Analysis of my project

Problem identification

For my project I will be making a turn-based RPG in python. It will have world traversal and a separate battle system, with dialogue being presented in the python shell as plain text and the user will input data using the keyboard. It will have a save system that will save the state of all of the variables into a separate database that will be called to when the program is next run, so all the positions of objects, state of characters etc are saved.

Why should it be computerised?

This is a problem that should be computerised as I believe it will offer a more engaging experience than if I were to make a card game, for example. It is easier to make a more engaging story and will entertain the user with a battle system to get them fully invested and always thinking ahead of their next move in order to obtain the best outcome possible. It will also be easier if it is computerised as I will need to make algorithms to calculate the amount of damage taken from each attack, which will be much quicker when it is computerised. My main features will be the battle system, which can be broken down into the individual commands for the party characters and enemies, and traversal of the world, which can be broken down into movement and scripted events. This will mean I am able to implement abstraction to break down the problem into smaller, more manageable parts. Some of these parts, such as the movement or the commands, can be separated into different procedures to make my code easier to debug as I can find exactly where the program goes wrong in my code.

Users can easily get frustrated with the overall system, causing them to potentially quit the game if the UI is difficult to navigate. My project will include a UI that is easy to navigate and is very self-explanatory, lessening any frustration that could occur from the user. Users will also need to be able to restart from the exact point they last saved after the code is rerun, so they don't need to slog through the same areas to get back to where they were. The state of variables will be saved to a database to be retrieved when the program is re-run and you will have an option to delete this data to start the game again from the beginning. The game will run on 1 window, switching between traversal and battles when necessary. There will be no need for a budget to make this, as all of the resources that I require are already available to me.

Stakeholders

My main stakeholders are Sam Davis and Alex Morrison, as they are very interested in games, are a similar age to me and my target audience so what they and I would like for the game should be similar to that of the target audience, and they know what goes into an engaging RPG. The primary way they will make use of my solution is entertainment, and a good engaging story to get invested into. They are looking for a new RPG experience to get invested into, with some replayability in order for them to get the most out of this game as possible. They also want the game to challenge them mentally requiring them to come up with a good strategy to overcome the harder challenges. They require a game that has 2 main parts: traversal and battle. The battle system will be turn-based, suiting their needs, with a party of characters either being controlled by the player, or given a role by the player, allowing the character to control themselves with their own AI. They want the traversal to be top-down with a fixed camera. My game will provide them with sufficient challenge to make them always think about their next move, but not so hard that they need to grind levels for a long time to be able to win. The battle system will have a party of 4 characters, each coming into the team at different parts of the story, and the player will have the option of controlling each of them themselves or allowing the AI to take control of the characters that aren't the main character. The story will also invest them with sidequests to flesh out the world to make it feel more alive, and that there are other things going on in the world than just what happens in the main story.

Research

My research methods are a questionnaire for my stakeholder and researching online into 3 existing turn-based RPGs: Final Fantasy VII, Pokemon Platinum and Persona 5. I am also interviewing my main stakeholder to see what they want in some of the more detailed features of the game.

I believe these are the best methods of research for making a game, as since there are already many turn-based RPGs on the market, I can look in-depth into some of them and try to extrapolate the best features out of them and try to improve upon them to make an overall better experience. For example, in Final Fantasy VII, traversal can be cumbersome and awkward, so I want to make it easier to see where it is you can go to make a more enjoyable experience for the player. A questionnaire is also a good research technique as it can give me a good idea for the sort of game that my stakeholder and my audience want, so I can tailor the game to their desires. These methods are better than something like observation as I can ask what the stakeholders really want for the game, instead of watching someone play a game and decide myself what I think should be implemented, as what I think should go into it may be different to what my stakeholders want. An interview is much better as I can see what the stakeholders want in the more detailed aspects like how random encounters are handled, or specifics in the battle system to make it more entertaining for them.

Research into existing products

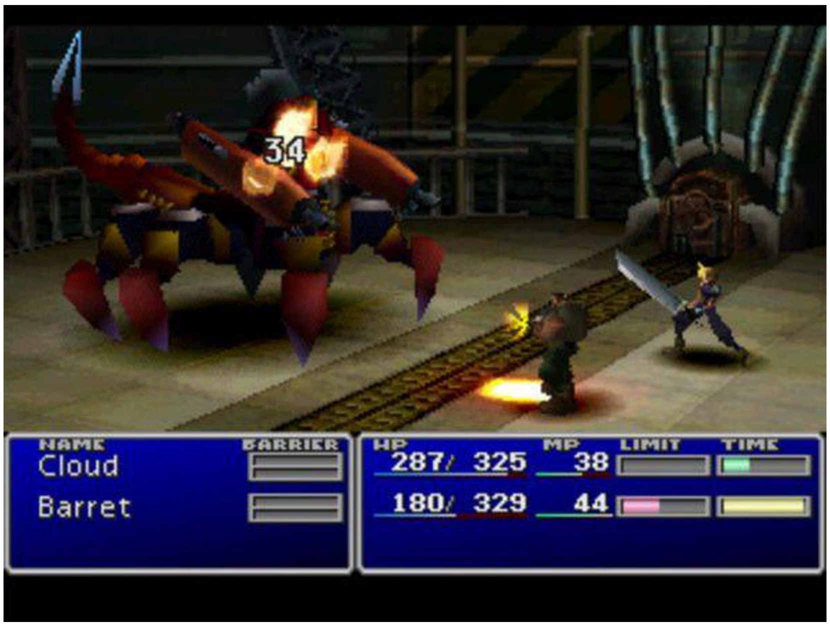
Product 1- Final Fantasy VII

Developer- SquareSoft

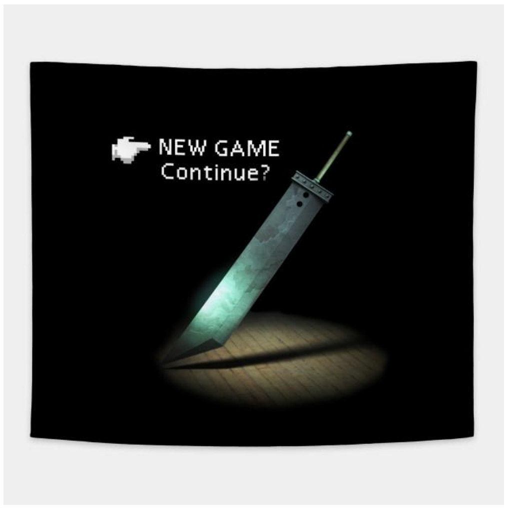
Publisher- JP: SquareSoft, WW: Sony Computer Entertainment

Platforms- Playstation, Windows, iOS, Playstation 4, Android, Nintendo Switch, Xbox One

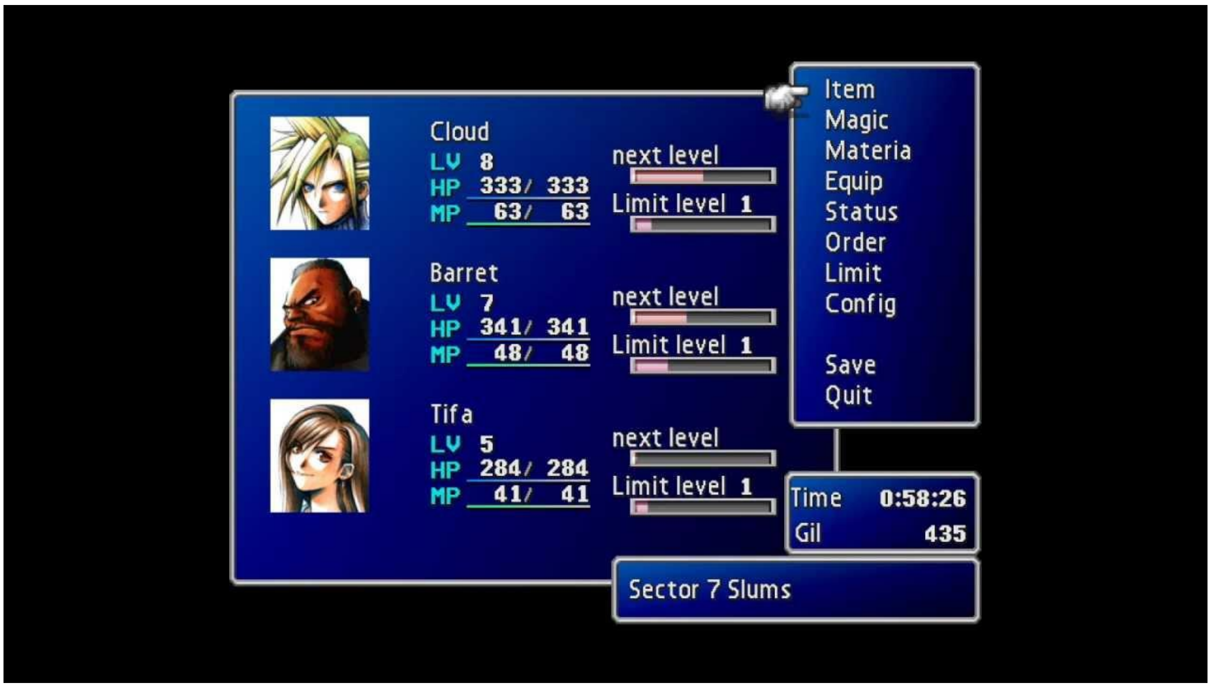
Initial release date- 31 January 1997



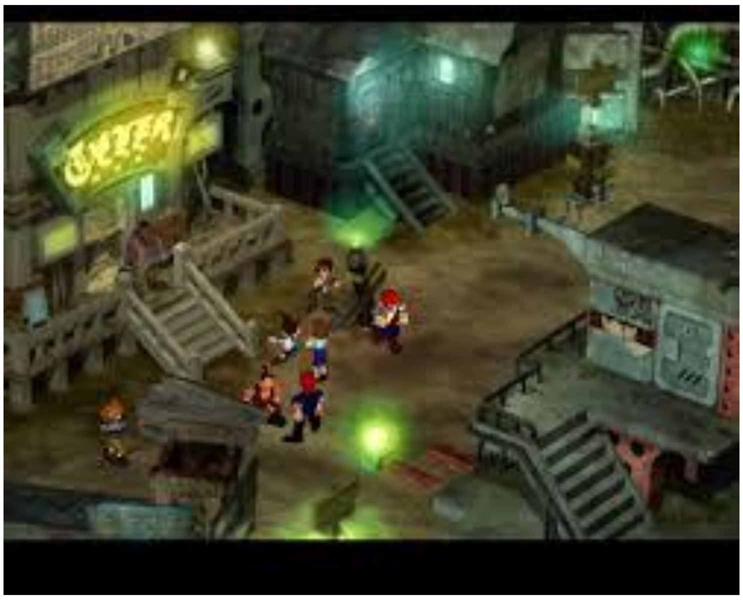
This picture shows part of the battle system in Final Fantasy VII. There is a bar that shows how much time each character has to wait to be able to attack again and, if the bar is full for more than one character, the player can choose which character will be the first to attack. The MP gauge is used up each time a magic attack is used, which can be equipped onto the characters' weapons. If you don't have enough MP for the next magic attack, the game will not allow the character to attack with magic. The 'Limit' gauge fills up a certain amount when the character attacks or is attacked. The amount it fills by depends on how much damage is taken. When the gauge is full, the character's normal attack is replaced by a 'limit break', which is a much more powerful attack. You are able to run away from these battles, but it is advised that you defeat all the enemies, as doing so will gain you experience points so that you can level up and grow stronger. This battle system is a lot more engaging than many turn-based RPGs that allow you to take as much time as you want before each attack, since the enemies attack on a timer, you are forced to think on your feet and to stay engaged at all times in the battle. Otherwise the enemies will attack you before you are able to attack them or use any items. Certain enemies are weaker to certain magic attacks, so it is encouraged to try different attacks on different enemies to see what works. It is good to have certain enemies weak to certain types of attacks, to reward the player for experimenting with attacks in battle. I will make enemies that are weak to some attacks and that resist others, but I will also make enemies that have no weaknesses to offer a higher level of challenge to the player.



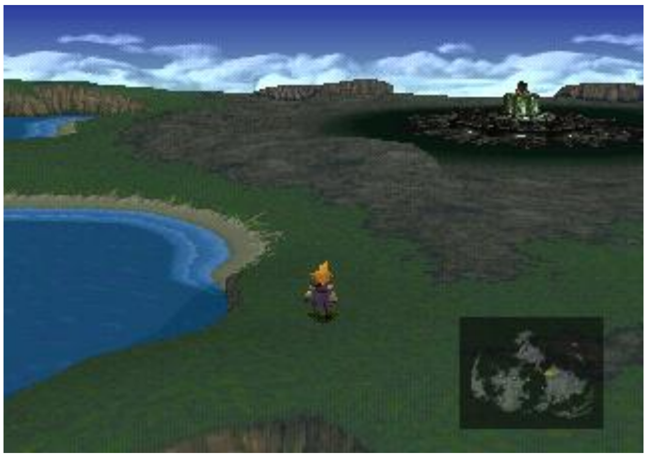
This picture is of the game's main menu. It is very simple, only offering 2 options: New game, which allows the player to play the game for the very start, or continue, which allows the player to load a save file from the PS1 memory card. My main menu will also be very simple, as there won't be any options for the player to change for the game, like difficulty.



This picture is of the pause menu. The 'item' option allows you to use items such as potions the replenish the health of your party. The 'magic' option allows you to choose which type of magic attacks each character in your party can use. The 'materia' option allows you to equip stones (called materia) to each characters' weapons to allow them to use magic attacks such as ice and fire. The 'equip' option allows you to equip weapons and armour for each character. The 'order' option allows you to change the order of your party and, as the maximum party size is 3, if there are more than 3 characters with you, you can switch characters in and out of your party. This pause menu also allows you to save your game onto a save file, and you can quit to the main menu. Gil is the currency in Final Fantasy VII, which you use to buy weapons, armour, items and materia. This menu, I believe, is quite useful as each option is relatively self-explanatory, making it very easy to quickly do what you need to and easily get back to the game.



This picture shows how the player traverses the main towns. This picture shows one of the slums in the main city of Midgar. The world mainly consists of static backgrounds with the character models over the top. This method of traversal can be quite confusing in more cramped areas of the game as it can be hard to tell which parts are interactable and which are just walls. Random encounters with enemies will occur in most of these sections with enemies that will allow you to gain experience. I won't be using this method of traversal in my game as I think it is too awkward to find your way, and I would need to draw every background as a static image which, with the time constraints, I will not be able to do.



This picture shows the other method of traversal in Final Fantasy VII when you aren't in any major towns and you are just traversing the world map. In this section, the player has control over the camera and can move all throughout the world. The map on the bottom right of the picture shows the character's current position and any towns they are able to visit. Random encounters can occur in this area as well with different enemies to those in the major cities. This a much easier method of traversal than in the towns as you can easily see the places that you are allowed to go to, and you have full control over the camera, allowing you to easily go in all directions.

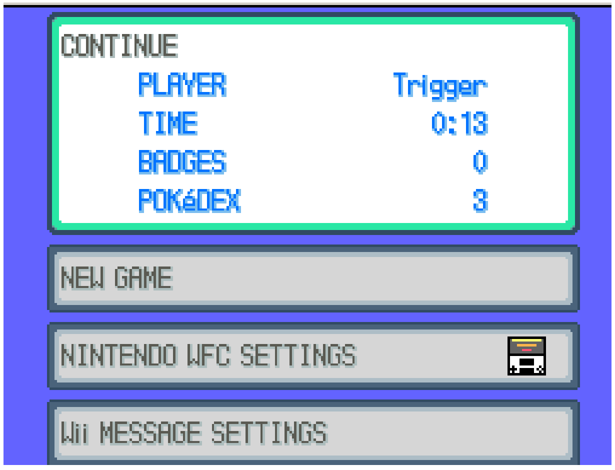
Product 2- Pokemon Platinum

Developer- Game Freak

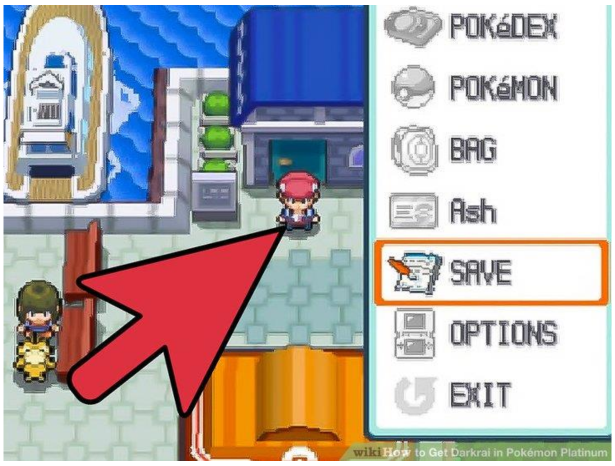
Publisher- Pokemon Company, Nintendo

Platforms- Nintendo DS

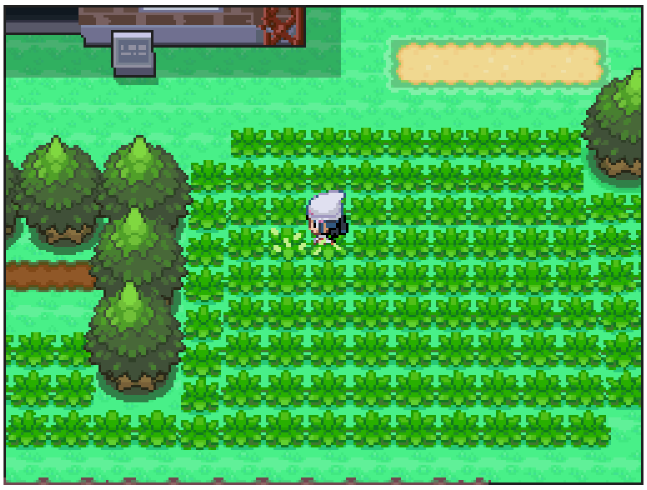
Initial release date- 13 September 2008


CONTINUE
PLAYER Trigger
TIME 0:13
BADGES 0
POKéDEX 3
NEW GAME
NINTENDO WFC SETTINGS
Wii MESSAGE SETTINGS

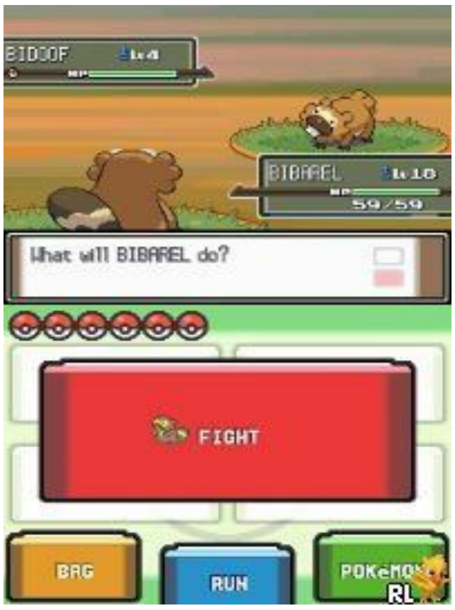
This picture shows the main menu of Pokemon Platinum, with options to continue playing the current save file from the last point you saved, a new game option to start the game from the beginning, and WiFi settings so that you could do things online such as receive 'Mystery Gifts' and battle/trade with others online. This menu is very concise and does exactly what it needs to, allowing the player to jump quickly into the game.



This picture shows the start menu that is activated in the main world when pressing the X button. This allows you to do things like checking the pokemon in your current party and switch who will come out first in the next battle in the 'Pokemon' option. You can also go into your 'bag' and use items that have different effects on the world or your pokemon like potions to heal your pokemon and repels to lure away wild pokemon. You can also view your 'Pokedex' to see how many species of pokemon that you have seen or caught. The 'save' option allows you to save your current progress to your save file. You can view your trainer card with the amount of money you currently have and your gym badges in the option labelled with your character's name (in this case: Ash). The 'Options' menu allows you to change the game settings such as text speed. This menu is useful in the way that it is obvious which option leads to which menu, but it can be quite clunky and can take a while to do what you need to at times. For example, if all of your pokemon need healing, you cannot use a potion on each of them at once, you need to heal one, then select the potion again to heal the next and so on.



This picture shows a portion of the world traversal. At the moment the character is in tall grass, meaning that on each step of a new tile, there is a chance that a wild pokemon will attack and the player will be forced into a battle. The chance of encountering a pokemon depends on the area and time of day you are currently in. The entire game world is split up into separate tiles and one step from the player moves them one tile in the direction that is pressed. Sometimes the path is blocked by an obstacle such as a rock or a tree that will require an HM move to be taught to a pokemon in your party so they can remove it for you to proceed. This method of traversal is much easier than Final Fantasy VII as you can always see where you can go and where you can't as it is blocked by a wall or an immovable object.



This picture shows the battle system for Pokemon Platinum, which utilizes both screens of the DS effectively. The top screen shows the health and level of the player's and opponent's pokemon, allowing the player to devise a strategy depending on the current state of each pokemon. The bottom screen shows all of the options the player can choose during the battle. The 'fight' option will bring you to a menu of a maximum of 4 moves that the pokemon can use to either attack the other pokemon or heal/buff themselves. The 'run' option can only be used in wild pokemon battles and will allow you to escape the battle but will give you no experience points. This cannot be used in trainer battles and will bring up a message saying that you cannot escape a trainer battle. The 'bag' option allows you to go into your bag and use any items that you have available, for example, to heal one of your pokemon. Pokeballs can be used to catch wild pokemon but cannot be used to steal pokemon from other trainers. The 'Pokemon' option allows the player to view the state of all pokemon in their party, all of their moves and a general summary of the pokemon. It allows the player to switch out the current pokemon they have in battle for another one in their party, for example, to gain the type advantage over the opponent. The touch screen is used very well in this game, as it is much quicker than in other pokemon games to select the options you want.

Each pokemon and move that a pokemon can learn has a certain type (a pokemon can have a maximum of 2 types) and, most of the time, is 'super effective' against other types, meaning it will do 2x damage against that type. If a move is super effective against both of a pokemon's types (e.g. a fire type move being used against a pokemon that is bug and steel type) it will do 4x damage. This, in my opinion, is a very good battle system as it rewards the player for learning the type combinations and for having a well-balanced team with many different types. Unlike in Final Fantasy VII, you have as much time as you want to decide what you want to do in the next turn, meaning you can think of a good strategy without any time pressure so you don't always need to be thinking on your feet. However, if you are not over-levelled, the game can still be very challenging in places, causing you to need to think very carefully about what you need to do next in order to win the battle.

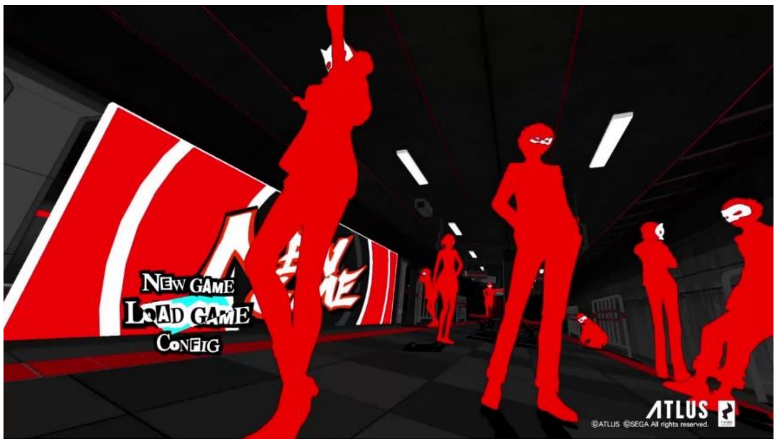
Product 3- Persona 5

Developer- P-Studio

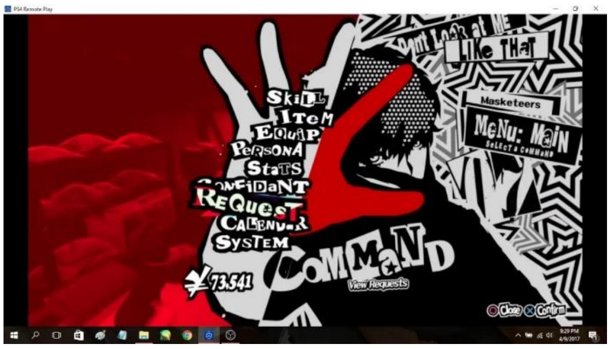
Publishers- JP: Atlus, NA: Atlus USA, PAL: Deep Silver

Platform- Playstation 4, Playstation 3

Initial release date- 15 September 2016

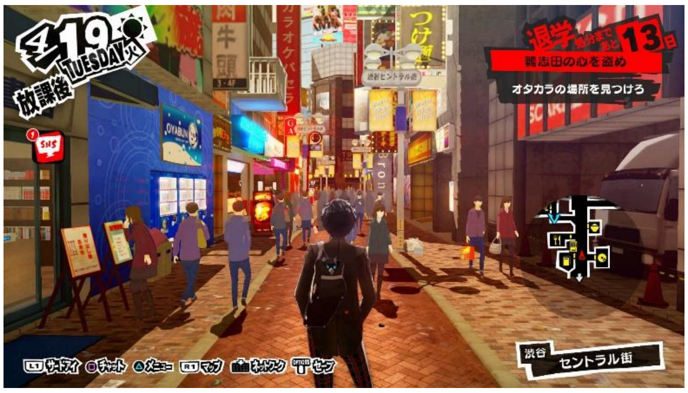


This picture shows the main menu for Persona 5. If you choose 'New Game', the game will begin from the start but all save files you already own will stay. If you choose 'Load Game', you can choose one of a maximum of 16 save files, and start the game from where you last saved to that file. The 'Config' option allows you to change the game's settings, such as the difficulty of the game and the language.

