diagram

Description automatically generated

Flow chart showing how the talking system works.

Diagram

Description automatically generated

Flow chart showing how the looting system works.

Diagram

Description automatically generated

# 4). Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test  Number | What I am testing | Expected Outcome | Outcome | Pass or  fail? | What I need to do |
| 1 | Loading a save. | For the inventory and stats to be updated with the ones in the save. | Inventory and stats updated with the new information. | Pass. | Nothing. |
| 2 | Saving a game. | For the contents of the inventory, the stats, and other relevant information is saved. | Created a new save with all relevant information. | Pass. | Nothing. |
| 3 | Inputting something other than a number in a number field. | For the program to reject the input but still carry on. | Some number inputs crash when a non-number is entered. | Pass/Fail. | Correct some of the inputs so they can still run if non-number inputted. |
| 4 | Dying. | When I die, it should close the program and tell the user to re-open and play again | When I die, the program closes and tells me to re-open to play again | Pass. | Nothing. |
| 5 | Trying to buy an item without enough material. | Shop does not sell me the item; the shop does not take any materials and the program tells me I need more of said item | Shop does not sell, instead, gives me a message saying I need more material. | Pass. | Nothing. |
| 6 | Trying to buy a potion without enough money. | Shop does not sell the potion, nor take any items, and gives a message stating I need more money. | Shop does not sell the potion and it does not take any money, but it does not print a message saying I need more money. | Pass/Fail. | Make it so that it tells the user they need more money. |
| 7 | Trying to access the boss fight early. | In the menu, once you have unlocked the boss fight, a hidden option appears to access boss fight. I am testing if I can use that button even if it is not displayed. | Menu does not take me to boss fight, instead returns me to the same menu. | Pass. | Nothing. |
| 8 | Boss fight access in menu. | Testing whether the boss fight option appears after 5 rooms. | After 5 rooms, the boss option appears in the menu. | Pass. | Nothing. |
| 9 | Saving a file with the same name as another file. | The file name will have a number added after it to make it unique. | Saves the file with a number at the end and tells the user the new file name. | Pass. | Nothing. |
| 10 | Saying the wrong thing in the text menu. | The enemy will attack you instead of leaving. | The enemy attacked and the fight continued until one of us was dead. | Pass. | Nothing. |
| 11 | Trying to get each ending. | Each ending gives their unique message and puts the user back in the menu. | All 3 endings gave their messages and gave the boss loot but got sent into the attack menu after the pacifist ending. | Pass/Fail. | Make the boss fight go back to the right menu after pacifist |
| 12 | Defeating an enemy. | When an enemy’s health hits 0, it should say they have died and give the user the loot. | When the enemy dies, the program tells me and then gives me their loot. | Pass. | Nothing. |
| 13 | Checking enemy rarity. | Enemies that are only meant to appear at higher levels will not appear at lower levels. | All enemies appear at the level barrier they are meant to. | Pass. | Nothing. |

After this testing, I was able to quickly fix the problems in test 6 and 11.

# 5). Critique

What worked

* The saving and loading systems and the loading menu
* The multiple endings
* The combat systems
* The looting system
* Putting most of the code in functions

What did not work

* Inputs when they get an invalid input from a user

What could have been improved

* Make it so that the shop tells you how much more money or how many more items the user needs to make the purchase
* Make it so the program restarts after death instead of telling the user to restart themselves.
* Add more items to the shop to spend money on.
* Add more enemies to make it more interesting
* Develop combat so there can be different types of attacks that will do more damage to certain enemies.
* Improve the inputs so all of them can take values they are not meant to.
* I could have made the visuals a bit nicer as some menus are more indented than others.
* I could have used pygame to implement imagery into it to make it more visually pleasing
* The save file naming as if people did not read which file name they where given when saving, they might get the wrong save