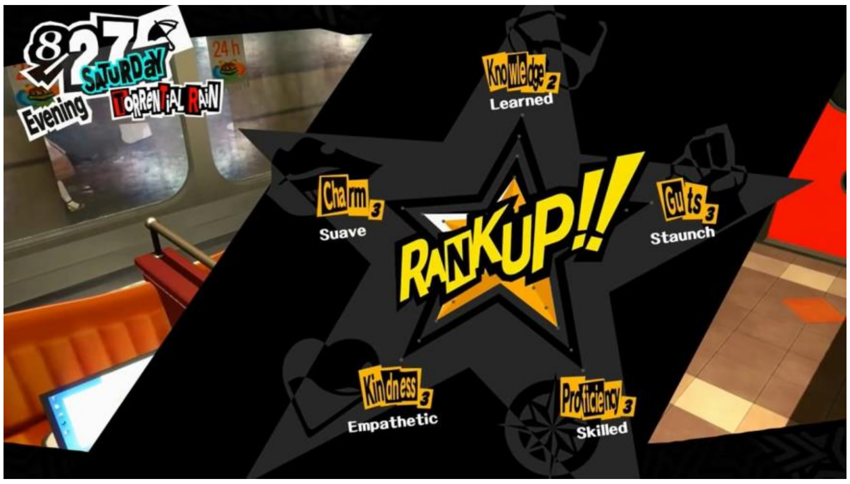


This picture shows the pause menu in Persona 5. The 'Skill' option is only available inside the Metaverse (see later picture) and allows you to use healing spells with a Persona from one of your party members if they have a healing spell. The spell will cost a certain amount of SP, depending on how powerful the spell is. You can also use items to heal your party's HP or SP. You can also use items in battle to buff your party or de-buff your enemies. The main character can hold multiple personas at once, and the 'Persona' option allows you to view these personas and choose which one will be equipped at the start of the next battle. The 'Stats' option allows you to view the levels and general stats of your party members, along with their personas. The 'Confidant' option allows you to view the 'contracts' you have made with characters throughout the story and see how far along their storyline you are. (see later picture). You can view the requests you have received from the general public to reform people by going into Mementos and defeating their 'shadow'. The game takes place over the course of around a year and you can view the calandar to plan your days ahead of time, as most things in the game have a deadline.

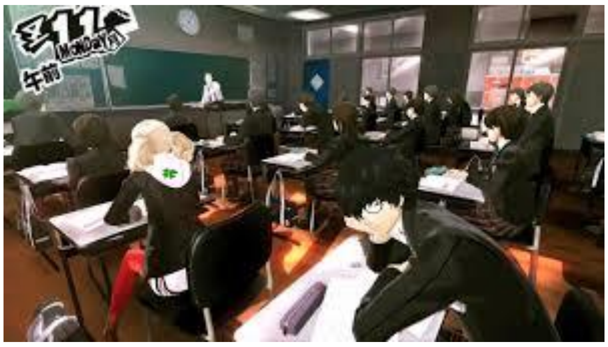
This menu is very stylised but it is easy to navigate so it works very well.



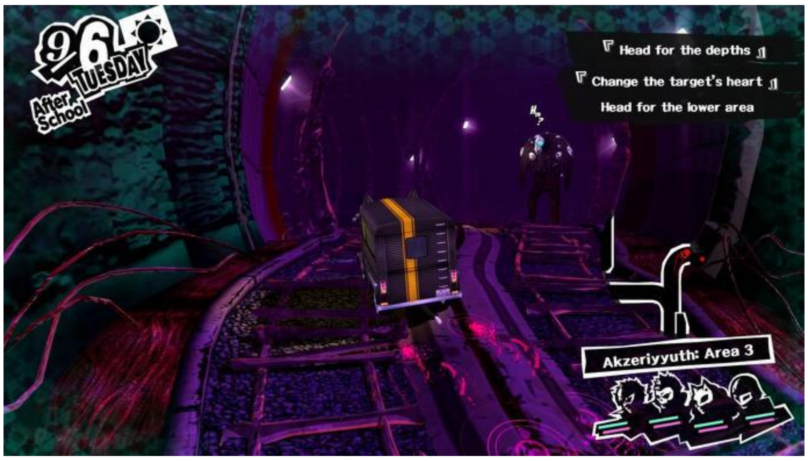
This picture shows one of the main cities in Persona 5: Shibuya. In these cities, you can sometimes meet with your confidants to progress through their storyline. If you choose to meet with a confidant, time will pass, and it will be the evening after you have met with them. You usually only have 2 timeslots per day if no story cutscenes happen in that day. This is why you need to plan your day ahead, as you cannot go back and redo any days. In these cities, there are also many shops where you can spend money on items for the Metaverse or buy things to increase your social stats.



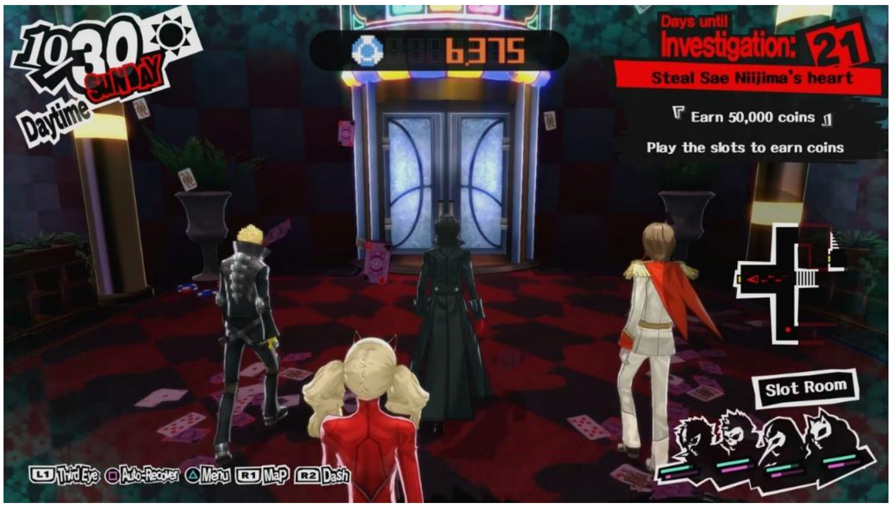
This picture shows the social stats that you can upgrade during your playthrough. At some point, during most of your confidant storylines, you need to have a certain social stat at a high enough level in order to trigger something to progress with your confidant. For example, to progress past level 6 confidant ranking with Makoto Niijima, you need charm level 5. This mechanic forces the player to balance ranking up confidants and ranking up social stats, to get the most out of the playthrough. You need a certain amount of points to rank up one of your stats. Each stat needs a different amount of points to rank up, knowledge needing the most as it is the easiest to get points for.



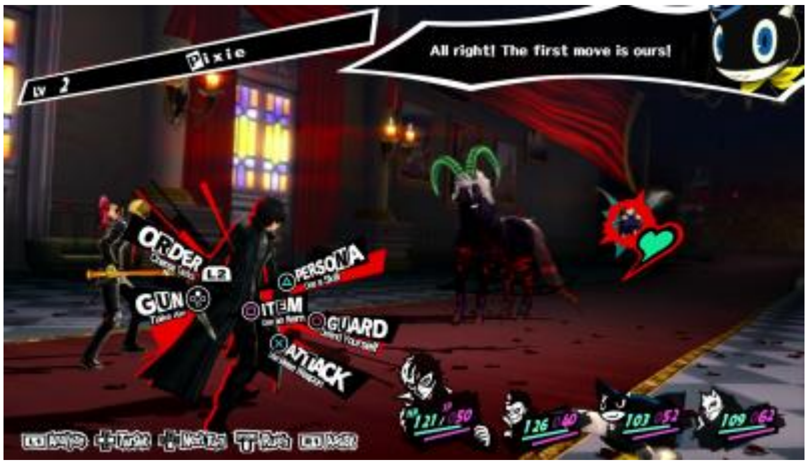
This picture shows the school segments in Persona 5. These segments happen most days (other than Sundays and holidays) and, some of the time, you get asked multiple choice questions by your teacher. If you get this question right, you get a point in your knowledge stat. Most of the time, though, you get text messages from your friends about the current situation in the main story during these segments.



This picture shows the group traversing through Mementos, dubbed as 'the public's palace' throughout the game. This area has many floors, all of which are procedurally generated and the enemies, named shadows, get harder the lower down through Mementos you go. This is a place that is often used for levelling up your party, and you go here to cause a change of heart in the general public if you get a request from someone on your 'Phan-site', run by a character called Mishima. You can go to this place any time, after a certain point in the story, but once you leave, it will be the evening and you won't be able to do anything in that time slot, as you will be too tired.

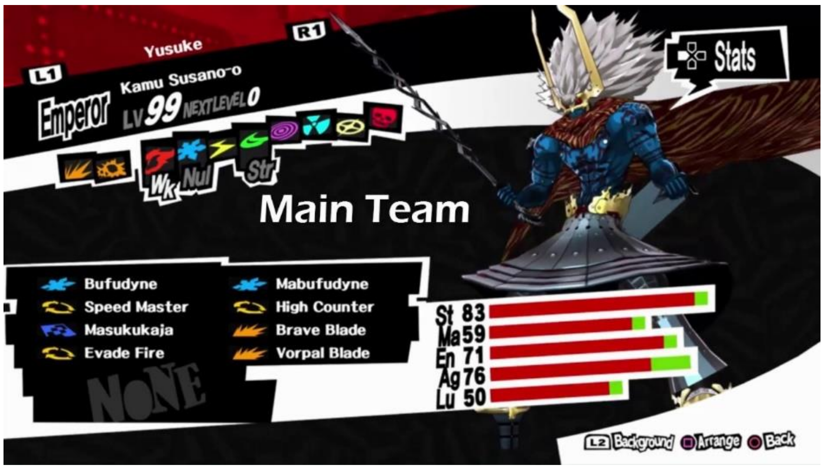

10/30 Sunday
Daytime
6375
Days until Investigation: 21
Steal Sae Niijima's heart
Earn 50,000 coins
Play the slots to earn coins
Slot Room
Third Eye
Auto-Recover
Menu
Map
Dash

This picture shows one of the main dungeons in the game, dubbed ‘Palaces’. These palaces, unlike Mementos, are not procedurally generated. The shadows will always be around the same places, but the enemies that will show up in the battle are random to a certain degree. Like when you go to Mementos, once you leave the palace, it will be the evening and you will be unable to do anything in that timeslot. Each palace has a certain deadline to finish it by and, if you don’t finish it by the deadline, you will get a game over and have to restart a week back from the deadline. Each palace has a different theme, the one in the picture is a casino and all of the palaces are based on the ruler’s cognition of how they think of the place in the real world. For example, the first palace is based off one of the teacher’s cognition of the school, and he views it as a castle and views himself as the king of the castle.

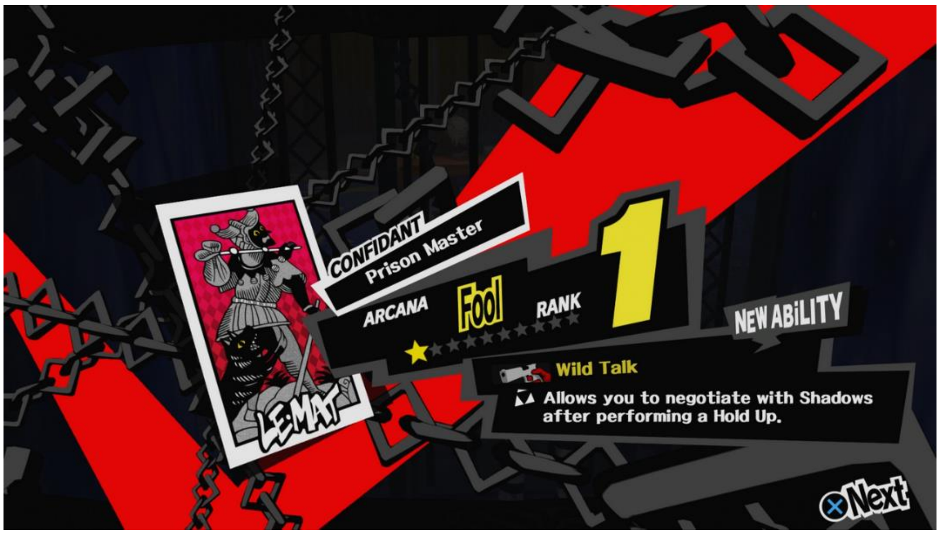


This picture shows one of the battles you encounter in the game. If you sneak up on the enemy and start the battle without them seeing you, you will get an ‘ambush’ and all of you party will be able to attack before any of the opponents. The picture shows all of the different attacks each person in your party can use, and the button that corresponds to each attack. When you use your persona skills, the enemy may be weak to that type of attack, and they will be knocked down. If you knock down an enemy, you will get a ‘1 more’ meaning you can attack again with the same character. If all of the opponents are knocked down in one turn, you can hold them up to either negotiate with them, or perform an ‘all out attack’ which does large amounts of damage to all of the enemies. If you negotiate with them, you can ask for money or items, or you can have a conversation with one of them to try and convince them to join you as a persona. This battle system is the most fluid of the

3 games I have researched as you can instantly use one of the attacks at your disposal instead of having to scroll through a menu to find the command you want to use.



This picture shows the stats and attacks of one of the personas. In this case, the types of attack at the top show that this persona is weak to fire, resistant to wind and immune to ice. Other effects can be 'drain', meaning the attack will heal them instead of attack them, and 'repel' reflects the attack back at the user. Each persona can have a maximum of 8 attacks and gain more as they level up. If it tries to learn an attack after already having 8, it will ask you to delete one or not learn this attack. Every persona will learn different attacks, have different stats and will receive different effects when attacked with certain skills.



This picture shows one of the confidant progression lines, each represented by one of the tarot cards. As each confidant levels up, if you fuse a persona in the Velvet Room, the resulting persona will gain extra experience points depending on the level of the corresponding confidant, as each persona is also represented by a tarot card. Most of the time when a confidant levels up, you will also get some sort of upgrade that will help you when you infiltrate palaces. For example, as you level up your confidant with Mishima, your party will gain more experience in battle, allowing them to level up quicker. Levelling up your confidants is crucial to having a chance in battle in later areas in the game, so you need to plan who you need to spend time with to get the most desired effects.

Essential features

I feel that one of the most essential features of my game will be the save system. I want the player to be able to save at any time outside of battle and dialogue, like in Pokemon Platinum and Persona 5 when out of the metaverse. This is an issue in Final Fantasy VII as there are designated save points and you can only save the game at those points, and they can be quite spread out, which can lead to a lot of frustration, especially if the game crashes.

Another essential feature for my game and most RPGs is sidequests. These often give more depth to the overall story and characters in the game, giving the player an overall better experience. For example, in Persona 5, you can spend time with many different characters, each with their own story, enriching the experience for the player. Without the sidequests, the game will be a very bare-bones experience and it will be much harder to flesh out the world.

The most essential features are story and the battle system. An engaging story is crucial to keeping the player interested in the world to carry on playing. The battle system is also crucial as that is what you will be doing for most of the game, so you need to be able to enjoy it to be able to carry on playing. It will need to have a party system, with the player being able to choose whether or not they control the rest of the party along with their own character, or if they will pick commands automatically. This adds extra depth to the game as the player will need to think about more than just their own character, so they need to think hard about their strategy in harder fights to keep all of their party members healthy. Different enemies will need to have different weaknesses and resistances, as well as the different characters in your party. The player and party characters will need to gain EXP from winning battles, or choose to run away but will not gain any EXP. Also, if they win the battle, there is a chance for them to get money and items as a reward as well as EXP. The EXP will cause them to level up over time, and they will gain more EXP from boss fights than anything else. Pokemon Platinum has a relatively basic story when compared to other RPGs, as it is mainly geared towards children, but Persona 5 and Final Fantasy VII have a constantly engaging story and, in Persona 5, there are plot twists around every corner to keep the player guessing, and wanting to carry on to see how the story progresses. Without a good story, the player may get bored with the game, as the story is normally one of the main appeals of an RPG, and they may stop playing without finishing it.

Questionnaire

1. Should there be separate difficulties to make the game more accessible to new players?
2. Should there be references to other media? If so, what sort of references should they be?
3. Should there be optional content such as bosses to reward extra exploration?
4. Should the battle system be based on a party of characters, or just one with multiple abilities?
5. What is more important, the battle system or ease of exploration?
6. Should there be random encounters or enemies that you can try and avoid or attack?
7. Should the game be sprite-based or hand-drawn?
8. Should there be multiple endings with a different final boss, or just one ending?

9. What could there be to make the game more replayable?
10. Should there be a main 'hub world' to go back to?

Stakeholder answers (Sam Davis)

1. No
2. Yes: Hidden references
3. Yes
4. Party
5. Battle system
6. Enemies that you can choose to avoid or attack
7. Sprite-based
8. Multiple endings
9. NG+ with new bosses/enemies
10. There should be a hub world

Stakeholder answers (Alex Morrison)

1. No
2. Hidden references
3. Yes
4. Party
5. Exploration
6. Random encounters
7. Sprite-based
8. One ending
9. NG+ with more sidequests
10. There should be a hub world

Interview

Q: What sort of save system do you require?

A: You should be able to save through the menu at any time in the game. You should be able to restart the game from the exact position that you saved at before.

Q: How should the battle system work?

A: The fastest character should go first but can choose whether or not the rest of your party is controlled by the player, or if it will be automatic. Each type of character should have different weaknesses and resistances, and if a character uses an attack of the same type as them, they get an attack bonus (like STAB in Pokemon). Characters will be able to learn new moves by level up, or by finding items in the world, if that move is compatible with the character. There should be items to increase basic stats like defence or attack or can find items that give you a benefit in battle, but with a drawback as well, like increasing speed but decreasing defence. Some moves should also inflict status effects, such as stun making them unable to attack for the next turn. These can be moves or items.

Q: What optional content should I add to the game?

A: Extra items to find off the beaten path, and if you find enough, you can open up a new area, should be more difficult than main game areas.

Summary of research

My project is a turn-based RPG made in Python using Pygame, inspired by the likes of Final Fantasy VII, Pokemon Platinum and Persona 5. I researched into these 3 games as part of my research into existing products. Final Fantasy VII and Persona 5 showed a battle system with a party of characters, each with different specialist abilities. Pokemon Platinum differed with this as it showed a battle system that only utilizes 1 character at a time, making it an easier system to understand for younger consumers, as they don't need to worry about the state of more than 1 party member at a time. A Pokemon style battle system will be easier to code but I feel a party battle system will be more complex and fun to code, as well as being more entertaining to play for a more experienced player.

The 3 games that I researched also presented differing traversal systems. Final Fantasy VII shows the models traversing a static background which can be very awkward to traverse, as it can be hard to tell which surfaces are interactable and which are just walls. The traversal system is much better in Pokemon Platinum as it is tile-based, and it is much easier to see what is interactable, making it less of a pain to get where you need to go. Persona 5 differs greatly to both of these as it has free traversal, so it is not tile-based, and you have a map to show where you can go.

My questionnaire with my stakeholders, for the most part, gave similar results to each other. They both believe there should be one fixed difficulty, as is the case for the vast majority of turn-based RPGs, as you can make the game a lot easier if you put in the time to grind levels and become 'over-levelled'. They both believe there should be optional content to reward the player's curiosity and exploration. They both believe that a party battle system is the better way to go as it is more complex and deepens the battle system. They offered differing opinions in how the player will encounter enemies: Sam said the enemies should be preset and you can choose to attack or avoid them, and Alex said that random encounters are better. I have decided on using random encounters, so I don't need to make individual enemies as separate objects in the world, making it simpler to code. They both believe it should be sprite-based, which I agree with as it will look much better than if it was hand-drawn. They offer differing opinions on whether or not there should be multiple endings: Sam said there should, Alex said there shouldn't. I am going to try and make multiple endings with different final bosses for each depending on how well you have done throughout the game. They both believe there should be a New Game + (NG+) to offer more replayability to the game, with new bosses/enemies and new sidequests. Finally, they both believe there should be a main 'hub world' to go back to for some reprieve and to be able to buy items, for example.

I feel this should be computerised as it will create a more engaging experience than something like a board game or card game. I will also need to create algorithms for damage taken and given for certain enemies and levels of characters and they will be much easier to incorporate in a program as the calculations are done in an instant.

Limitations of my Product

Given the time constraints of this project, and the limitations of Pygame, I will not be able to make a game that looks as good as the games I have researched and will not be able to implement any 3D elements, but as it is going to be sprite-based, like Pokemon Platinum but fully 2D, I think I will still be able to make it aesthetically pleasing to the user. I will not be able to implement something to do when you aren't travelling on a quest, like in Persona 5 when, outside of the metaverse, you have a sort of life simulator. There will be the main story, sidequests and interactions with characters within and outside your party, but nothing on quite the scale of Persona 5 when living as a student. There also will not be time constraints within the game, you will be able to do sidequests while something is going on in the main story, if you come across them. I will not be able to implement this game on anything other than a computer as that would require a development kit and license to develop for a console, which would be very costly and, since I have no budget, I cannot buy extra tools and equipment. Also, I will only be using a keyboard to control the game, as there will be limited commands for the player to input so they won't need a controller to be able to control it effectively, and it will be simpler to code for the keyboard on my laptop. I will only be able to output dialogue on the Python shell as having to output text on the game window along with other animations will be a lot harder and will make the program less efficient. The game will only be able to run on my laptop as all of the sprites for the game will be saved on it, so the program needs to call from my laptop's files to use them, and the program needs a Python interpreter and the Pygame library downloaded in order to run.

Proposed Solution

The game should be displayed in 1 window, switching between battles and traversal when necessary. This is because displaying 2 windows is very difficult in Pygame and requires a lot of hassle, so it will be much easier to just switch between the 2 main parts of the game using the same window. The Python shell will be used for displaying dialogue, so animations can be displayed fully on the game window without covering anything up, and there is then a designated place specifically for dialogue. The player will have a party of characters for battles, but only the main character will be displayed when traversing the world to avoid cluttering the screen with too many sprites. During battles, each character will take a turn with the person or enemy with the highest speed stat going first and so on. The player will be able to control the actions of each of the party members, giving them full control over each battle so they can plan their attack more efficiently, or they can leave the AI to choose the other characters' attacks, making the player have to improvise after each turn as they won't know what each character will do. If the player wins the battle, they will be rewarded with EXP which will lead to levelling up, and they have a chance of getting items and money after each battle. They can also run away from the battle, but they will gain no EXP from it if no enemies are beaten.

The game will have random encounters with enemies and there will be 5 bosses to fight through the story, with 2 being optional to reward exploration. Once you have entered a boss fight, it becomes a scripted event, so you will not be able to run away from the battle.

The method of input will be the keyboard, with the arrow keys being used to move throughout the world, and certain keys being used for each character and attack during battles, being explained in a separate text file.

The text file will explain the controls and mechanics of the game, allowing the player to jump right into the game after reading, eliminating the need for a tutorial section.

There will be a GUI menu that the player can navigate using the arrow keys and the enter key, to choose whether they want to start a new game or continue from a previous save. Once the game is completed for one save file, that file will then have access to NG+, with harder enemies, new sidequests and a new boss, adding to the replayability of the game. The player will also be able to keep all of the equipment and money that they finished the game with before starting NG+. You can enter into NG+ by just hitting continue on the main menu with a finished save file.

The game will be sprite-based, with all of the sprites being created in Piskel online, because it is easy to make sprites and animations with, and it is free since I have no budget. I will make all of the character sprites, NPCs, enemies/bosses and backgrounds/terrain in Piskel.

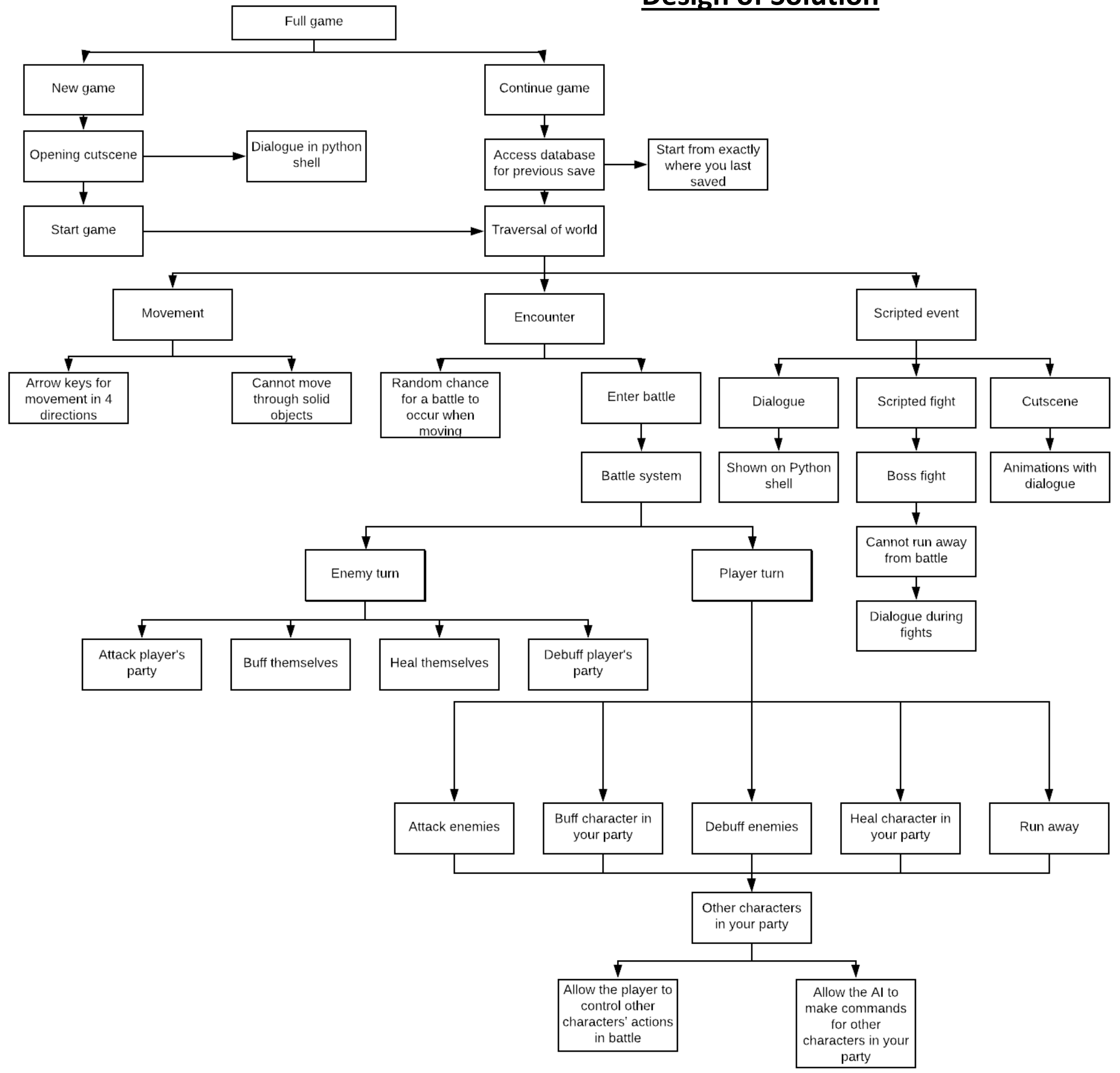
My game will only be able to work on this laptop, unless all of the sprites are copied to a separate computer that it needs to run on and can be accessed properly by the program. The computer will also need to have the Python interpreter and to have Pygame installed onto it to be able to run the game.

Success Criteria

1. Create all character and background sprites to be implemented into the game for the character to traverse the world while always being able to see where they can go. Each sprite for characters will be unique, and each area will be unique.
2. Create a battle system to be used on the same window with random encounters.
    - There will be a random number between 1 and 3 of enemies in each battle and the types of enemies will depend on the area the player is currently in.
    - Enemies will be weak to certain attacks and resistant to others, but some enemies will not be weak to any attacks.
    - There will be 4 people in your party that the player can control. The characters will be able to have their own spells to either attack enemies, buff allies or de-buff enemies.
    - There will also be consumable items that have similar effects but will not use SP.
3. There will be optional content to reward the player for exploring well, like an optional area based on Bloodborne with a boss fight and good opportunities to gain experience and items.
4. Create multiple endings based on whether or not the player gained the necessary items throughout their playthrough to encounter the true final boss.
5. Create 2 sidequests in each area for the player to get invested into, if they want a break from the main story to do new things with interesting characters, fleshing out the world more and making each area have more life to it.
6. Create a NG+ feature to add to the replayability of the game, making enemy and boss encounters harder, but the player keeps the equipment that they built up over the course of the previous playthrough. It will also have new bosses to encounter and 2 extra sidequests to incentivise the player to do another playthrough, keeping the player interested in the game for longer.

7. Create a save system that stores the state of all variables into a separate text file to be called to when the program is re-run to save the state of those variables back into the program, so the player can start exactly back from when they saved, unless they choose to start a new game, in which case, the values called will be the default values for each variable.
8. There will be 4 different areas for the player to explore throughout the story, 1 being optional, with 2 exclusive enemies for each area, and some that appear in multiple areas.
9. The game will have a balanced difficulty, so it won’t be too easy to plough through, but it won’t be so hard that it is impossible to get through without grinding levels for hours.
10. The program needs to be efficient, taking up as little space as possible, with no load times as the world will be fully interconnected to keep the player as immersed as possible.
11. My project will be finished by the 14th February 2020.

Design of Solution


Full game
New game
Continue game
Opening cutscene
Dialogue in python shell
Access database for previous save
Start from exactly where you last saved
Start game
Traversal of world
Movement
Encounter
Scripted event
Arrow keys for movement in 4 directions
Cannot move through solid objects
Random chance for a battle to occur when moving
Enter battle
Dialogue
Scripted fight
Cutscene
Battle system
Shown on Python shell
Boss fight
Animations with dialogue
Cannot run away from battle
Dialogue during fights
Enemy turn
Player turn
Attack player's party
Buff themselves
Heal themselves
Debuff player's party
Attack enemies
Buff character in your party
Debuff enemies
Heal character in your party
Run away
Other characters in your party
Allow the player to control other characters' actions in battle
Allow the AI to make commands for other characters in your party

Decompose the Problem

The full game is being split into ‘New Game’ and ‘Continue Game’. This is because the option you choose determines how the program finds the values for the variables, and these are the 2 main menu options that you get when starting up the program. If ‘New Game’ is chosen at the main menu, the variables are reset to their default value, starting the game from the very beginning. If ‘Continue Game’ is chosen and you already have a save file, the program will find the values of the variables when you last saved to that file from a separate database. It also shows that any dialogue in the opening cutscene, and any cutscenes during the game, will be shown on the python shell to make it much easier to program by just printing it to the screen, rather than trying to print it directly to the game window while you are playing.

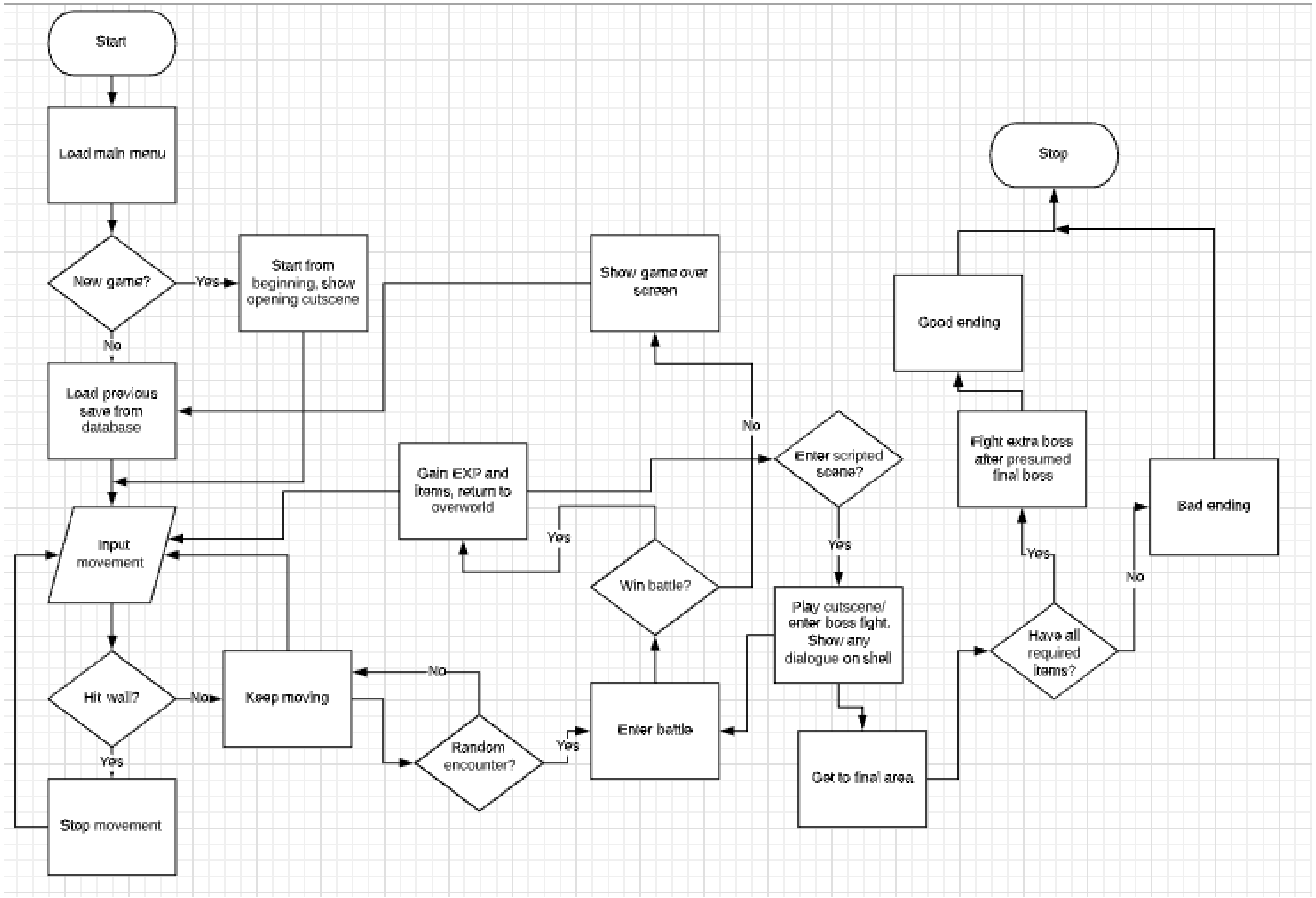
The game from here is split into 2 main sections: Traversal of world and battles. These are the 2 main things you will encounter during your playthrough. The ‘traversal’ section is split into movement, encounters and scripted events since those are what will happen while you are traversing the world; you may come across other characters for a sidequest, being a scripted event once you talk to them, and, most of the time, there is a random chance for the player to enter a battle while moving.

The scripted events are split into dialogue, scripted fights(ie boss fights) and cutscenes. The dialogue and cutscenes go together as the cutscenes will have dialogue shown on the python shell along with sprite animations. The boss fights are required for the player to proceed in the story, so you are unable to run away from the fight. There will also be some dialogue during boss fights along with a phase change when the boss has lost a certain amount of health. The scripted events are split this way because they are the only sorts of scripted events I plan on incorporating into my game.

The battle system will be split into the enemy turn and the player turn. The enemies will have a few unique attacks to attack the player’s party with, and they will have spells to heal themselves, buff themselves or debuff a character in the player’s party. The player turn will include all of the commands that the enemies can do, but the player will be able to choose what spells each character can have with the use of items, similar to Final Fantasy VII, they will be able to run away from the battle and, outside of battle, the player will be able to choose whether or not to control the other party members during battles. The battle system is split up this way because it means I can program all of the different commands one at a time, and test them against each other.

Enumeration is used throughout my game to calculate things such as damage given by attacks and health lost by attacks, along with the stats of different characters. Visualisation is used throughoit the whole game with the game window showing the game world using sprites that will be made from scratch.

The HIPO chart shows that I am able to implement procedural abstraction to make my game, as each main section is split into different parts to make it easier to manage, such as the battle system being split into enemy and player turns, and then those sections being split into the individual commands that can be run by each character.


Start
Load main menu
New game?
Yes
Start from beginning, show opening cutscene
No
Load previous save from database
Show game over screen
Input movement
Gain EXP and items, return to overworld
Yes
Win battle?
No
Enter scripted scene?
Yes
Play cutscene/ enter boss fight. Show any dialogue on shell
Hit wall?
No
Keep moving
No
Random encounter?
Yes
Enter battle
Yes
Stop movement
Get to final area
Have all required items?
Yes
Fight extra boss after presumed final boss
Good ending
No
Bad ending
Stop

Describe the Solution

There will only be 4 areas in the game, meaning there won't be too many different sprites for the overworld that I will need to make, and there will be 2 exclusive enemies to each area, and 2 enemies that will show up over more than 1 area, so I will only need to make sprites for 10 enemies. As there aren't too many unique sprites to create compared to bigger projects, I believe my target is achieveable to make each and every sprite for the game.

There will be 2 sidequests in each area to flesh out the world with small storylines for the characters you encounter. These may include teaming up with side characters or fetch quests for them, giving some insight into the inhabitants of the world. This will not require too much alteration to how the game is played, just some added dialogue for the stories, meaning I will be able to fit 8 sidequests throughout the game.

To reward exploration, there will be 3 items to acquire in order for the player to achieve the true ending, and there will be a full optional area based on Bloodborne, with a boss fight with Ludwig the Accursed/Holy Blade, with a phase change halfway through the fight. There will also be a fight with the Orphan of Kos at the end of the Bloodborne area, scaled with endgame damage to add more challenge to the player. The Orphan of Kos will also have a phase change halfway through the fight, similar to Ludwig. This area will be harder than the others, but will allow the player to gain useful items and gain lots of EXP to level up. This will be achieveable for me as it will be easy to place the items in each place and I won't need to think of a full new set of sprites, as I can base them on the base material from Bloodborne.

NG+ will not be too hard to implement as I will be able to make it a boolean variable and scale damage and level of enemies based on whether or not the player is in NG+ or not. There will also be a new boss to encounter in an area that you be able to get to in your first playthrough, but there will be no boss there if you aren't in NG+.

I will be able to balance the game well by testing the amount of damage each enemy and party character to check it will not be too hard or too easy for the player to make it through, so it will require some decent strategy to overcome, but won't be impossible to beat without grinding levels for a long time.

Examples of usability features will be easy movement options and quick, fluid battle controls. This is because I want the player to ave as fun a time as possible with the game, and be able to pick it up and play very quickly, with easy-to-follow controls, like single key presses in battles to attack an enemy or use items. Easy movement options and battle controls will mean the player will get invested into the game as soon as they pick it up, as there will be no frustration due to controls.

There will be no HUD for traversal if the menu hasn't been opened by the player, as it will be unnecessary clutter for the screen, and it would block the view for the player. For battles,the only display other than the sprites and backround will be the health of each character and enemy, and it will show who's turn it is with an arrow above the character who's turn it is. It will also show the attacks of the party character who's turn it is, with a key next to each move to show the key for the player to press for the character to use that move. This will make the game easier to play for the player, as it will always show which key the player will be able to press and will show the effect tied to that key, so the player won't need to memorize a lot of different controls in order to control the game effectively. The main menu will be quite basic, showing the title of the game and the 2 options: “New Game” and “Continue Game”.

There will be some boolean variables in my game, such as whether or not the player is in NG+, and if the player has beaten a boss fight, or entered a scripted scene.

There will be lists for enemies, party characters, bosses, items and walls so that, if, for example, I need to add wall, I can create the object and add it to the walls list, and iterate through the list for collision and scrolling, so that I don’t need to copy and paste every wall into each method, making my code much more efficient.

Most of the variables, however, will be real or integers, such as damage dealt by the player or enemies, and different stats for each character, like health, damage or defence.

There will be no need for validation in my game since I will make the code so that it only checks for correct inputs from the player so, if any invalid data is input by the player, nothing will happen.

Once the player saves the game, the values of every variable will be stored into a separate database to be called upon if the player dies in battle, or starts the program back up again and chooses “Continue Game”

Describe the Approach to Testing

My program will be fully object-oriented, so every function to be implemented into the game will be a method in one of the classes, such as movement options being a method in my player class. This will make it much easier to test each section of the code since every part in the HIPO chart will either be its own method or part of a larger method, so if something is not working, it will be easy to tell which part of the code will need to be altered. It will also mean that the code will be structured in an easy-to-read way, making it easier for a tester to look through the code to find bugs and make any alterations.

Once each section, described above in my HIPO chart, has been coded, such as traversal controls or placing sprites into the world, I will run the program to test whether or not everything works together and, if not, I will alter the code and test it again until everything up to that point works in tandem with all the rest.

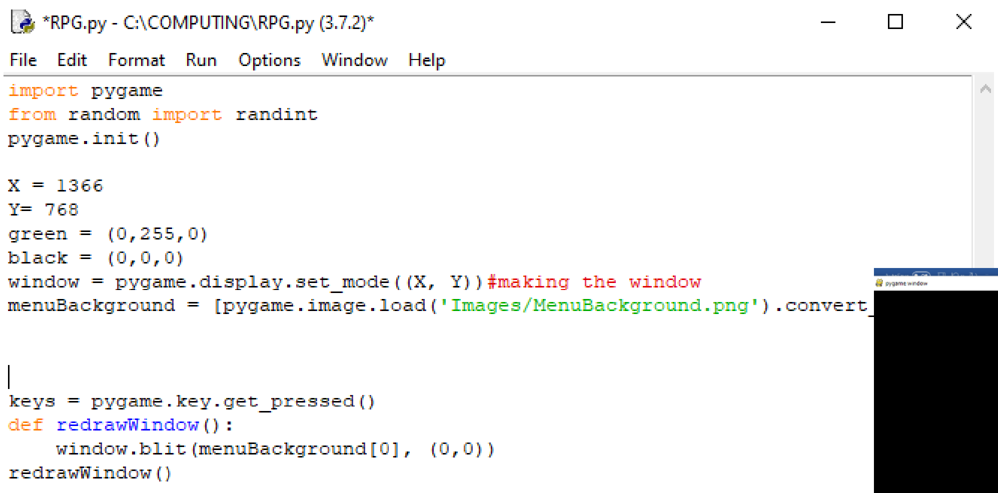
I will be able to generate normal data in battle by using the exact key inputs to run certain commands for each character. For example if ‘S’ causes the player to attack the enemy, I will input ’S’. I can generate extreme data by entering the correct key but not in the same case as shown in the game. Using the example above, I can input ‘s’ instead of ‘S’. I can generate abnormal data by inputting a key that will not work at that time. For example, if ‘S’ is the only key that will work, I can input ‘Q’ and it should either output an error message or do nothing instead of causing the program to crash.

Testing/Implementation

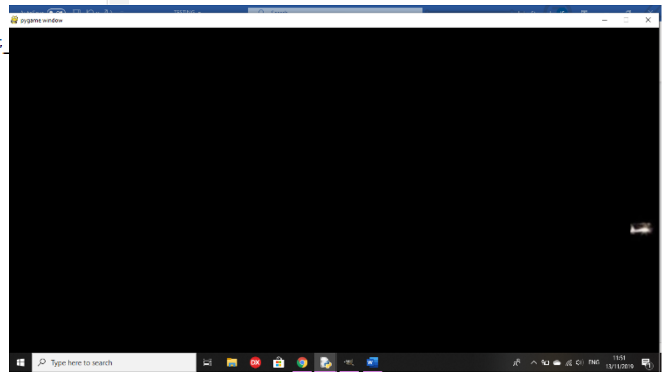
Function to test	Working?
New game function on the main menu- does it start the game from the beginning using the default variable values?	Yes
Continue game function on the main menu- does it start the game from where the player last saved, getting variable values from the database?	Yes
Movement using the arrow keys while traversing the world	Yes
Random encounters cause the window to switch from traversal to a battle	Yes
Scripted events- do they trigger at the right time and does dialogue show with the cutscenes?	Yes
Battle functions- does the character do the correct command based on the player's input?	Yes
Can the player control the party characters properly?	Yes
Battle functions- does the enemy attack the player's party effectively or buff themselves to make it harder for the player to win?	Yes
NG+- does the player enter NG+ when they finish the game, and does the extra boss trigger in NG+?	No
Does the true final boss trigger when the player has found all 3 key items?	Yes
Does the background change colour depending on the area that you're in?	Yes
Do the sidequests activate when you talk to an NPC?	Yes
Does the player get rewards for completing sidequests?	Yes

There is no validation testing in here as, the way I have coded it, the game only checks for correct inputs by the player so, if there is an incorrect input, the game does nothing.

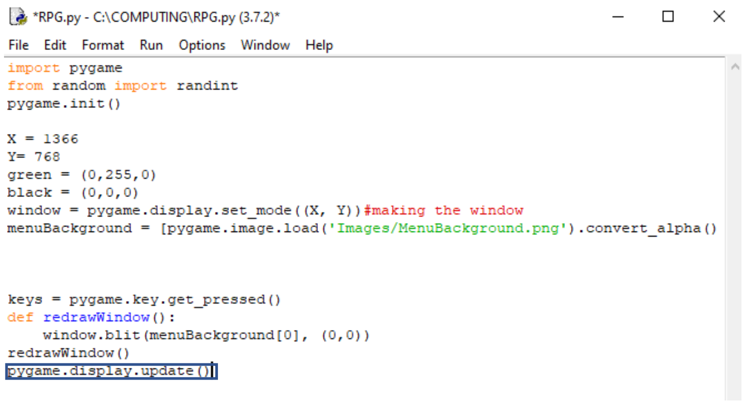
Test 1: Does the main menu image appear on the game menu when the code is run?


RPG.py - C:\COMPUTING\RPG.py (3.7.2)

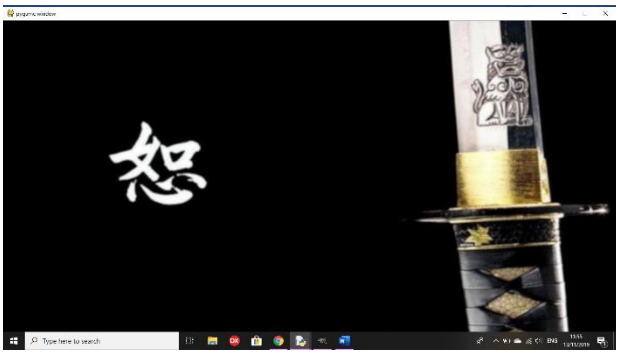
```
import pygame
from random import randint
pygame.init()

X = 1366
Y= 768
green = (0,255,0)
black = (0,0,0)
window = pygame.display.set_mode((X, Y))#making the window
menuBackground = [pygame.image.load('Images/MenuBackground.png').convert_


keys = pygame.key.get_pressed()
def redrawWindow():
    window.blit(menuBackground[0], (0,0))
redrawWindow()
```



Outcome: Does not load the full image on the window

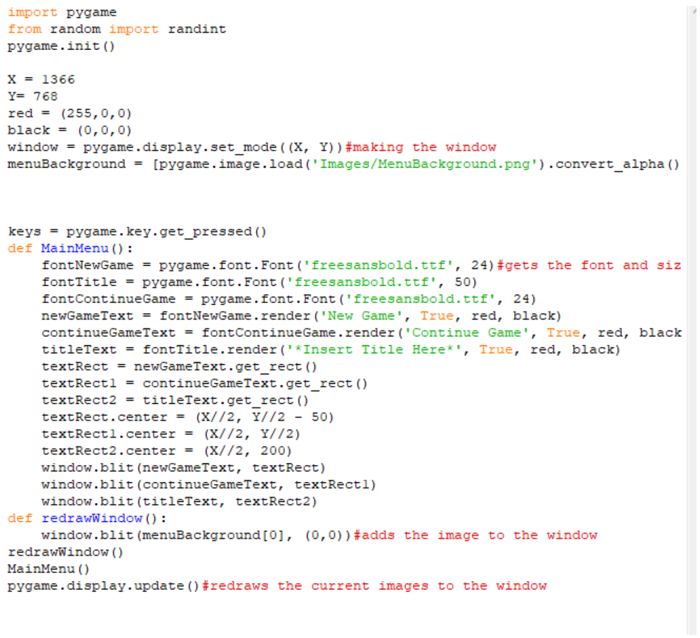

RPG.py - C:\COMPUTING\RPG.py (3.7.2)

```
import pygame
from random import randint
pygame.init()

X = 1366
Y= 768
green = (0,255,0)
black = (0,0,0)
window = pygame.display.set_mode((X, Y))#making the window
menuBackground = [pygame.image.load('Images/MenuBackground.png').convert_alpha()


keys = pygame.key.get_pressed()
def redrawWindow():
    window.blit(menuBackground[0], (0,0))
redrawWindow()
pygame.display.update()
```

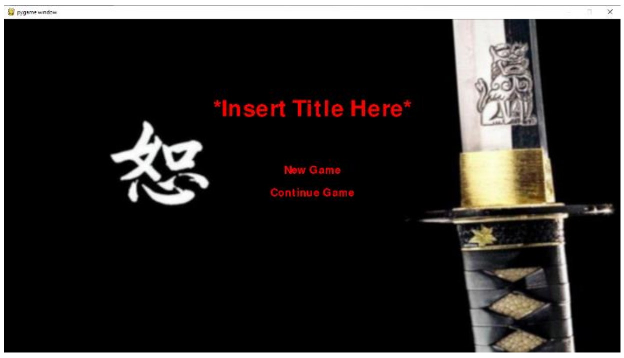


Remedy: ‘pygame.display.update()’ loads the full image onto the screen without having to move the window offscreen.

Test 2: does the text for the main menu appear on the window?

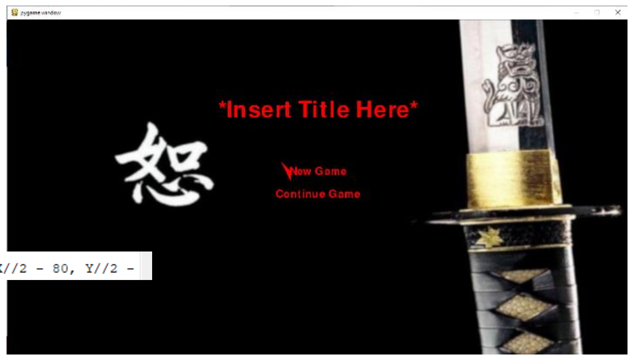


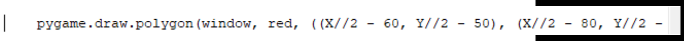
```
import pygame
from random import randint
pygame.init()

X = 1366
Y= 768
red = (255,0,0)
black = (0,0,0)
window = pygame.display.set_mode((X, Y))#making the window
menuBackground = [pygame.image.load('Images/MenuBackground.png').convert_alpha()


keys = pygame.key.get_pressed()
def MainMenu():
    fontNewGame = pygame.font.Font('freesansbold.ttf', 24)#gets the font and siz
    fontTitle = pygame.font.Font('freesansbold.ttf', 50)
    fontContinueGame = pygame.font.Font('freesansbold.ttf', 24)
    newGameText = fontNewGame.render('New Game', True, red, black)
    continueGameText = fontContinueGame.render('Continue Game', True, red, black
    titleText = fontTitle.render('*Insert Title Here*', True, red, black)
    textRect = newGameText.get_rect()
    textRect1 = continueGameText.get_rect()
    textRect2 = titleText.get_rect()
    textRect.center = (X//2, Y//2 - 50)
    textRect1.center = (X//2, Y//2)
    textRect2.center = (X//2, 200)
    window.blit(newGameText, textRect)
    window.blit(continueGameText, textRect1)
    window.blit(titleText, textRect2)
def redrawWindow():
    window.blit(menuBackground[0], (0,0))#adds the image to the window
redrawWindow()
MainMenu()
pygame.display.update()#redraws the current images to the window
```



Outcome: All of the text for the main menu has now been placed, with a placeholder for the title, but neither of the options have any functions yet.

Test 3: Does the arrow cursor appear on the menu options as intended?





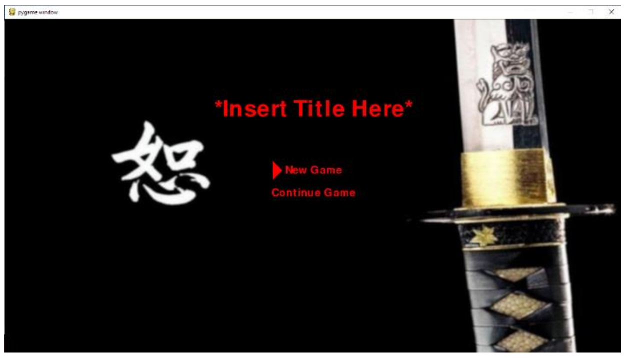
```
    pygame.draw.polygon(window, red, ((X//2 - 60, Y//2 - 50), (X//2 - 80, Y//2 -
```

Outcome: The cursor is drawn , but the vertices are not in the correct place to be to the left of ‘New Game’

Remedy:

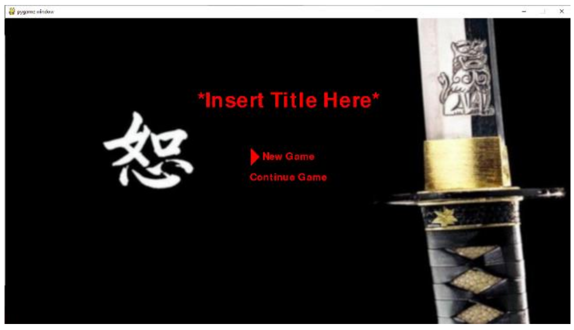
```
(X//2 - 70, Y//2 - 50), (X//2 - 90, Y//2 - 70), (X//2 - 90, Y//2 - 30)))
```

I changed the vertices of the cursor so it is drawn correctly.



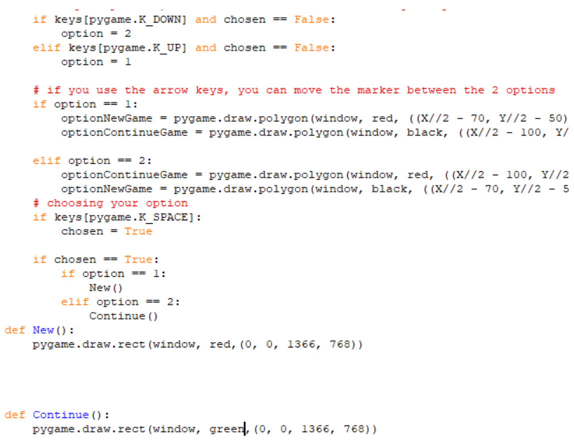
Test 4: Does the cursor move from ‘New Game’ to ‘Continue Game’ when the arrow keys are pressed?

```
if keys[pygame.K_DOWN]:
    option = 2
if keys[pygame.K_UP]:
    option = 1
# if you use the arrow keys, you can move the marker between the 2 options
if option == 1:
    optionNewGame = pygame.draw.polygon(window, red, ((X//2 - 70, Y//2 - 50)
elif option == 2:
    optionContinueGame = pygame.draw.polygon(window, red, ((X//2 - 70, Y//2)
```

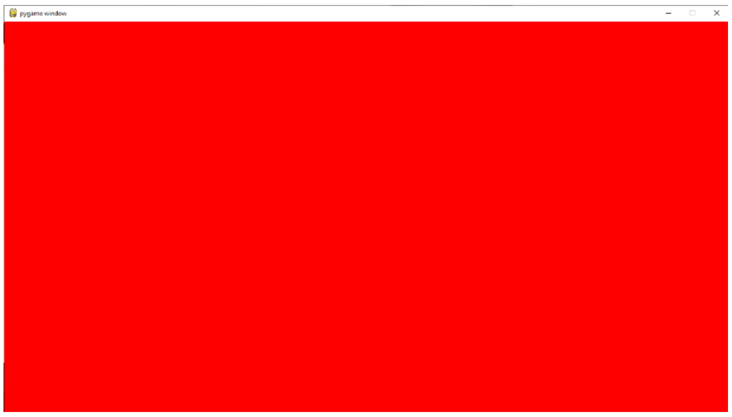


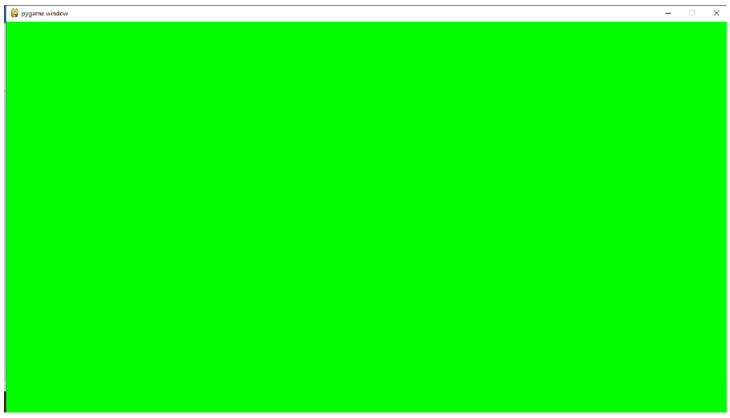
Outcome: The arrow marker should move between the 2 options as you press the arrow keys, but stays on ‘New Game’

Test 5: Do the new game and continue game options do as intended?

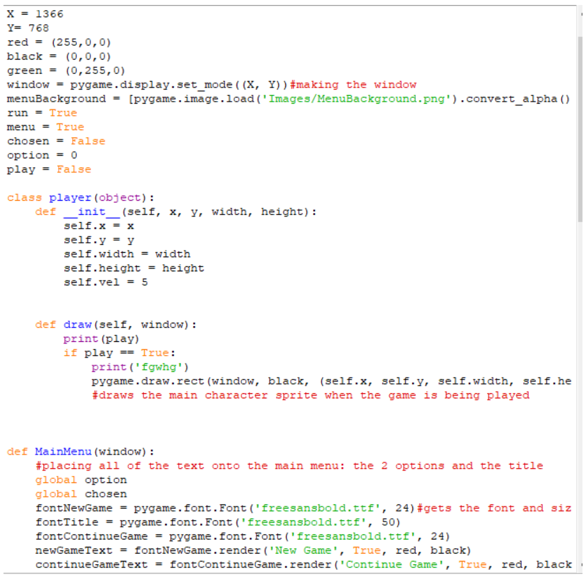


```
    if keys[pygame.K_DOWN] and chosen == False:
        option = 2
    elif keys[pygame.K_UP] and chosen == False:
        option = 1

    # if you use the arrow keys, you can move the marker between the 2 options
    if option == 1:
        optionNewGame = pygame.draw.polygon(window, red, ((X//2 - 70, Y//2 - 50)
        optionContinueGame = pygame.draw.polygon(window, black, ((X//2 - 100, Y/

    elif option == 2:
        optionContinueGame = pygame.draw.polygon(window, red, ((X//2 - 100, Y//2
        optionNewGame = pygame.draw.polygon(window, black, ((X//2 - 70, Y//2 - 5
    # choosing your option
    if keys[pygame.K_SPACE]:
        chosen = True

    if chosen == True:
        if option == 1:
            New()
        elif option == 2:
            Continue()
def New():
    pygame.draw.rect(window, red,(0, 0, 1366, 768))



def Continue():
    pygame.draw.rect(window, green,(0, 0, 1366, 768))
```



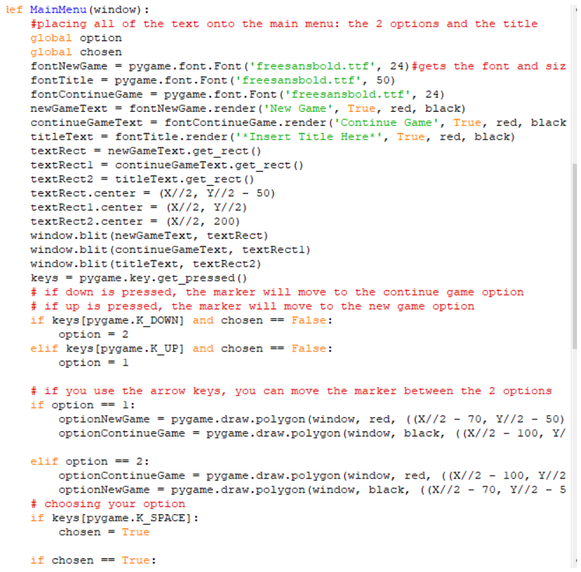


Outcome: I have used placeholders for new and continue game to show that the functions work as intended when the correct options are chosen.

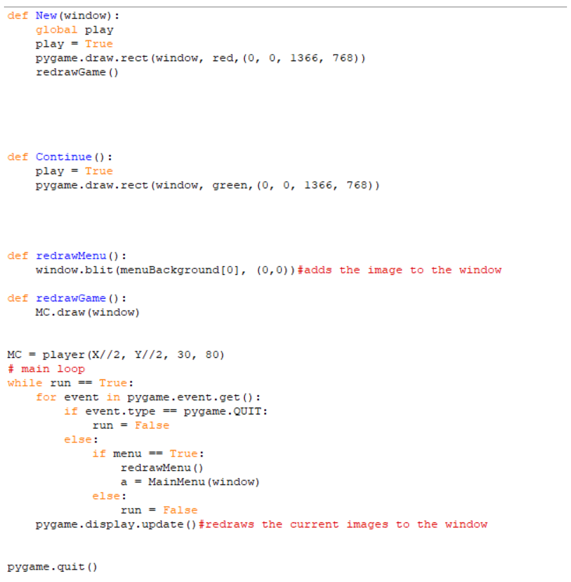


```
X = 1366
Y= 768
red = (255,0,0)
black = (0,0,0)
green = (0,255,0)
window = pygame.display.set_mode((X, Y))#making the window
menuBackground = [pygame.image.load('Images/MenuBackground.png').convert_alpha()
run = True
menu = True
chosen = False
option = 0
play = False

class player(object):
    def __init__(self, x, y, width, height):
        self.x = x
        self.y = y
        self.width = width
        self.height = height
        self.vel = 5


    def draw(self, window):
        print(play)
        if play == True:
            print('fgwhg')
            pygame.draw.rect(window, black, (self.x, self.y, self.width, self.he
            #draws the main character sprite when the game is being played



def MainMenu(window):
    #placing all of the text onto the main menu: the 2 options and the title
    global option
    global chosen
    fontNewGame = pygame.font.Font('freesansbold.ttf', 24)#gets the font and siz
    fontTitle = pygame.font.Font('freesansbold.ttf', 50)
    fontContinueGame = pygame.font.Font('freesansbold.ttf', 24)
    newGameText = fontNewGame.render('New Game', True, red, black)
    continueGameText = fontContinueGame.render('Continue Game', True, red, black
```

Review Point 1



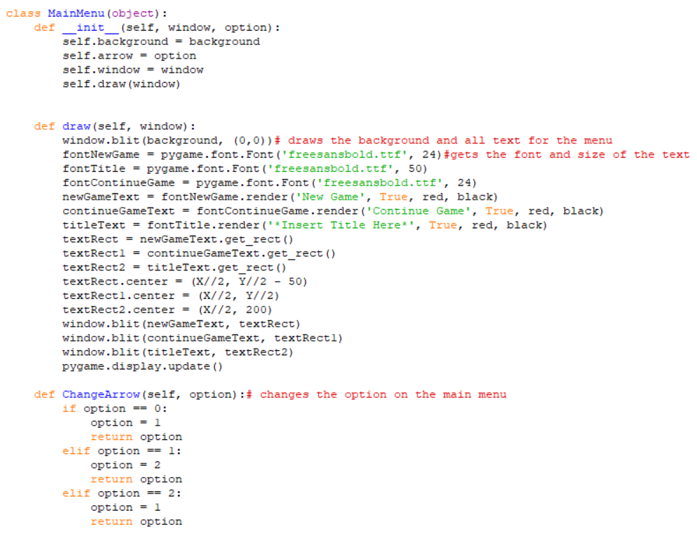
```
def MainMenu(window):
    #placing all of the text onto the main menu: the 2 options and the title
    global option
    global chosen
    fontNewGame = pygame.font.Font('freesansbold.ttf', 24)#gets the font and siz
    fontTitle = pygame.font.Font('freesansbold.ttf', 50)
    fontContinueGame = pygame.font.Font('freesansbold.ttf', 24)
    newGameText = fontNewGame.render('New Game', True, red, black)
    continueGameText = fontContinueGame.render('Continue Game', True, red, black
    titleText = fontTitle.render('*Insert Title Here*', True, red, black)
    textRect = newGameText.get_rect()
    textRect1 = continueGameText.get_rect()
    textRect2 = titleText.get_rect()
    textRect.center = (X//2, Y//2 - 50)
    textRect1.center = (X//2, Y//2)
    textRect2.center = (X//2, 200)
    window.blit(newGameText, textRect)
    window.blit(continueGameText, textRect1)
    window.blit(titleText, textRect2)
    keys = pygame.key.get_pressed()
    # if down is pressed, the marker will move to the continue game option
    # if up is pressed, the marker will move to the new game option
    if keys[pygame.K_DOWN] and chosen == False:
        option = 2
    elif keys[pygame.K_UP] and chosen == False:
        option = 1

    # if you use the arrow keys, you can move the marker between the 2 options
    if option == 1:
        optionNewGame = pygame.draw.polygon(window, red, ((X//2 - 70, Y//2 - 50)
        optionContinueGame = pygame.draw.polygon(window, black, ((X//2 - 100, Y/

    elif option == 2:
        optionContinueGame = pygame.draw.polygon(window, red, ((X//2 - 100, Y//2
        optionNewGame = pygame.draw.polygon(window, black, ((X//2 - 70, Y//2 - 5
    # choosing your option
    if keys[pygame.K_SPACE]:
        chosen = True

    if chosen == True:
```



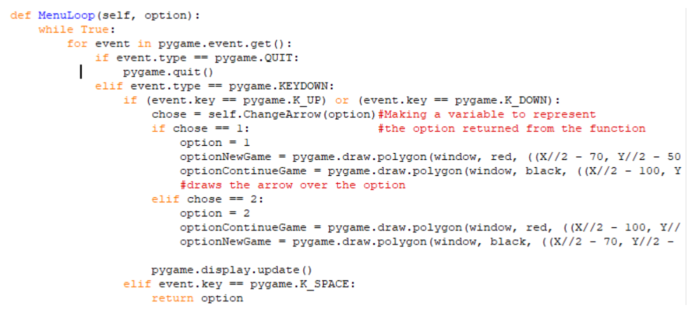
```
def New(window):
    global play
    play = True
    pygame.draw.rect(window, red,(0, 0, 1366, 768))
    redrawGame()




def Continue():
    play = True
    pygame.draw.rect(window, green,(0, 0, 1366, 768))




def redrawMenu():
    window.blit(menuBackground[0], (0,0))#adds the image to the window

def redrawGame():
    MC.draw(window)


MC = player(X//2, Y//2, 30, 80)
# main loop
while run == True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False
        else:
            if menu == True:
                redrawMenu()
                a = MainMenu(window)
            else:
                run = False
    pygame.display.update()#redraws the current images to the window


pygame.quit()
```

This is a good starting point for making my game, however there is a mix of OOP and external functions at the moment, making it easy to lose track of where variables are being sent to, so the code will be altered to make it so the whole code is object oriented, making it much easier to track.

Remedial action:



```
class MainMenu(object):
    def __init__(self, window, option):
        self.background = background
        self.arrow = option
        self.window = window
        self.draw(window)


    def draw(self, window):
        window.blit(background, (0,0))# draws the background and all text for the menu
        fontNewGame = pygame.font.Font('freesansbold.ttf', 24)#gets the font and size of the text
        fontTitle = pygame.font.Font('freesansbold.ttf', 50)
        fontContinueGame = pygame.font.Font('freesansbold.ttf', 24)
        newGameText = fontNewGame.render('New Game', True, red, black)
        continueGameText = fontContinueGame.render('Continue Game', True, red, black)
        titleText = fontTitle.render('*Insert Title Here*', True, red, black)
        textRect = newGameText.get_rect()
        textRect1 = continueGameText.get_rect()
        textRect2 = titleText.get_rect()
        textRect.center = (X//2, Y//2 - 50)
        textRect1.center = (X//2, Y//2)
        textRect2.center = (X//2, 200)
        window.blit(newGameText, textRect)
        window.blit(continueGameText, textRect1)
        window.blit(titleText, textRect2)
        pygame.display.update()

    def ChangeArrow(self, option):# changes the option on the main menu
        if option == 0:
            option = 1
            return option
        elif option == 1:
            option = 2
            return option
        elif option == 2:
            option = 1
            return option
```



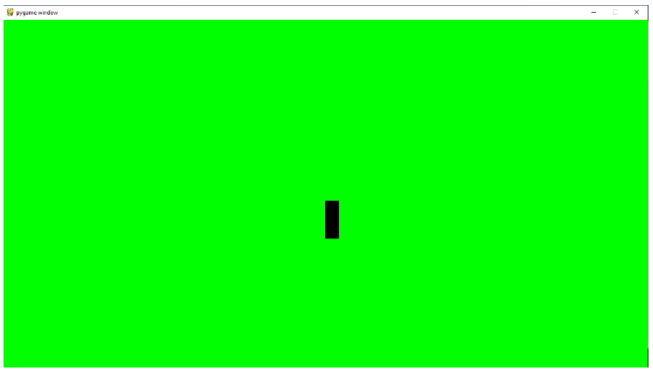
```
def MenuLoop(self, option):
    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
            elif event.type == pygame.KEYDOWN:
                if (event.key == pygame.K_UP) or (event.key == pygame.K_DOWN):
                    chose = self.ChangeArrow(option)#Making a variable to represent
                    if chose == 1:                  #the option returned from the function
                        option = 1
                        optionNewGame = pygame.draw.polygon(window, red, ((X//2 - 70, Y//2 - 50
                        optionContinueGame = pygame.draw.polygon(window, black, ((X//2 - 100, Y
                        #draws the arrow over the option
                    elif chose == 2:
                        option = 2
                        optionContinueGame = pygame.draw.polygon(window, red, ((X//2 - 100, Y//
                        optionNewGame = pygame.draw.polygon(window, black, ((X//2 - 70, Y//2 -

                    pygame.display.update()
                elif event.key == pygame.K_SPACE:
                    return option
```

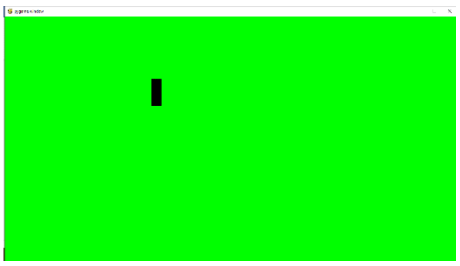
The main menu is now its own class, making it much easier to track variables going in and out of functions, and keeping the code much tidier and easier to see what is going on.

Test 6: Are the character and the background drawn onscreen?

```
MC = player(X//2, Y//2, 30, 80)
menu = MainMenu(window, option)
a = menu.MenuLoop(option)
game = Game(a)
game.NewOrContinue(a, window, MC)
```

```
class Game(object):
    def __init__(self, a):
        self.a = a

    def NewOrContinue(self, a, window, MC):#detects if the player chose new game or continue
        if self.a == 1:
            self.DrawNew(window, MC)
        elif self.a == 2:
            selfDrawLoad()


    def DrawNew(self, window, MC):
        pygame.draw.rect(window, green, (0,0,X,Y))
        MC.draw(window)
        pygame.display.update()
        self.Control(window, MC)

    def DrawLoad():
        pass #Will get variable values from database from when the player last saved

    def Control(self, window, MC):
        while True:
            for event in pygame.event.get():
                if event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_UP:
                        MC.y -= 5
                    if event.key == pygame.K_DOWN:
                        MC.y += 5
                    if event.key == pygame.K_LEFT:
                        MC.x -= 5
                    if event.key == pygame.K_RIGHT:
                        MC.x += 5
            pygame.draw.rect(window, green, (0,0,X,Y))
            MC.draw(window)
            pygame.display.update()
```

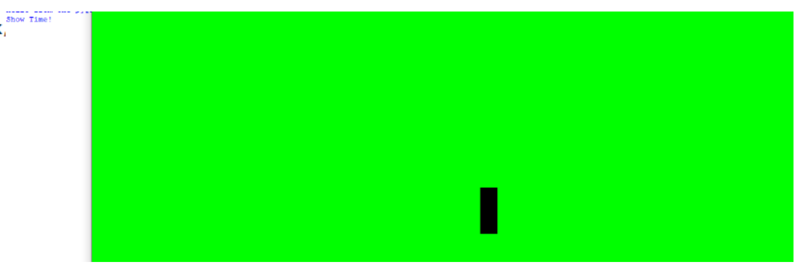




Outcome: The game now runs when the new game option is pressed, showing a green background and a character sprite being the black rectangle. The Control function allows the player to move the character using the arrow keys, as shown in the image on the right.

Test 7: Does the character stop moving when a battle is initiated?

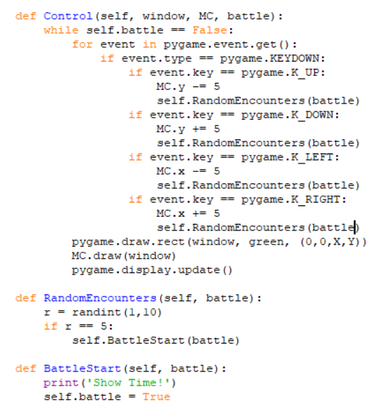
```
def Control(self, window, MC):
    while True:
        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_UP:
                    MC.y -= 5
                    self.RandomEncounters()
                if event.key == pygame.K_DOWN:
                    MC.y += 5
                    self.RandomEncounters()
                if event.key == pygame.K_LEFT:
                    MC.x -= 5
                    self.RandomEncounters()
                if event.key == pygame.K_RIGHT:
                    MC.x += 5
                    self.RandomEncounters()
        pygame.draw.rect(window, green, (0,0,X,
        MC.draw(window)
        pygame.display.update()

def RandomEncounters(self):
    r = randint(1,10)
    if r == 5:
        self.BattleStart()

def BattleStart(self):
    print('Show Time!')
```



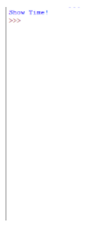
```
Show Time!
```

Outcome: The program goes to the BattleStart method as it should, but still allows the player to move around the space.

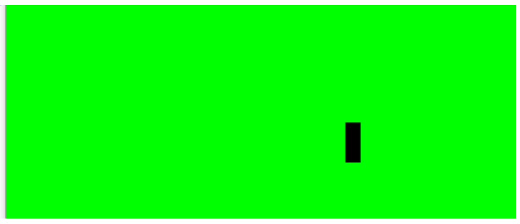
Remedy:



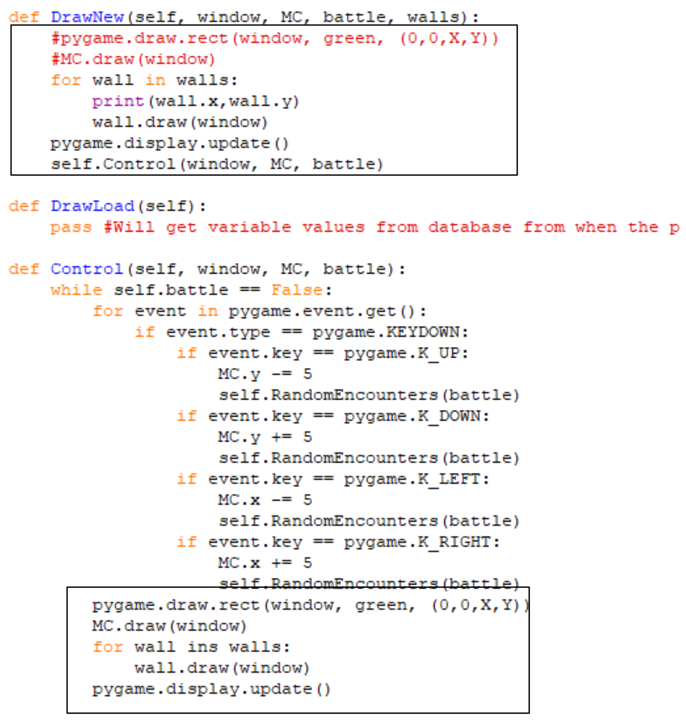
```
def Control(self, window, MC, battle):
    while self.battle == False:
        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_UP:
                    MC.y -= 5
                    self.RandomEncounters(battle)
                if event.key == pygame.K_DOWN:
                    MC.y += 5
                    self.RandomEncounters(battle)
                if event.key == pygame.K_LEFT:
                    MC.x -= 5
                    self.RandomEncounters(battle)
                if event.key == pygame.K_RIGHT:
                    MC.x += 5
                    self.RandomEncounters(battle)
        pygame.draw.rect(window, green, (0,0,X,Y))
        MC.draw(window)
        pygame.display.update()

def RandomEncounters(self, battle):
    r = randint(1,10)
    if r == 5:
        self.BattleStart(battle)

def BattleStart(self, battle):
    print('Show Time!')
    self.battle = True
```



```
Show Time!
>>>
```

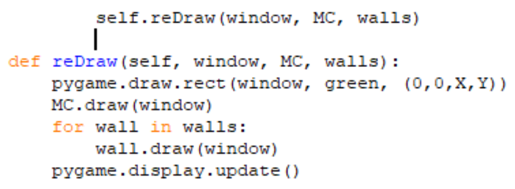


The code now ends when a random encounter is triggered, as you can no longer control the character when it is triggered, taking the game out of the control method, therefore ending the code.



```
def DrawNew(self, window, MC, battle, walls):
    #pygame.draw.rect(window, green, (0,0,X,Y))
    #MC.draw(window)
    for wall in walls:
        print(wall.x,wall.y)
        wall.draw(window)
    pygame.display.update()
    self.Control(window, MC, battle)

def DrawLoad(self):
    pass #Will get variable values from database from when the p

def Control(self, window, MC, battle):
    while self.battle == False:
        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_UP:
                    MC.y -= 5
                    self.RandomEncounters(battle)
                if event.key == pygame.K_DOWN:
                    MC.y += 5
                    self.RandomEncounters(battle)
                if event.key == pygame.K_LEFT:
                    MC.x -= 5
                    self.RandomEncounters(battle)
                if event.key == pygame.K_RIGHT:
                    MC.x += 5
                    self.RandomEncounters(battle)
        pygame.draw.rect(window, green, (0,0,X,Y))
        MC.draw(window)
        for wall ins walls:
            wall.draw(window)
        pygame.display.update()
```

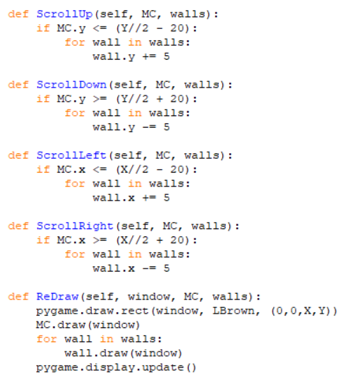
The code to draw the world is repeated in the control method, making the code less efficient, so I will need to make a 'ReDraw' method, instead of repeating the code, to redraw every object that is or will be onscreen.



```
        self.reDraw(window, MC, walls)
        |
def reDraw(self, window, MC, walls):
    pygame.draw.rect(window, green, (0,0,X,Y))
    MC.draw(window)
    for wall in walls:
        wall.draw(window)
    pygame.display.update()
```

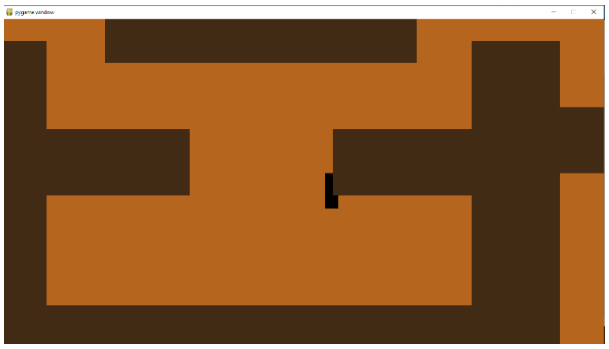
Remedy:

The code will now be more efficient since all I need to do is call the method to redraw all of the objects in the game world.



```
def ScrollUp(self, MC, walls):
    if MC.y <= (Y//2 - 20):
        for wall in walls:
            wall.y += 5

def ScrollDown(self, MC, walls):
    if MC.y >= (Y//2 + 20):
        for wall in walls:
            wall.y -= 5

def ScrollLeft(self, MC, walls):
    if MC.x <= (X//2 - 20):
        for wall in walls:
            wall.x += 5

def ScrollRight(self, MC, walls):
    if MC.x >= (X//2 + 20):
        for wall in walls:
            wall.x -= 5

def ReDraw(self, window, MC, walls):
    pygame.draw.rect(window, LBrown, (0,0,X,Y))
    MC.draw(window)
    for wall in walls:
        wall.draw(window)
    pygame.display.update()
```

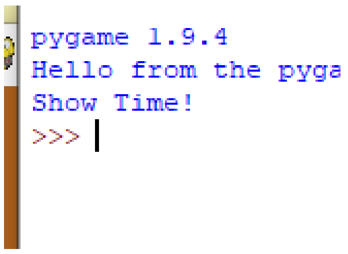
I have separate methods to scroll the objects if the main character is at a certain point on the screen, so they don't just go offscreen if they get too far, and the objects that were offscreen will be pulled onscreen if the player is going in that direction.



There is no collision on the walls yet, so I will make a collision method that will check the position of the walls against the player every time they make a movement.

Review Point 2

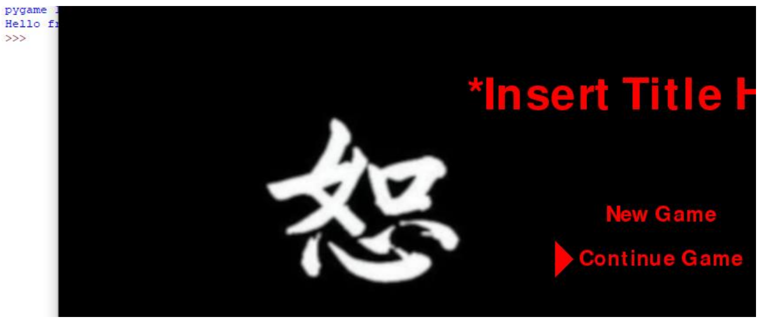
The collision on the walls somewhat works but, if you hit the side of the wall, it sends you straight to the top. (See test 8). Random encounters are triggered as you move around the world, but battles are not yet implemented. It only says 'Show time' then ends.



```
pygame 1.9.4
Hello from the pyga
Show Time!
>>> |
```

```
def RandomEncounters(self, battle):
    r = randint(1,30)
    if r == 5:
        self.BattleStart(battle)

def BattleStart(self, battle):
    print('Show Time!')
    self.battle = True
```

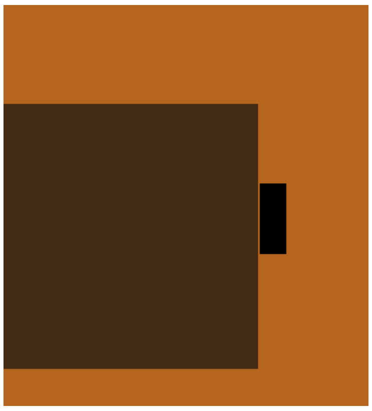
The 'Continue Game' option does not work yet as I have not implemented the save feature that would allow the player to start back from their save point yet. If you pick this option, the code ends as there is only a placeholder method for the 'Continue Game' option.

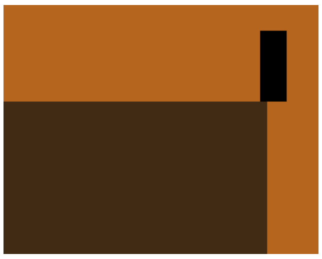


```
pygame 1
Hello fr
>>>
```

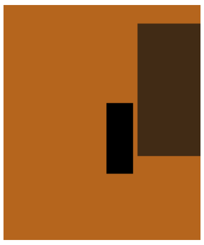
```
def DrawLoad(self):
    pass #Will get variable values from dat
```

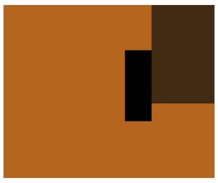
Test 8: Does the collision block you from moving into the walls as intended?





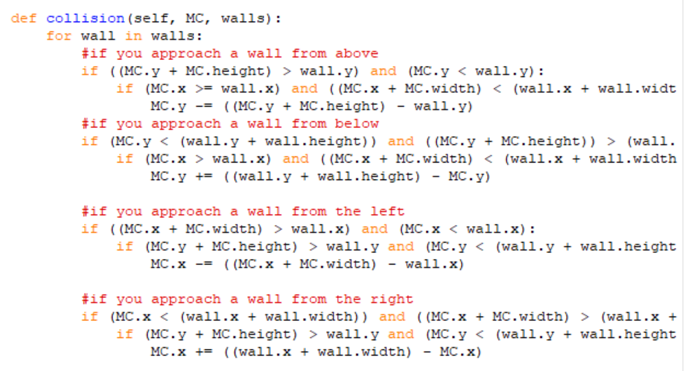
Outcome: At the moment, the collision does not work quite as intended as, if you hit the sides of the wall, you will be placed right at the top, rather than stopping at the side.





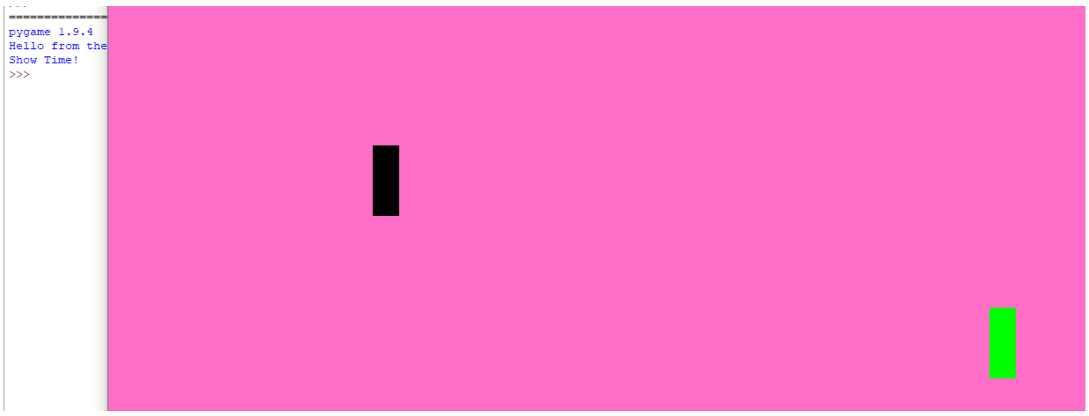
Remedy:

The character now colllides with the wall as intended, so it doesn't jump to the top of the wall when hitting the side of it.

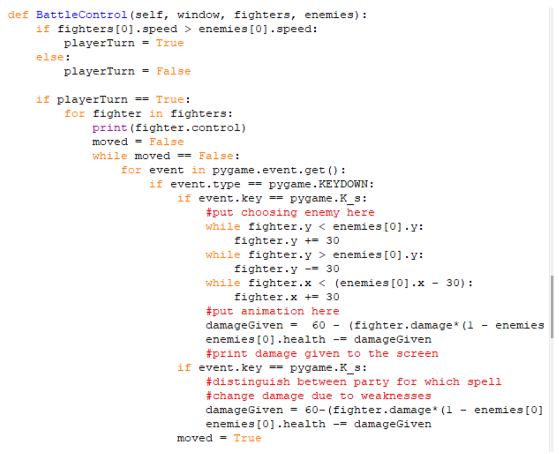


```
def collision(self, MC, walls):
    for wall in walls:
        #if you approach a wall from above
        if ((MC.y + MC.height) > wall.y) and (MC.y < wall.y):
            if (MC.x >= wall.x) and ((MC.x + MC.width) < (wall.x + wall.widt
                MC.y -= ((MC.y + MC.height) - wall.y)
        #if you approach a wall from below
        if (MC.y < (wall.y + wall.height)) and ((MC.y + MC.height)) > (wall.
            if (MC.x > wall.x) and ((MC.x + MC.width) < (wall.x + wall.width
                MC.y += ((wall.y + wall.height) - MC.y)

        #if you approach a wall from the left
        if ((MC.x + MC.width) > wall.x) and (MC.x < wall.x):
            if (MC.y + MC.height) > wall.y and (MC.y < (wall.y + wall.height
                MC.x -= ((MC.x + MC.width) - wall.x)

        #if you approach a wall from the right
        if (MC.x < (wall.x + wall.width)) and ((MC.x + MC.width) > (wall.x +
            if (MC.y + MC.height) > wall.y and (MC.y < (wall.y + wall.height
                MC.x += ((wall.x + wall.width) - MC.x)
```

Test 9: Is the battle screen drawn and does it allow for player input?



```
==============
pygame 1.9.4
Hello from the
Show Time!
>>>
```



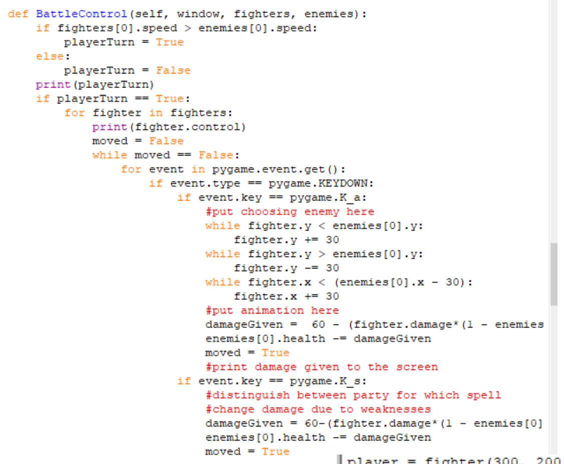
```
def BattleControl(self, window, fighters, enemies):
    if fighters[0].speed > enemies[0].speed:
        playerTurn = True
    else:
        playerTurn = False

    if playerTurn == True:
        for fighter in fighters:
            print(fighter.control)
            moved = False
            while moved == False:
                for event in pygame.event.get():
                    if event.type == pygame.KEYDOWN:
                        if event.key == pygame.K_s:
                            #put choosing enemy here
                            while fighter.y < enemies[0].y:
                                fighter.y += 30
                            while fighter.y > enemies[0].y:
                                fighter.y -= 30
                            while fighter.x < (enemies[0].x - 30):
                                fighter.x += 30
                            #put animation here
                            damageGiven =  60 - (fighter.damage*(1 - enemies
                            enemies[0].health -= damageGiven
                            #print damage given to the screen
                        if event.key == pygame.K_s:
                            #distinguish between party for which spell
                            #change damage due to weaknesses
                            damageGiven = 60-(fighter.damage*(1 - enemies[0]
                            enemies[0].health -= damageGiven
                        moved = True
```

Outcome:

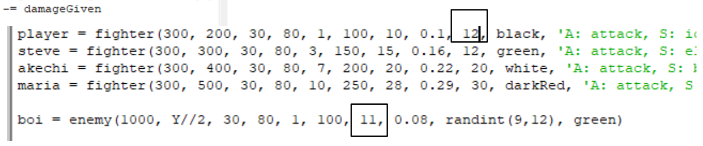
The battle screen is drawn, but the code ends as soon as it does so, whereas it should wait for a key input from the player.

Remedy:

I had to alter the speed stat of my character to be higher than the enemy's, because I haven't coded the enemy turn yet, so now it waits for the player's input.

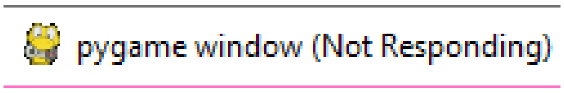


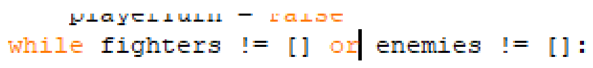
```
def BattleControl(self, window, fighters, enemies):
    if fighters[0].speed > enemies[0].speed:
        playerTurn = True
    else:
        playerTurn = False
    print(playerTurn)
    if playerTurn == True:
        for fighter in fighters:
            print(fighter.control)
            moved = False
            while moved == False:
                for event in pygame.event.get():
                    if event.type == pygame.KEYDOWN:
                        if event.key == pygame.K_a:
                            #put choosing enemy here
                            while fighter.y < enemies[0].y:
                                fighter.y += 30
                            while fighter.y > enemies[0].y:
                                fighter.y -= 30
                            while fighter.x < (enemies[0].x - 30):
                                fighter.x += 30
                            #put animation here
                            damageGiven =  60 - (fighter.damage*(1 - enemies
                            enemies[0].health -= damageGiven
                            moved = True
                            #print damage given to the screen
                        if event.key == pygame.K_s:
                            #distinguish between party for which spell
                            #change damage due to weaknesses
                            damageGiven = 60-(fighter.damage*(1 - enemies[0]
                            enemies[0].health -= damageGiven
                            moved = True
```



```
player = fighter(300, 200, 30, 80, 1, 100, 10, 0.1, 12, black, 'A: attack, S: i
steve = fighter(300, 300, 30, 80, 3, 150, 15, 0.16, 12, green, 'A: attack, S: e
akechi = fighter(300, 400, 30, 80, 7, 200, 20, 0.22, 20, white, 'A: attack, S: 
maria = fighter(300, 500, 30, 80, 10, 250, 28, 0.29, 30, darkRed, 'A: attack, S

boi = enemy(1000, Y//2, 30, 80, 1, 100, 11, 0.08, randint(9,12), green)
```

Test 10: Does the battle end and return to the overworld?


pygame window (Not Responding)

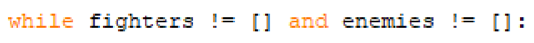

```
        playerTurn = False
    while fighters != [] or enemies != []:
```

Outcome:

The game crashes after all of the enemies on screen have been killed.

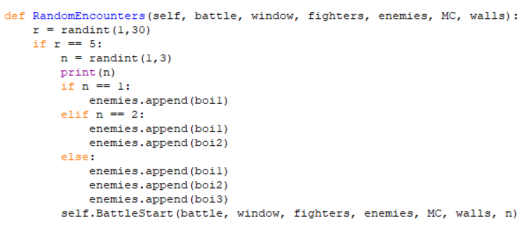
This is because it was put into an infinite loop since there were no enemies to kill the remaining fighters.

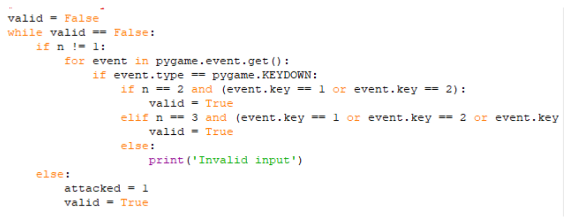
Remedy:

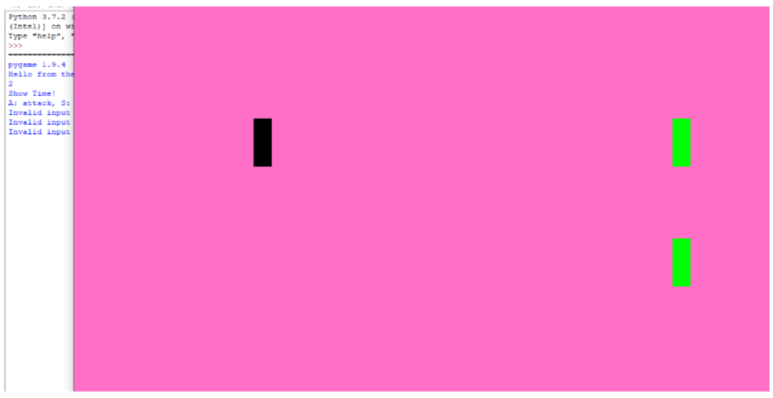

```
    while fighters != [] and enemies != []:
```

The battle now ends once all the elements of either list have been removed, eliminating the infinite loop

Test 12: When there are multiple enemies, can the player choose who to attack?

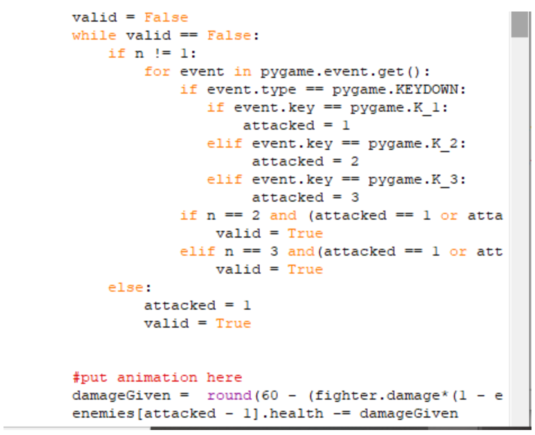

```
def RandomEncounters(self, battle, window, fighters, enemies, MC, walls):
    r = randint(1,30)
    if r == 5:
        n = randint(1,3)
        print(n)
        if n == 1:
            enemies.append(boi1)
        elif n == 2:
            enemies.append(boi1)
            enemies.append(boi2)
        else:
            enemies.append(boi1)
            enemies.append(boi2)
            enemies.append(boi3)
        self.BattleStart(battle, window, fighters, enemies, MC, walls, n)
```

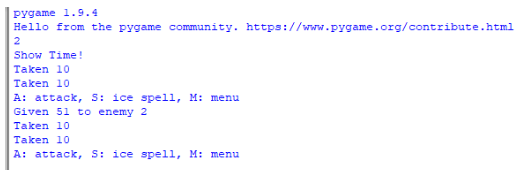

```
    valid = False
    while valid == False:
        if n != 1:
            for event in pygame.event.get():
                if event.type == pygame.KEYDOWN:
                    if n == 2 and (event.key == 1 or event.key == 2):
                        valid = True
                    elif n == 3 and (event.key == 1 or event.key == 2 or event.key
                        valid = True
                    else:
                        print('Invalid input')
        else:
            attacked = 1
            valid = True
```


```
Python 3.7.2 (
(Intel)] on wi
Type "help", "
>>>
--------------
pygame 1.9.4
Hello from the
2
Show Time!
A: attack, S:
Invalid input
Invalid input
Invalid input
```

Outcome: All inputs are ‘invalid inputs’, so you cannot attack any enemies

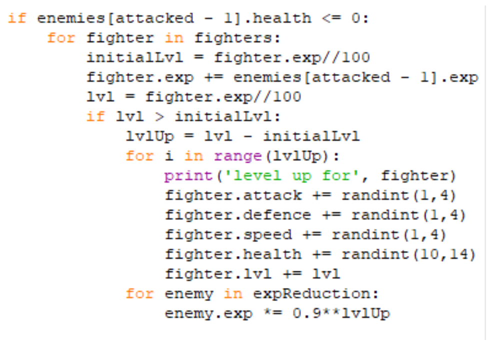
Remedy:


```
    valid = False
    while valid == False:
        if n != 1:
            for event in pygame.event.get():
                if event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_1:
                        attacked = 1
                    elif event.key == pygame.K_2:
                        attacked = 2
                    elif event.key == pygame.K_3:
                        attacked = 3
                if n == 2 and (attacked == 1 or atta
                    valid = True
                elif n == 3 and(attacked == 1 or att
                    valid = True
        else:
            attacked = 1
            valid = True


    #put animation here
    damageGiven =  round(60 - (fighter.damage*(1 - e
    enemies[attacked - 1].health -= damageGiven
```

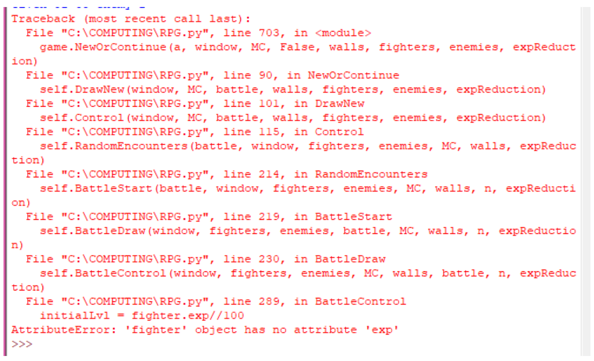

```
pygame 1.9.4
Hello from the pygame community. https://www.pygame.org/contribute.html
2
Show Time!
Taken 10
Taken 10
A: attack, S: ice spell, M: menu
Given 51 to enemy 2
Taken 10
Taken 10
A: attack, S: ice spell, M: menu
```

I gave ‘attacked’ a value based on the input, which determines the enemy that is attacked by the player

Test 13: does the player gain EXP upon killing enemies?



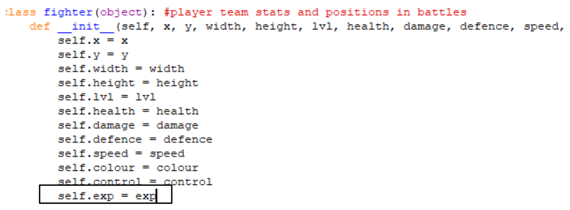
```python
if enemies[attacked - 1].health <= 0:
    for fighter in fighters:
        initialLvl = fighter.exp//100
        fighter.exp += enemies[attacked - 1].exp
        lvl = fighter.exp//100
        if lvl > initialLvl:
            lvlUp = lvl - initialLvl
            for i in range(lvlUp):
                print('level up for', fighter)
                fighter.attack += randint(1,4)
                fighter.defence += randint(1,4)
                fighter.speed += randint(1,4)
                fighter.health += randint(10,14)
                fighter.lvl += lvl
            for enemy in expReduction:
                enemy.exp *= 0.9**lvlUp
```



```
Traceback (most recent call last):
  File "C:\COMPUTING\RPG.py", line 703, in <module>
    game.NewOrContinue(a, window, MC, False, walls, fighters, enemies, expReduct
ion)
  File "C:\COMPUTING\RPG.py", line 90, in NewOrContinue
    self.DrawNew(window, MC, battle, walls, fighters, enemies, expReduction)
  File "C:\COMPUTING\RPG.py", line 101, in DrawNew
    self.Control(window, MC, battle, walls, fighters, enemies, expReduction)
  File "C:\COMPUTING\RPG.py", line 115, in Control
    self.RandomEncounters(battle, window, fighters, enemies, MC, walls, expReduc
tion)
  File "C:\COMPUTING\RPG.py", line 214, in RandomEncounters
    self.BattleStart(battle, window, fighters, enemies, MC, walls, n, expReducti
on)
  File "C:\COMPUTING\RPG.py", line 219, in BattleStart
    self.BattleDraw(window, fighters, enemies, battle, MC, walls, n, expReductio
n)
  File "C:\COMPUTING\RPG.py", line 230, in BattleDraw
    self.BattleControl(window, fighters, enemies, MC, walls, battle, n, expReduc
tion)
  File "C:\COMPUTING\RPG.py", line 289, in BattleControl
    initialLvl = fighter.exp//100
AttributeError: 'fighter' object has no attribute 'exp'
>>>
```

Outcome: The fighter does not have an EXP value

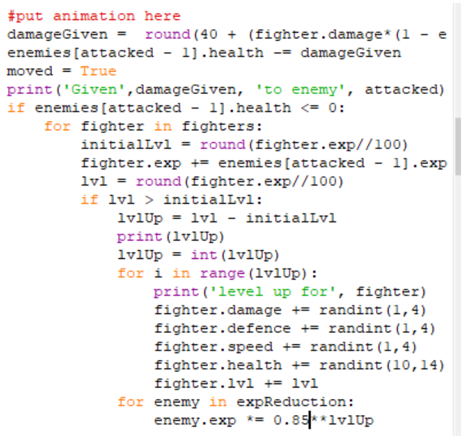
Remedy: exp is added to the initialisation method for fighter



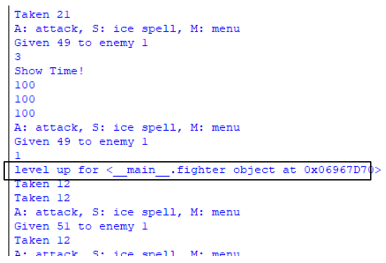
```python
:lass fighter(object): #player team stats and positions in battles
    def __init__(self, x, y, width, height, lvl, health, damage, defence, speed,
        self.x = x
        self.y = y
        self.width = width
        self.height = height
        self.lvl = lvl
        self.health = health
        self.damage = damage
        self.defence = defence
        self.speed = speed
        self.colour = colour
        self.control = control
        self.exp = exp
```

Test 14: Does the player level up upon receiving enough EXP?

Outcome: The player levels up upon receiving 100 EXP and the exp gain from enemies reduces upon each level gain.



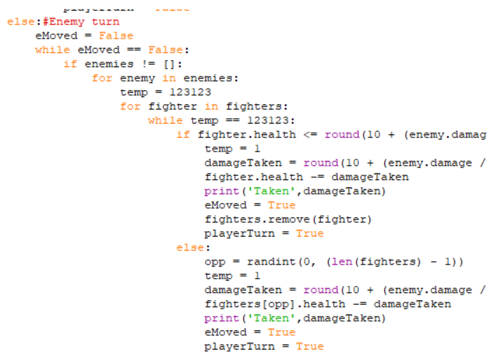
```python
#put animation here
damageGiven =  round(40 + (fighter.damage*(1 - e
enemies[attacked - 1].health -= damageGiven
moved = True
print('Given',damageGiven, 'to enemy', attacked)
if enemies[attacked - 1].health <= 0:
    for fighter in fighters:
        initialLvl = round(fighter.exp//100)
        fighter.exp += enemies[attacked - 1].exp
        lvl = round(fighter.exp//100)
        if lvl > initialLvl:
            lvlUp = lvl - initialLvl
            print(lvlUp)
            lvlUp = int(lvlUp)
            for i in range(lvlUp):
                print('level up for', fighter)
                fighter.damage += randint(1,4)
                fighter.defence += randint(1,4)
                fighter.speed += randint(1,4)
                fighter.health += randint(10,14)
                fighter.lvl += lvl
            for enemy in expReduction:
                enemy.exp *= 0.85**lvlUp
```



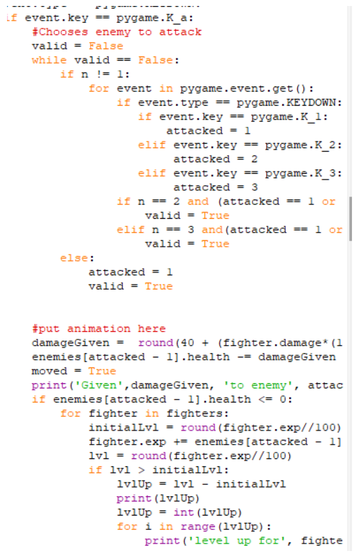
```
Taken 21
A: attack, S: ice spell, M: menu
Given 49 to enemy 1
3
Show Time!
100
100
100
A: attack, S: ice spell, M: menu
Given 49 to enemy 1
1
level up for <__main__.fighter object at 0x06967D70>
Taken 12
Taken 12
A: attack, S: ice spell, M: menu
Given 51 to enemy 1
Taken 12
```

<u>Review Point 3</u>

Random encounters cause battles to take place, with the player being able to attack the enemies, and the enemies being able to attack the player. The enemies drop EXP upon being defeated and the player characters will level up once they gain enough experience. The enemies do not yet drop items when defeated, the player cannot yet use items during battles and the enemies cannot buff themselves yet.

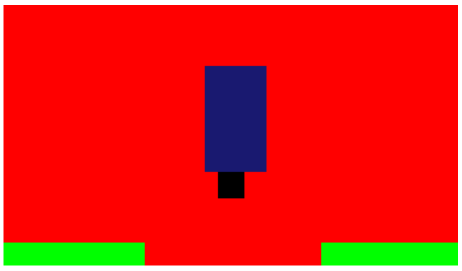


```
else:#Enemy turn
    eMoved = False
    while eMoved == False:
        if enemies != []:
            for enemy in enemies:
                temp = 123123
                for fighter in fighters:
                    while temp == 123123:
                        if fighter.health <= round(10 + (enemy.damag
                            temp = 1
                            damageTaken = round(10 + (enemy.damage /
                            fighter.health -= damageTaken
                            print('Taken',damageTaken)
                            eMoved = True
                            fighters.remove(fighter)
                            playerTurn = True
                        else:
                            opp = randint(0, (len(fighters) - 1))
                            temp = 1
                            damageTaken = round(10 + (enemy.damage /
                            fighters[opp].health -= damageTaken
                            print('Taken',damageTaken)
                            eMoved = True
                            playerTurn = True
```



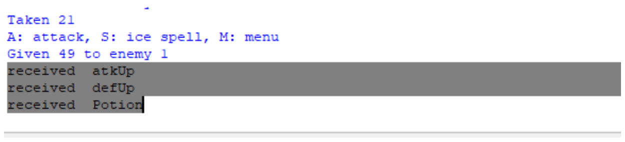
```
if event.key == pygame.K_a:
    #Chooses enemy to attack
    valid = False
    while valid == False:
        if n != 1:
            for event in pygame.event.get():
                if event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_1:
                        attacked = 1
                    elif event.key == pygame.K_2:
                        attacked = 2
                    elif event.key == pygame.K_3:
                        attacked = 3
                if n == 2 and (attacked == 1 or
                    valid = True
                elif n == 3 and(attacked == 1 or
                    valid = True
        else:
            attacked = 1
            valid = True


    #put animation here
    damageGiven =  round(40 + (fighter.damage*(1
    enemies[attacked - 1].health -= damageGiven
    moved = True
    print('Given',damageGiven, 'to enemy', attac
    if enemies[attacked - 1].health <= 0:
        for fighter in fighters:
            initialLvl = round(fighter.exp//100)
            fighter.exp += enemies[attacked - 1]
            lvl = round(fighter.exp//100)
            if lvl > initialLvl:
                lvlUp = lvl - initialLvl
                print(lvlUp)
                lvlUp = int(lvlUp)
                for i in range(lvlUp):
                    print('level up for', fighte
```

The sprite for the first boss is drawn to the screen, but you cannot yet encounter him as the function to encounter a boss has not yet been made.

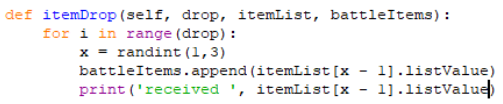


Test 15: Do enemies drop battle items upon defeat?

Outcome: Upon ending the battle, each enemy drop one of 3 battle items



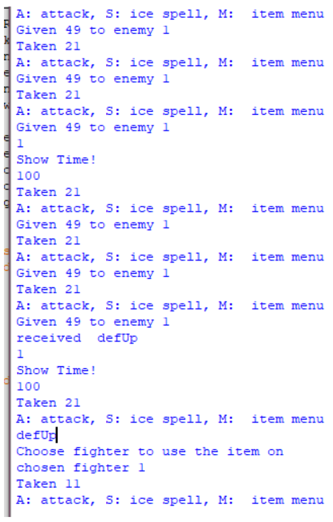
```
Taken 21
A: attack, S: ice spell, M: menu
Given 49 to enemy 1
received  atkUp
received  defUp
received  Potion
```



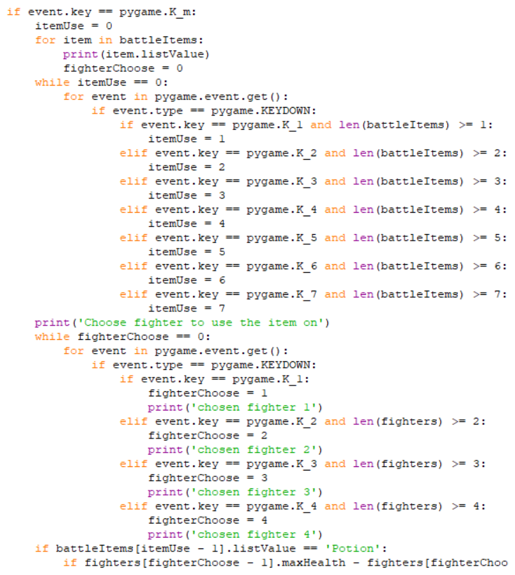
```
def itemDrop(self, drop, itemList, battleItems):
    for i in range(drop):
        x = randint(1,3)
        battleItems.append(itemList[x - 1].listValue)
        print('received ', itemList[x - 1].listValue)
```

Test 16: Do items work in battle?

Outcome: In this example, a defUp is used in battle, permanently increasing the character's defence stat.



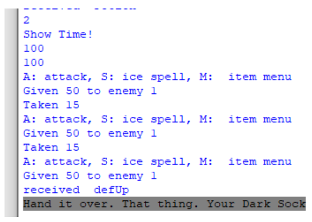
```
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
1
Show Time!
100
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
received  defUp
1
Show Time!
100
Taken 21
A: attack, S: ice spell, M:  item menu
defUp
Choose fighter to use the item on
chosen fighter 1
Taken 11
A: attack, S: ice spell, M:  item menu
```



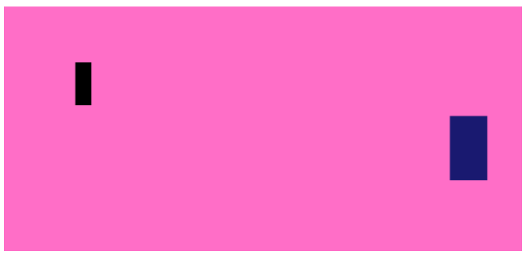
```
if event.key == pygame.K_m:
    itemUse = 0
    for item in battleItems:
        print(item.listValue)
        fighterChoose = 0
    while itemUse == 0:
        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_1 and len(battleItems) >= 1:
                    itemUse = 1
                elif event.key == pygame.K_2 and len(battleItems) >= 2:
                    itemUse = 2
                elif event.key == pygame.K_3 and len(battleItems) >= 3:
                    itemUse = 3
                elif event.key == pygame.K_4 and len(battleItems) >= 4:
                    itemUse = 4
                elif event.key == pygame.K_5 and len(battleItems) >= 5:
                    itemUse = 5
                elif event.key == pygame.K_6 and len(battleItems) >= 6:
                    itemUse = 6
                elif event.key == pygame.K_7 and len(battleItems) >= 7:
                    itemUse = 7
    print('Choose fighter to use the item on')
    while fighterChoose == 0:
        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_1:
                    fighterChoose = 1
                    print('chosen fighter 1')
                elif event.key == pygame.K_2 and len(fighters) >= 2:
                    fighterChoose = 2
                    print('chosen fighter 2')
                elif event.key == pygame.K_3 and len(fighters) >= 3:
                    fighterChoose = 3
                    print('chosen fighter 3')
                elif event.key == pygame.K_4 and len(fighters) >= 4:
                    fighterChoose = 4
                    print('chosen fighter 4')
    if battleItems[itemUse - 1].listValue == 'Potion':
        if fighters[fighterChoose - 1].maxHealth - fighters[fighterChoo
```

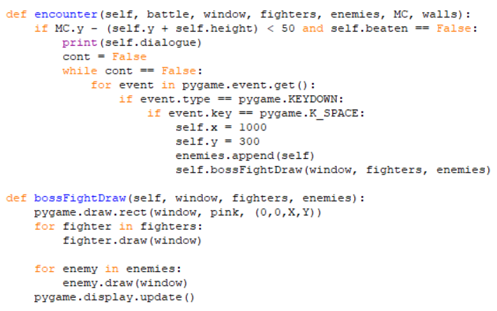
Test 17: Does the boss fight begin when the dialogue has been shown for the boss?

Outcome: When the player presses space after the dialogue is shown, the battle scene is drawn.



```
2
Show Time!
100
100
A: attack, S: ice spell, M:  item menu
Given 50 to enemy 1
Taken 15
A: attack, S: ice spell, M:  item menu
Given 50 to enemy 1
Taken 15
A: attack, S: ice spell, M:  item menu
Given 50 to enemy 1
received  defUp
Hand it over. That thing. Your Dark Sock
```

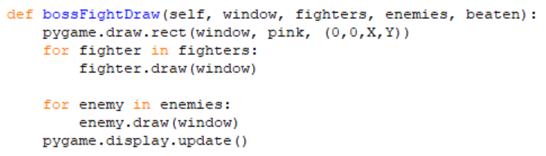




```
def encounter(self, battle, window, fighters, enemies, MC, walls):
    if MC.y - (self.y + self.height) < 50 and self.beaten == False:
        print(self.dialogue)
        cont = False
        while cont == False:
            for event in pygame.event.get():
                if event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_SPACE:
                        self.x = 1000
                        self.y = 300
                        enemies.append(self)
                        self.bossFightDraw(window, fighters, enemies)

def bossFightDraw(self, window, fighters, enemies):
    pygame.draw.rect(window, pink, (0,0,X,Y))
    for fighter in fighters:
        fighter.draw(window)

    for enemy in enemies:
        enemy.draw(window)
    pygame.display.update()
```

Test 18: Does the boss fight control work?

Outcome: I forgot to call the method, so you cannot attack the boss.



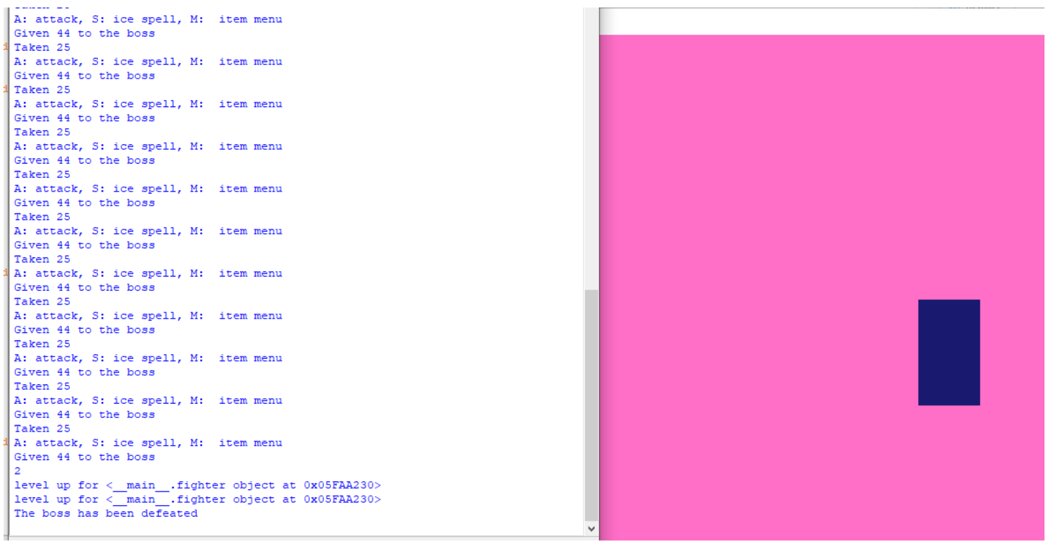
```
def bossFightDraw(self, window, fighters, enemies, beaten):
    pygame.draw.rect(window, pink, (0,0,X,Y))
    for fighter in fighters:
        fighter.draw(window)

    for enemy in enemies:
        enemy.draw(window)
    pygame.display.update()
```

Remedy: I called the method in the bossFightDraw method so you will be able to attack the boss.

```
def bossFightDraw(self, window, fighters, enemies, beaten):
    pygame.draw.rect(window, pink, (0,0,X,Y))
    for fighter in fighters:
        fighter.draw(window)

    for enemy in enemies:
        enemy.draw(window)
    pygame.display.update()
    self.bossFightControl(fighters, enemies, beaten)
```

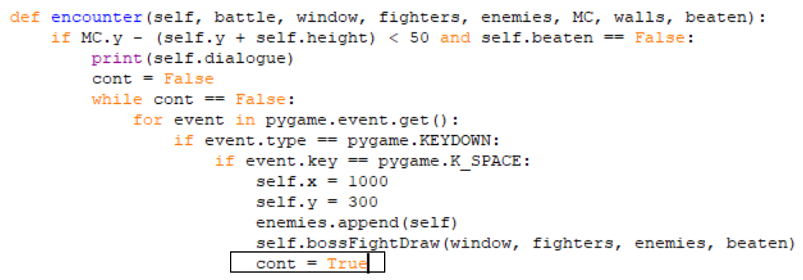
Test 19: When the boss is defeated, does gameplay continue in the overworld?

Outcome: The game acknowledges that the boss is defeated but does not return to the overworld.



```
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
Taken 25
A: attack, S: ice spell, M:  item menu
Given 44 to the boss
2
level up for <__main__.fighter object at 0x05FAA230>
level up for <__main__.fighter object at 0x05FAA230>
The boss has been defeated
```

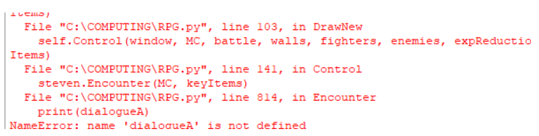
Remedy: I made the cont variable true after the fight, causing the fight to end and the game to carry on.



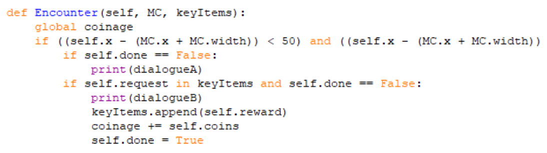
```
def encounter(self, battle, window, fighters, enemies, MC, walls, beaten):
    if MC.y - (self.y + self.height) < 50 and self.beaten == False:
        print(self.dialogue)
        cont = False
        while cont == False:
            for event in pygame.event.get():
                if event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_SPACE:
                        self.x = 1000
                        self.y = 300
                        enemies.append(self)
                        self.bossFightDraw(window, fighters, enemies, beaten)
                        cont = True
```

Test 20: Does the NPC speak its first dialogue line when interacted with?

Outcome: The interaction causes an error.

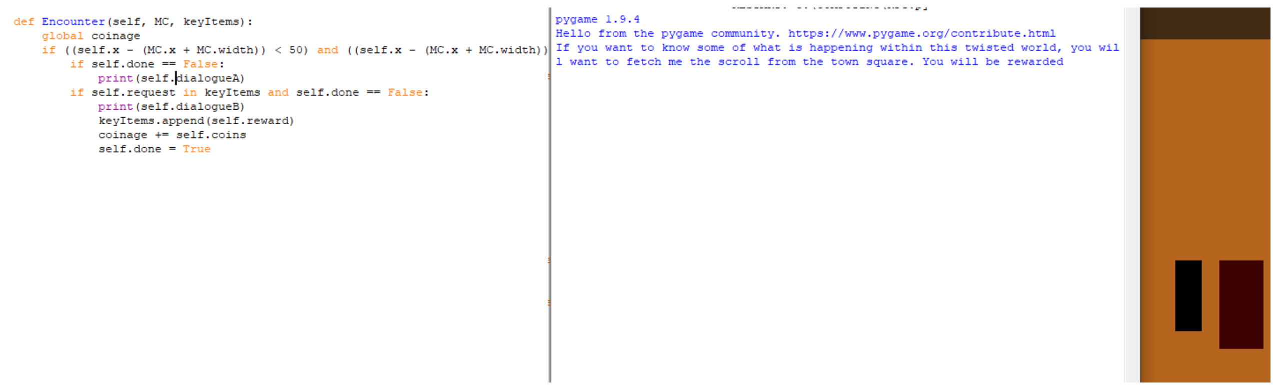


```
  File "C:\COMPUTING\RPG.py", line 103, in DrawNew
    self.Control(window, MC, battle, walls, fighters, enemies, expReductio
Items)
  File "C:\COMPUTING\RPG.py", line 141, in Control
    steven.Encounter(MC, keyItems)
  File "C:\COMPUTING\RPG.py", line 814, in Encounter
    print(dialogueA)
NameError: name 'dialogueA' is not defined
```



```
def Encounter(self, MC, keyItems):
    global coinage
    if ((self.x - (MC.x + MC.width)) < 50) and ((self.x - (MC.x + MC.width))
        if self.done == False:
            print(dialogueA)
        if self.request in keyItems and self.done == False:
            print(dialogueB)
            keyItems.append(self.reward)
            coinage += self.coins
            self.done = True
```

Remedy: I replaced ‘dialogueA’ and ‘dialogueB’ with ‘self.dialogueA’ and self.dialogueB’ respectively.



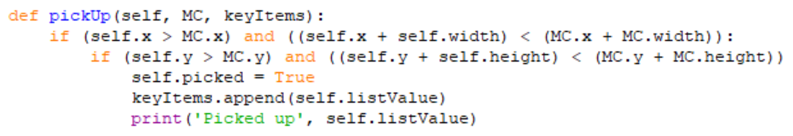
```
def Encounter(self, MC, keyItems):
    global coinage
    if ((self.x - (MC.x + MC.width)) < 50) and ((self.x - (MC.x + MC.width))
        if self.done == False:
            print(self.dialogueA)
        if self.request in keyItems and self.done == False:
            print(self.dialogueB)
            keyItems.append(self.reward)
            coinage += self.coins
            self.done = True
```

```
pygame 1.9.4
Hello from the pygame community. https://www.pygame.org/contribute.html
If you want to know some of what is happening within this twisted world, you wil
l want to fetch me the scroll from the town square. You will be rewarded
```

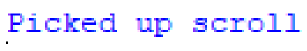
Test 21: Can the player pick up an overworld item and have it be placed in the keyItems list?

Outcome:

When the player is over the item and presses space, the item is picked up and placed in keyItems.



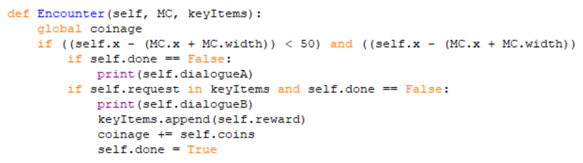
```
def pickUp(self, MC, keyItems):
    if (self.x > MC.x) and ((self.x + self.width) < (MC.x + MC.width)):
        if (self.y > MC.y) and ((self.y + self.height) < (MC.y + MC.height))
            self.picked = True
            keyItems.append(self.listValue)
            print('Picked up', self.listValue)
```



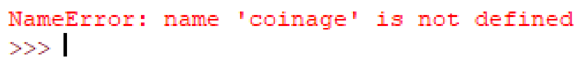
```
Picked up scroll
```

Test 22: Can the player finish the sidequest when they have the correct item in their inventory?

Outcome: The player can complete the sidequest but the rewards don't work as I haven't defined 'Coinage' before adding to it.

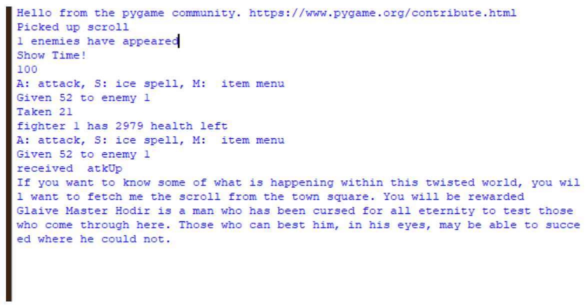


```
def Encounter(self, MC, keyItems):
    global coinage
    if ((self.x - (MC.x + MC.width)) < 50) and ((self.x - (MC.x + MC.width))
        if self.done == False:
            print(self.dialogueA)
        if self.request in keyItems and self.done == False:
            print(self.dialogueB)
            keyItems.append(self.reward)
            coinage += self.coins
            self.done = True
```

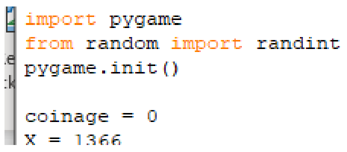


```
NameError: name 'coinage' is not defined
>>> 
```

Remedy: I set 'coinage' to 0 at the top of the program.



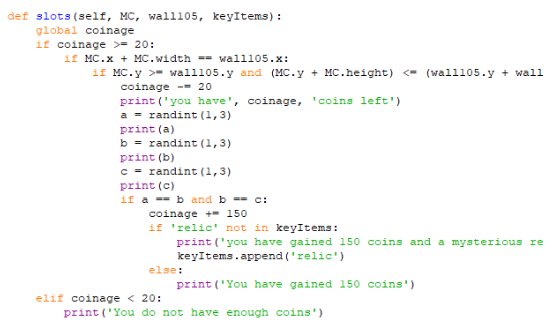
```
Hello from the pygame community. https://www.pygame.org/contribute.html
Picked up scroll
1 enemies have appeared
Show Time!
100
A: attack, S: ice spell, M:  item menu
Given 52 to enemy 1
Taken 21
fighter 1 has 2979 health left
A: attack, S: ice spell, M:  item menu
Given 52 to enemy 1
received  atkUp
If you want to know some of what is happening within this twisted world, you wil
l want to fetch me the scroll from the town square. You will be rewarded
Glaive Master Hodir is a man who has been cursed for all eternity to test those
who come through here. Those who can best him, in his eyes, may be able to succe
ed where he could not.
```



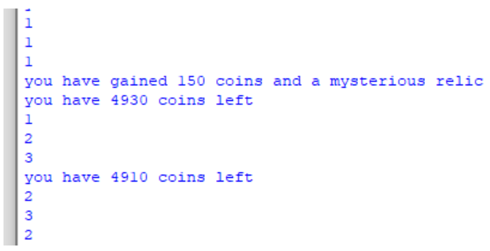
```
import pygame
from random import randint
pygame.init()

coinage = 0
X = 1366
```

Test 23: Does the slot machine pay out if all 3 numbers are the same?

Outcome: The slot machine gives you 150 coins and a key item(on the first win)



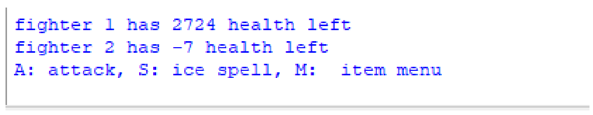
```
def slots(self, MC, wall105, keyItems):
    global coinage
    if coinage >= 20:
        if MC.x + MC.width == wall105.x:
            if MC.y >= wall105.y and (MC.y + MC.height) <= (wall105.y + wall
                coinage -= 20
                print('you have', coinage, 'coins left')
                a = randint(1,3)
                print(a)
                b = randint(1,3)
                print(b)
                c = randint(1,3)
                print(c)
                if a == b and b == c:
                    coinage += 150
                    if 'relic' not in keyItems:
                        print('you have gained 150 coins and a mysterious re
                        keyItems.append('relic')
                    else:
                        print('You have gained 150 coins')
    elif coinage < 20:
        print('You do not have enough coins')
```



```
1
1
1
you have gained 150 coins and a mysterious relic
you have 4930 coins left
1
2
3
you have 4910 coins left
2
3
2
```

Test 24: If a character dies in battle, do they return with 1 health after the battle?

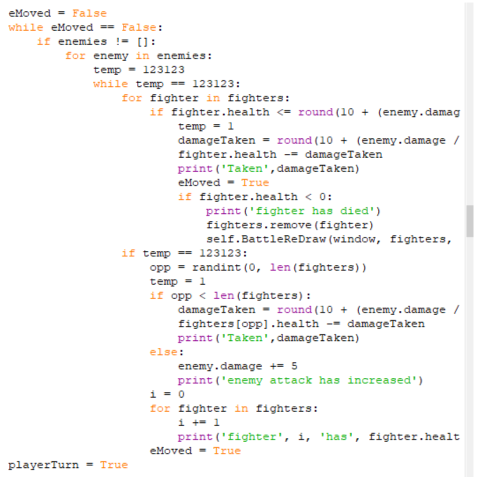
Outcome: They character has negative health but does not die.



```
fighter 1 has 2724 health left
fighter 2 has -7 health left
A: attack, S: ice spell, M:  item menu
```

Remedy:

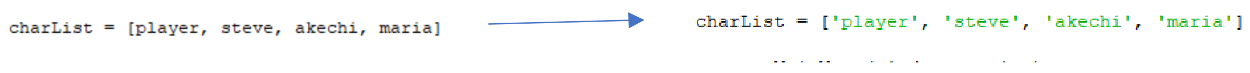
I altered the battle loop to kill the character when they got to below 0 health.



```
eMoved = False
while eMoved == False:
    if enemies != []:
        for enemy in enemies:
            temp = 123123
            while temp == 123123:
                for fighter in fighters:
                    if fighter.health <= round(10 + (enemy.damag
                        temp = 1
                        damageTaken = round(10 + (enemy.damage /
                        fighter.health -= damageTaken
                        print('Taken',damageTaken)
                        eMoved = True
                        if fighter.health < 0:
                            print('fighter has died')
                            fighters.remove(fighter)
                            self.BattleReDraw(window, fighters,
                if temp == 123123:
                    opp = randint(0, len(fighters))
                    temp = 1
                    if opp < len(fighters):
                        damageTaken = round(10 + (enemy.damage /
                        fighters[opp].health -= damageTaken
                        print('Taken',damageTaken)
                    else:
                        enemy.damage += 5
                        print('enemy attack has increased')
                    i = 0
                    for fighter in fighters:
                        i += 1
                        print('fighter', i, 'has', fighter.healt
                    eMoved = True
playerTurn = True
```

Other outcome: The dead character does not return once they have died.

Remedy: I changed the charList from having the objects in the list, to having strings with their names.

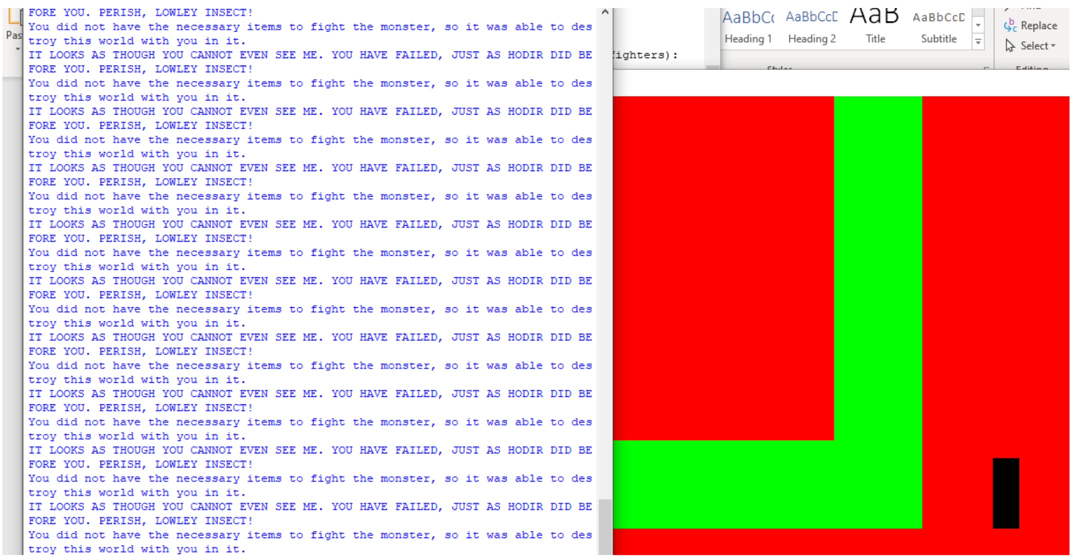


```
charList = [player, steve, akechi, maria]
```

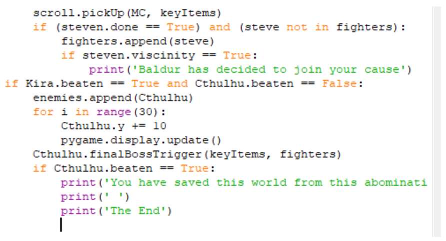
```
charList = ['player', 'steve', 'akechi', 'maria']
```

Test 24: Does the final boss trigger with separate endings based on completion?

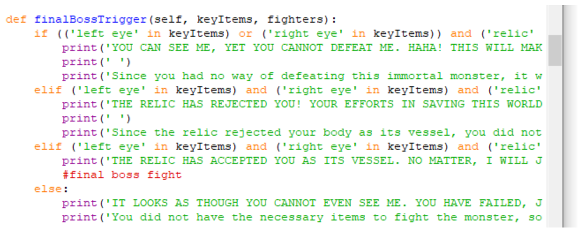
Outcome: The first ending occurs how it should, but the game does not end.



```
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
```

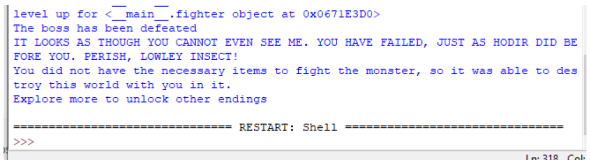


```
    scroll.pickUp(MC, keyItems)
    if (steven.done == True) and (steve not in fighters):
        fighters.append(steve)
        if steven.viscinity == True:
            print('Baldur has decided to join your cause')
if Kira.beaten == True and Cthulhu.beaten == False:
    enemies.append(Cthulhu)
    for i in range(30):
        Cthulhu.y += 10
        pygame.display.update()
    Cthulhu.finalBossTrigger(keyItems, fighters)
    if Cthulhu.beaten == True:
        print('You have saved this world from this abominati
        print(' ')
        print('The End')
```



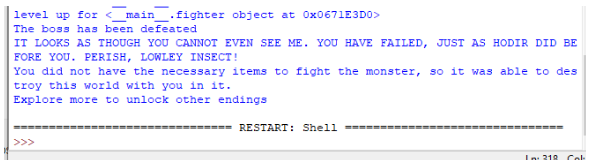
```
def finalBossTrigger(self, keyItems, fighters):
    if (('left eye' in keyItems) or ('right eye' in keyItems)) and ('relic'
        print('YOU CAN SEE ME, YET YOU CANNOT DEFEAT ME. HAHA! THIS WILL MAK
        print(' ')
        print('Since you had no way of defeating this immortal monster, it w
    elif ('left eye' in keyItems) and ('right eye' in keyItems) and ('relic'
        print('THE RELIC HAS REJECTED YOU! YOUR EFFORTS IN SAVING THIS WORLD
        print(' ')
        print('Since the relic rejected your body as its vessel, you did not
    elif ('left eye' in keyItems) and ('right eye' in keyItems) and ('relic'
        print('THE RELIC HAS ACCEPTED YOU AS ITS VESSEL. NO MATTER, I WILL J
        #final boss fight
    else:
        print('IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, J
        print('You did not have the necessary items to fight the monster, so
```

Remedy: I added something to the check, ending the game after the ending is shown.



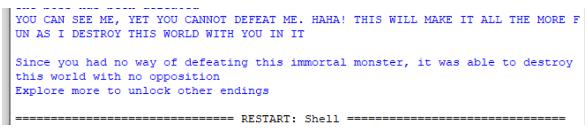
```
level up for <__main__.fighter object at 0x0671E3D0>
The boss has been defeated
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
Explore more to unlock other endings

============================== RESTART: Shell ==============================
>>>
```

Outcome for ending 2: The first ending triggers instead of the second

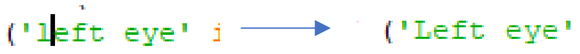


```
level up for <__main__.fighter object at 0x0671E3D0>
The boss has been defeated
IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, JUST AS HODIR DID BE
FORE YOU. PERISH, LOWLEY INSECT!
You did not have the necessary items to fight the monster, so it was able to des
troy this world with you in it.
Explore more to unlock other endings

============================== RESTART: Shell ==============================
>>>
```

Remedy: I altered the item names to be consistent.



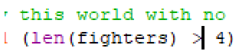
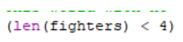
```
YOU CAN SEE ME, YET YOU CANNOT DEFEAT ME. HAHA! THIS WILL MAKE IT ALL THE MORE F
UN AS I DESTROY THIS WORLD WITH YOU IN IT

Since you had no way of defeating this immortal monster, it was able to destroy
this world with no opposition
Explore more to unlock other endings

============================== RESTART: Shell ==============================
```

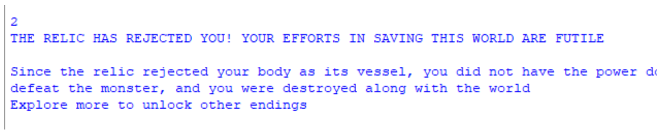


```
('left eye' i  ⟶  ('Left eye'
```

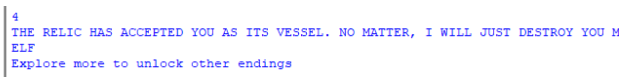
Outcome for 3rd ending: The first ending is played instead

Remedy: I altered the check for the third ending.

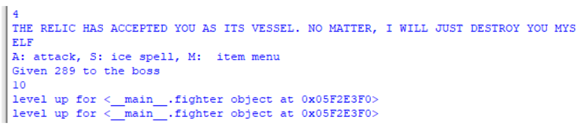
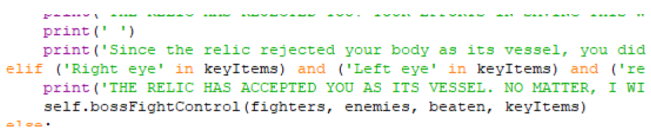

```
' this world with no
l (len(fighters) > 4)
```

```
(len(fighters) < 4)
```


```
2
THE RELIC HAS REJECTED YOU! YOUR EFFORTS IN SAVING THIS WORLD ARE FUTILE

Since the relic rejected your body as its vessel, you did not have the power d
defeat the monster, and you were destroyed along with the world
Explore more to unlock other endings
```

Outcome for true ending: The text shows but the final boss doesn't trigger.

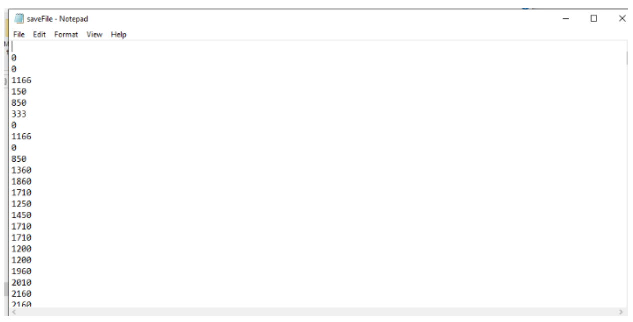

```
4
THE RELIC HAS ACCEPTED YOU AS ITS VESSEL. NO MATTER, I WILL JUST DESTROY YOU M
ELF
Explore more to unlock other endings
```

Remedy: I called the bossFightControl method in the true ending section of the finalBossTrigger method

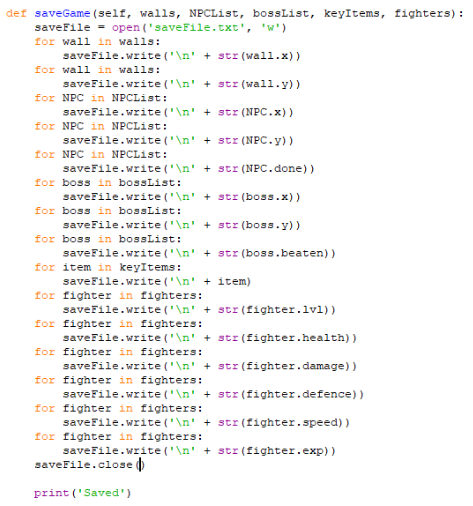

```
4
THE RELIC HAS ACCEPTED YOU AS ITS VESSEL. NO MATTER, I WILL JUST DESTROY YOU MYS
ELF
A: attack, S: ice spell, M:  item menu
Given 289 to the boss
10
level up for <__main__.fighter object at 0x05F2E3F0>
level up for <__main__.fighter object at 0x05F2E3F0>
```

```
        print(' ')
        print('Since the relic rejected your body as its vessel, you did
    elif ('Right eye' in keyItems) and ('Left eye' in keyItems) and ('re
        print('THE RELIC HAS ACCEPTED YOU AS ITS VESSEL. NO MATTER, I WI
        self.bossFightControl(fighters, enemies, beaten, keyItems)
```

Test 25: does the game save variables to a text file?

Outcome: all variable that need to be saved are saved to a text file line-by-line.

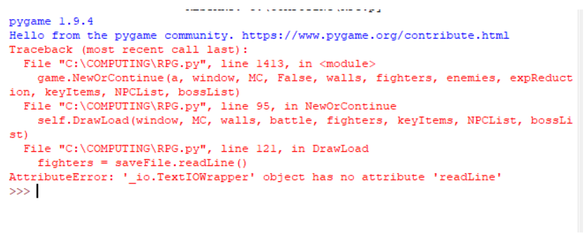


```

0
0
1166
150
850
333
0
1166
0
850
1360
1860
1710
1250
1450
1710
1710
1200
1200
1960
2010
2160
2160
```


```
def saveGame(self, walls, NPCList, bossList, keyItems, fighters):
    saveFile = open('saveFile.txt', 'w')
    for wall in walls:
        saveFile.write('\n' + str(wall.x))
    for wall in walls:
        saveFile.write('\n' + str(wall.y))
    for NPC in NPCList:
        saveFile.write('\n' + str(NPC.x))
    for NPC in NPCList:
        saveFile.write('\n' + str(NPC.y))
    for NPC in NPCList:
        saveFile.write('\n' + str(NPC.done))
    for boss in bossList:
        saveFile.write('\n' + str(boss.x))
    for boss in bossList:
        saveFile.write('\n' + str(boss.y))
    for boss in bossList:
        saveFile.write('\n' + str(boss.beaten))
    for item in keyItems:
        saveFile.write('\n' + item)
    for fighter in fighters:
        saveFile.write('\n' + str(fighter.lvl))
    for fighter in fighters:
        saveFile.write('\n' + str(fighter.health))
    for fighter in fighters:
        saveFile.write('\n' + str(fighter.damage))
    for fighter in fighters:
        saveFile.write('\n' + str(fighter.defence))
    for fighter in fighters:
        saveFile.write('\n' + str(fighter.speed))
    for fighter in fighters:
        saveFile.write('\n' + str(fighter.exp))
    saveFile.close()

    print('Saved')
```

Test 26: If you press Continue game, does it get the variables from the text file and draw it in the same way as you last saved?

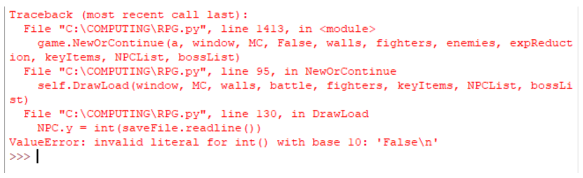
Outcome: An error occurred.



```
pygame 1.9.4
Hello from the pygame community. https://www.pygame.org/contribute.html
Traceback (most recent call last):
  File "C:\COMPUTING\RPG.py", line 1413, in <module>
    game.NewOrContinue(a, window, MC, False, walls, fighters, enemies, expReduct
ion, keyItems, NPCList, bossList)
  File "C:\COMPUTING\RPG.py", line 95, in NewOrContinue
    self.DrawLoad(window, MC, walls, battle, fighters, keyItems, NPCList, bossLi
st)
  File "C:\COMPUTING\RPG.py", line 121, in DrawLoad
    fighters = saveFile.readLine()
AttributeError: '_io.TextIOWrapper' object has no attribute 'readLine'
>>> 
```

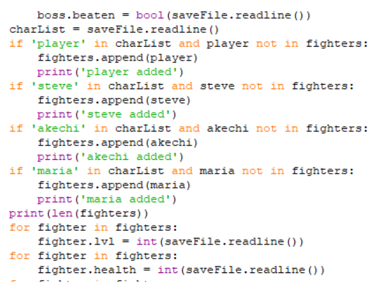
Remedy: I changed 'readLine' to 'readline'

Outcome 2:

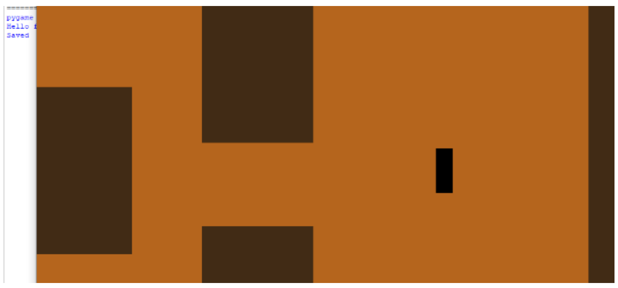


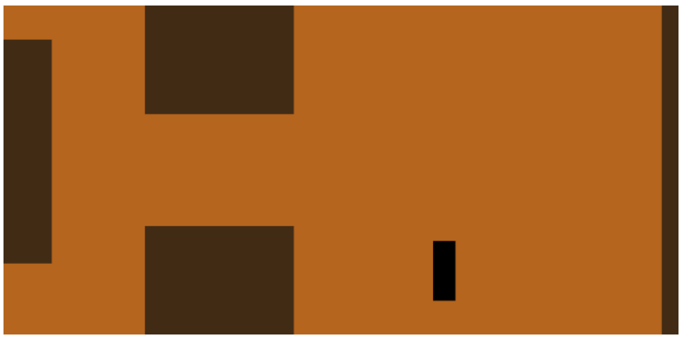
```
Traceback (most recent call last):
  File "C:\COMPUTING\RPG.py", line 1413, in <module>
    game.NewOrContinue(a, window, MC, False, walls, fighters, enemies, expReduct
ion, keyItems, NPCList, bossList)
  File "C:\COMPUTING\RPG.py", line 95, in NewOrContinue
    self.DrawLoad(window, MC, walls, battle, fighters, keyItems, NPCList, bossLi
st)
  File "C:\COMPUTING\RPG.py", line 130, in DrawLoad
    NPC.y = int(saveFile.readline())
ValueError: invalid literal for int() with base 10: 'False\n'
>>> 
```

Remedy: I only added characters to lists if they weren't already in there, so they weren't getting the wrong variables from the text file.



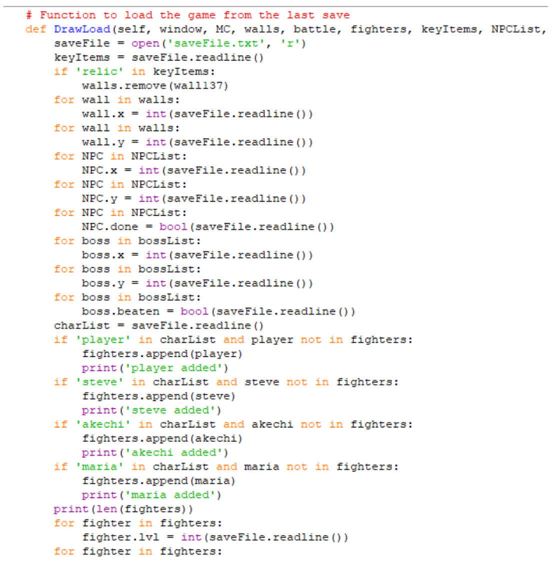
```
    boss.beaten = bool(saveFile.readline())
charList = saveFile.readline()
if 'player' in charList and player not in fighters:
    fighters.append(player)
    print('player added')
if 'steve' in charList and steve not in fighters:
    fighters.append(steve)
    print('steve added')
if 'akechi' in charList and akechi not in fighters:
    fighters.append(akechi)
    print('akechi added')
if 'maria' in charList and maria not in fighters:
    fighters.append(maria)
    print('maria added')
print(len(fighters))
for fighter in fighters:
    fighter.lvl = int(saveFile.readline())
for fighter in fighters:
    fighter.health = int(saveFile.readline())
```



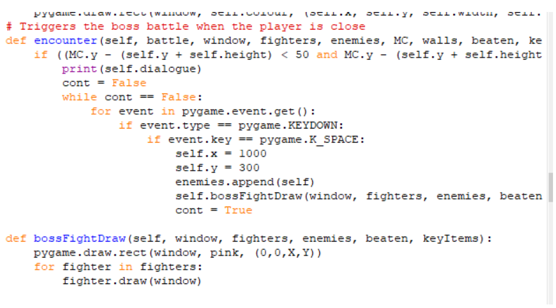


Final Review

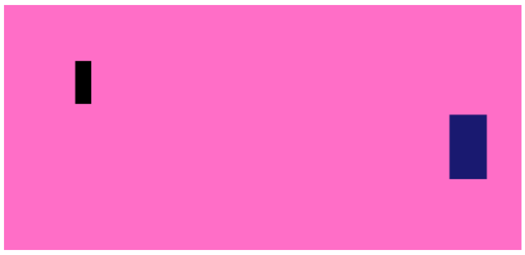
All of the main features of the game now fully function, like boss encounters, the ‘Continue Game’ option and NPC side quests. Enemies can now buff their strength in battle and bosses can buff strength or defence, or they can heal themselves. The final boss now triggers one of four endings, based on the player’s key items list and the number of characters in their party.

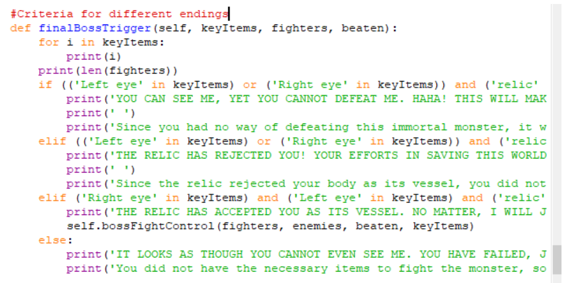


```
# Function to load the game from the last save
def DrawLoad(self, window, MC, walls, battle, fighters, keyItems, NPCList,
    saveFile = open('saveFile.txt', 'r')
    keyItems = saveFile.readline()
    if 'relic' in keyItems:
        walls.remove(wall137)
    for wall in walls:
        wall.x = int(saveFile.readline())
    for wall in walls:
        wall.y = int(saveFile.readline())
    for NPC in NPCList:
        NPC.x = int(saveFile.readline())
    for NPC in NPCList:
        NPC.y = int(saveFile.readline())
    for NPC in NPCList:
        NPC.done = bool(saveFile.readline())
    for boss in bossList:
        boss.x = int(saveFile.readline())
    for boss in bossList:
        boss.y = int(saveFile.readline())
    for boss in bossList:
        boss.beaten = bool(saveFile.readline())
    charList = saveFile.readline()
    if 'player' in charList and player not in fighters:
        fighters.append(player)
        print('player added')
    if 'steve' in charList and steve not in fighters:
        fighters.append(steve)
        print('steve added')
    if 'akechi' in charList and akechi not in fighters:
        fighters.append(akechi)
        print('akechi added')
    if 'maria' in charList and maria not in fighters:
        fighters.append(maria)
        print('maria added')
    print(len(fighters))
    for fighter in fighters:
        fighter.lvl = int(saveFile.readline())
    for fighter in fighters:
```



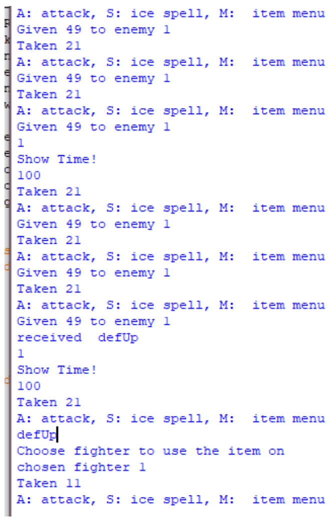
```
    pygame.draw.rect(window, self.colour, (self.x, self.y, self.width, self.
# Triggers the boss battle when the player is close
def encounter(self, battle, window, fighters, enemies, MC, walls, beaten, ke
    if ((MC.y - (self.y + self.height) < 50 and MC.y - (self.y + self.height
        print(self.dialogue)
        cont = False
        while cont == False:
            for event in pygame.event.get():
                if event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_SPACE:
                        self.x = 1000
                        self.y = 300
                        enemies.append(self)
                        self.bossFightDraw(window, fighters, enemies, beaten
                        cont = True

def bossFightDraw(self, window, fighters, enemies, beaten, keyItems):
    pygame.draw.rect(window, pink, (0,0,X,Y))
    for fighter in fighters:
        fighter.draw(window)
```



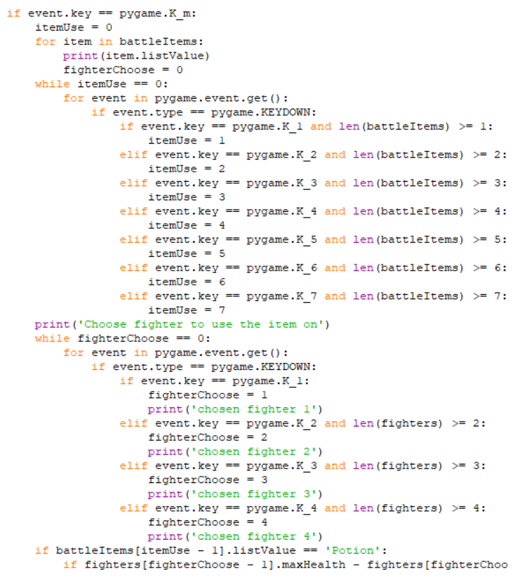


```
#Criteria for different endings
def finalBossTrigger(self, keyItems, fighters, beaten):
    for i in keyItems:
        print(i)
    print(len(fighters))
    if (('Left eye' in keyItems) or ('Right eye' in keyItems)) and ('relic'
        print('YOU CAN SEE ME, YET YOU CANNOT DEFEAT ME. HAHA! THIS WILL MAK
        print(' ')
        print('Since you had no way of defeating this immortal monster, it w
    elif (('Left eye' in keyItems) or ('Right eye' in keyItems)) and ('relic
        print('THE RELIC HAS REJECTED YOU! YOUR EFFORTS IN SAVING THIS WORLD
        print(' ')
        print('Since the relic rejected your body as its vessel, you did not
    elif ('Right eye' in keyItems) and ('Left eye' in keyItems) and ('relic'
        print('THE RELIC HAS ACCEPTED YOU AS ITS VESSEL. NO MATTER, I WILL J
        self.bossFightControl(fighters, enemies, beaten, keyItems)
    else:
        print('IT LOOKS AS THOUGH YOU CANNOT EVEN SEE ME. YOU HAVE FAILED, J
        print('You did not have the necessary items to fight the monster, so
```

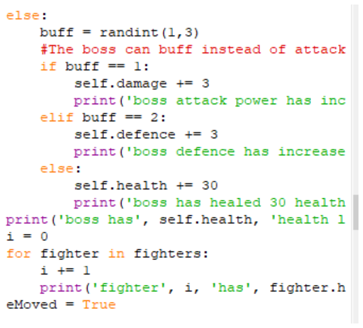
Using items in-battle:



```
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
1
Show Time!
100
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
Taken 21
A: attack, S: ice spell, M:  item menu
Given 49 to enemy 1
received  defUp
1
Show Time!
100
Taken 21
A: attack, S: ice spell, M:  item menu
defUp
Choose fighter to use the item on
chosen fighter 1
Taken 11
A: attack, S: ice spell, M:  item menu
```



```
if event.key == pygame.K_m:
    itemUse = 0
    for item in battleItems:
        print(item.listValue)
        fighterChoose = 0
    while itemUse == 0:
        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_1 and len(battleItems) >= 1:
                    itemUse = 1
                elif event.key == pygame.K_2 and len(battleItems) >= 2:
                    itemUse = 2
                elif event.key == pygame.K_3 and len(battleItems) >= 3:
                    itemUse = 3
                elif event.key == pygame.K_4 and len(battleItems) >= 4:
                    itemUse = 4
                elif event.key == pygame.K_5 and len(battleItems) >= 5:
                    itemUse = 5
                elif event.key == pygame.K_6 and len(battleItems) >= 6:
                    itemUse = 6
                elif event.key == pygame.K_7 and len(battleItems) >= 7:
                    itemUse = 7
    print('Choose fighter to use the item on')
    while fighterChoose == 0:
        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_1:
                    fighterChoose = 1
                    print('chosen fighter 1')
                elif event.key == pygame.K_2 and len(fighters) >= 2:
                    fighterChoose = 2
                    print('chosen fighter 2')
                elif event.key == pygame.K_3 and len(fighters) >= 3:
                    fighterChoose = 3
                    print('chosen fighter 3')
                elif event.key == pygame.K_4 and len(fighters) >= 4:
                    fighterChoose = 4
                    print('chosen fighter 4')
    if battleItems[itemUse - 1].listValue == 'Potion':
        if fighters[fighterChoose - 1].maxHealth - fighters[fighterChoo
```



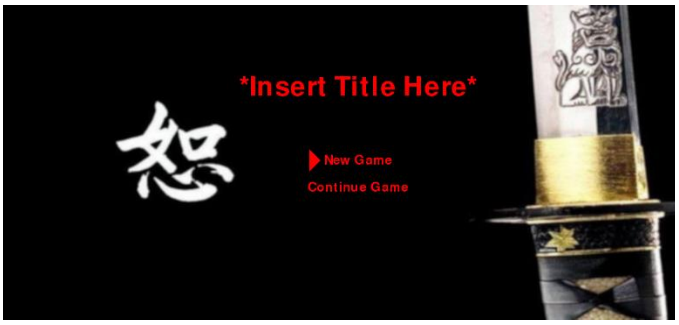
```
else:
    buff = randint(1,3)
    #The boss can buff instead of attack
    if buff == 1:
        self.damage += 3
        print('boss attack power has inc
    elif buff == 2:
        self.defence += 3
        print('boss defence has increase
    else:
        self.health += 30
        print('boss has healed 30 health
print('boss has', self.health, 'health l
i = 0
for fighter in fighters:
    i += 1
    print('fighter', i, 'has', fighter.h
eMoved = True
```

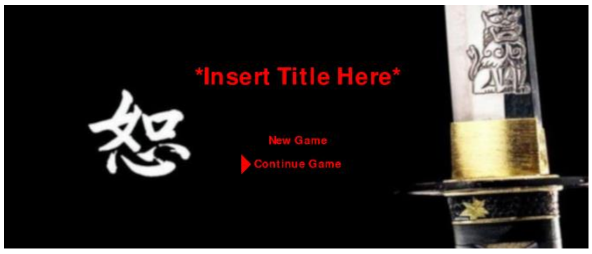
Evaluation

Functional Testing

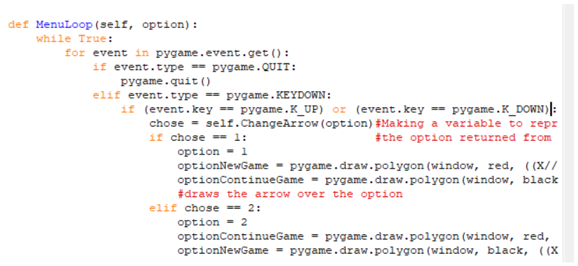
Test 1: Is the main menu drawn and does the cursor move?

Outcome:





The cursor moves when the up or down arrow keys are pressed, and it doesn't move when any other key is pressed as it only checks for these keys for moving the cursor.



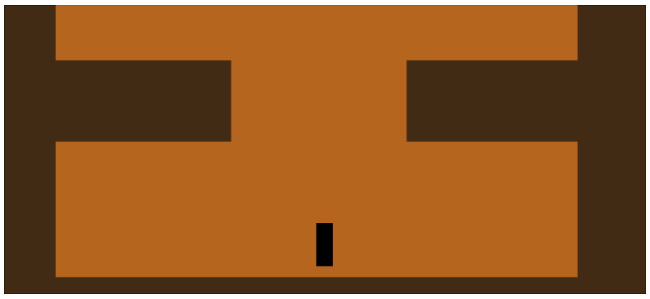
```
def MenuLoop(self, option):
    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
            elif event.type == pygame.KEYDOWN:
                if (event.key == pygame.K_UP) or (event.key == pygame.K_DOWN):
                    chose = self.ChangeArrow(option)#Making a variable to repr
                    if chose == 1:                  #the option returned from
                        option = 1
                        optionNewGame = pygame.draw.polygon(window, red, ((X//
                        optionContinueGame = pygame.draw.polygon(window, black
                        #draws the arrow over the option
                    elif chose == 2:
                        option = 2
                        optionContinueGame = pygame.draw.polygon(window, red,
                        optionNewGame = pygame.draw.polygon(window, black, ((X
```

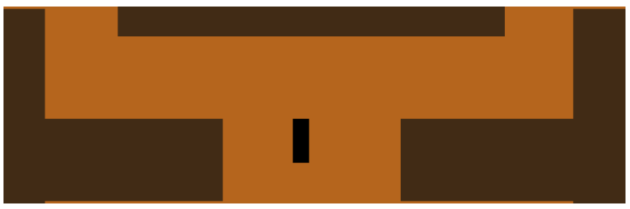
Test 2: when the player chooses an option on the menu, does it go to the correct function?

Outcome: See test 5 in testing for development.

Test 3: Does the player move only if the arrow keys are pressed and do the walls scroll?

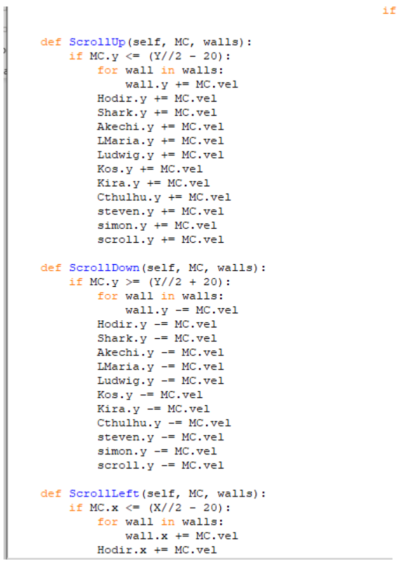
Outcome:



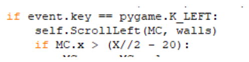


The character moves when the arrow keys are pressed and the walls scroll onscreen so the player doesn't move offscreen. If other keys are pressed, nothing happens as the code only checks for specific keys, so if other keys are pressed, nothing happens. Links to my HIPO chart for movement being controlled by arrow keys.

Scrolling: Moving:



```
def ScrollUp(self, MC, walls):
    if MC.y <= (Y//2 - 20):
        for wall in walls:
            wall.y += MC.vel
        Hodir.y += MC.vel
        Shark.y += MC.vel
        Akechi.y += MC.vel
        LMaria.y += MC.vel
        Ludwig.y += MC.vel
        Kos.y += MC.vel
        Kira.y += MC.vel
        Cthulhu.y += MC.vel
        steven.y += MC.vel
        simon.y += MC.vel
        scroll.y += MC.vel

def ScrollDown(self, MC, walls):
    if MC.y >= (Y//2 + 20):
        for wall in walls:
            wall.y -= MC.vel
        Hodir.y -= MC.vel
        Shark.y -= MC.vel
        Akechi.y -= MC.vel
        LMaria.y -= MC.vel
        Ludwig.y -= MC.vel
        Kos.y -= MC.vel
        Kira.y -= MC.vel
        Cthulhu.y -= MC.vel
        steven.y -= MC.vel
        simon.y -= MC.vel
        scroll.y -= MC.vel

def ScrollLeft(self, MC, walls):
    if MC.x <= (X//2 - 20):
        for wall in walls:
            wall.x += MC.vel
        Hodir.x += MC.vel
```



```
if event.key == pygame.K_LEFT:
    self.ScrollLeft(MC, walls)
    if MC.x > (X//2 - 20):
```

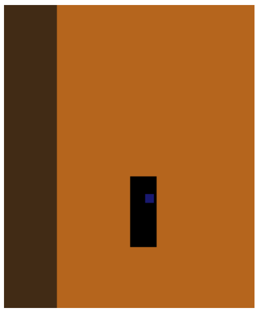
It checks for scrolling before moving the character so, if they have moved too far, only the walls will move, not the player, creating the illusion of movement from the player.

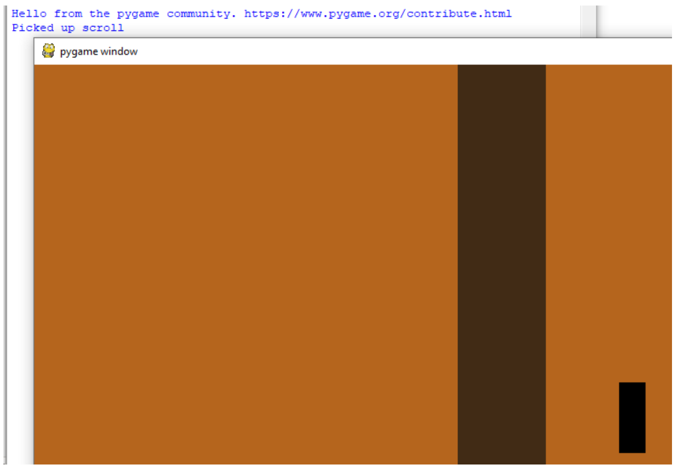
Test 4: Does the collision on the walls work?

Outcome: See test 8 in testing for development. Links to my HIPO chart which says: 'cannot move through solid objects'.

Test 5: Does the player pick up items when space is pressed while standing over one?

Outcome:





```
Hello from the pygame community. https://www.pygame.org/contribute.html
Picked up scroll
```

The scroll is added to the player's key items array for later use in a side quest.

```
def pickUp(self, MC, keyItems):
    if (self.x > MC.x) and ((self.x + self.width) < (MC.x + MC.width)):
        if (self.y > MC.y) and ((self.y + self.height) < (MC.y + MC.height))
            self.picked = True
            keyItems.append(self.listValue)
            print('Picked up', self.listValue)
```

Test 5: Can the player talk to NPCs and complete side quests?

Outcome: See test 20 and 22 in testing for development. Links to my HIPO chart for dialogue from NPCs shown on the python shell when completing a sidequest.

Test 6: Does the player trigger boss fights when they have moved near the overworld sprite of the boss?

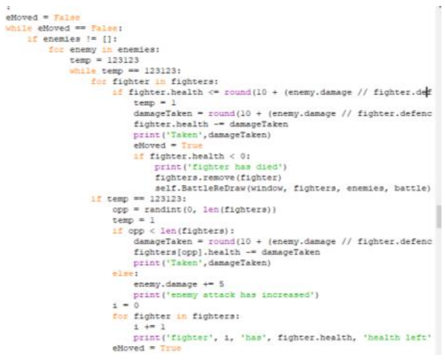
Outcome: See test 17 in testing for development. Links to the scripted events part in my HIPO chart for boss fights and dialogue.

Test 7: Does the battle screen get drawn and allow the player to attack in a battle?

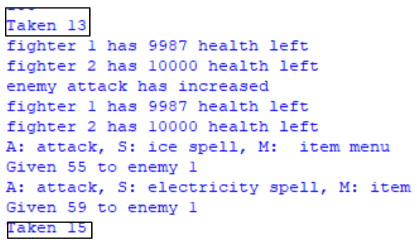
Outcome: See test 9 in testing for development.

Test 8: Do enemies attack the player or buff themselves in battle?

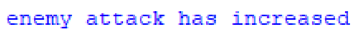
Outcome: When the enemies have a turn, they will randomly attack a party member or buff themselves, unless one of the party members has low health, in which case, the enemy will attack that party member, defeating them. Links to battles in my HIPO chart which shows the different commands for enemies and players.



```
eMoved = False
while eMoved == False:
    if enemies != []:
        for enemy in enemies:
            temp = 123123
            while temp == 123123:
                for fighter in fighters:
                    if fighter.health <= round(10 + (enemy.damage // fighter.def
                        temp = 1
                        damageTaken = round(10 + (enemy.damage // fighter.defenc
                        fighter.health -= damageTaken
                        print('Taken',damageTaken)
                        eMoved = True
                        if fighter.health < 0:
                            print('fighter has died')
                            fighters.remove(fighter)
                            self.BattleReDraw(window, fighters, enemies, battle)
                if temp == 123123:
                    opp = randint(0, len(fighters))
                    temp = 1
                    if opp < len(fighters):
                        damageTaken = round(10 + (enemy.damage // fighter.defenc
                        fighters[opp].health -= damageTaken
                        print('Taken',damageTaken)
                    else:
                        enemy.damage += 5
                        print('enemy attack has increased')
                    i = 0
                    for fighter in fighters:
                        i += 1
                        print('fighter', i, 'has', fighter.health, 'health left'
                    eMoved = True
```



```
Taken 13
fighter 1 has 9987 health left
fighter 2 has 10000 health left
enemy attack has increased
fighter 1 has 9987 health left
fighter 2 has 10000 health left
A: attack, S: ice spell, M:  item menu
Given 55 to enemy 1
A: attack, S: electricity spell, M: item
Given 59 to enemy 1
Taken 15
```



```
enemy attack has increased
```

Test 9: Does the player gain EXP and level up upon gaining enough EXP?

Outcome: See test 13 and 14 in testing for development.

Test 10: Does the correct ending trigger depending on the contents of the player’s key items array and the number of party members?

Outcome: See test 24 in testing for development. Links to my flow chart which shows that there are different endings depending on the items held by the player.

Test 11: Does the game save and return to the correct point when re-loading the game?

Outcome: See tests 25 and 26 in testing for development.

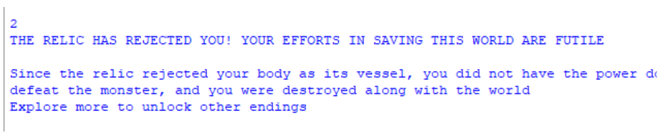
Usability testing

Test 1: Is the movement of the character fluid and quick?

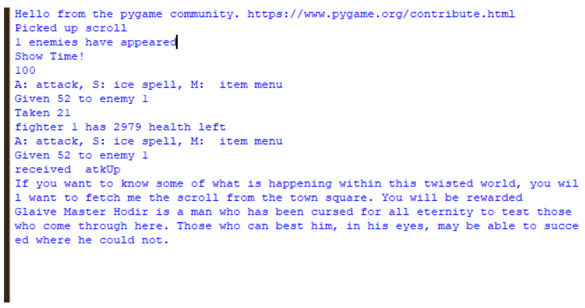
Outcome: The player has to press the arrow keys each time they want to move, so they cannot hold the button down. This is because, if I build the code so the player can hold the button down, the game crashes without an error in the code, making it so the player cannot do anything. In order to make it more fluid, I will need to find a way for the player to be able to hold down the arrow keys to move without crashing the game.

Test 2: Are dialogue and battle controls presented in an easy-to-read way?

Outcome: The dialogue and battle controls are shown on the python shell, making it easy for the player to read as all dialogue is in the same place so there is no need to look for other places to read dialogue.



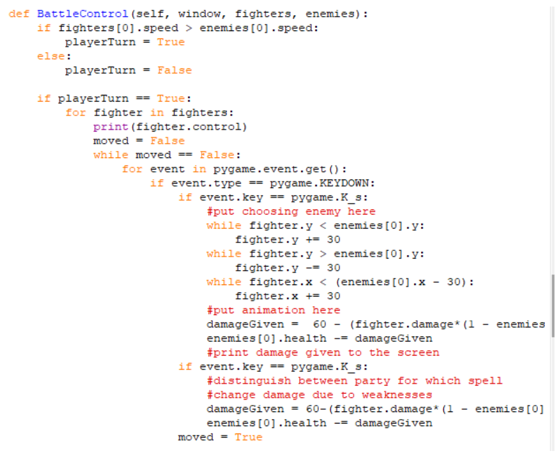
```
2
THE RELIC HAS REJECTED YOU! YOUR EFFORTS IN SAVING THIS WORLD ARE FUTILE

Since the relic rejected your body as its vessel, you did not have the power d
defeat the monster, and you were destroyed along with the world
Explore more to unlock other endings
```



```
Hello from the pygame community. https://www.pygame.org/contribute.html
Picked up scroll
1 enemies have appeared
Show Time!
100
A: attack, S: ice spell, M:  item menu
Given 52 to enemy 1
Taken 21
fighter 1 has 2979 health left
A: attack, S: ice spell, M:  item menu
Given 52 to enemy 1
received  atkUp
If you want to know some of what is happening within this twisted world, you wil
l want to fetch me the scroll from the town square. You will be rewarded
Glaive Master Hodir is a man who has been cursed for all eternity to test those
who come through here. Those who can best him, in his eyes, may be able to succe
ed where he could not.
```

Test 3: Are the battle controls easy to understand and use?

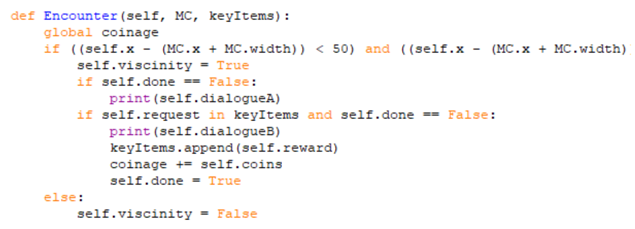
Outcome: The controls are easy to use as they only require a single key press to choose the attack, and another press to choose the enemy to attack (only if there are multiple enemies onscreen). This means there is no requirement for a tutorial as all of the controls are very intuitive and the minimal controls are shown onscreen for if the player gets confused or forgets the controls.



```python
def BattleControl(self, window, fighters, enemies):
    if fighters[0].speed > enemies[0].speed:
        playerTurn = True
    else:
        playerTurn = False

    if playerTurn == True:
        for fighter in fighters:
            print(fighter.control)
            moved = False
            while moved == False:
                for event in pygame.event.get():
                    if event.type == pygame.KEYDOWN:
                        if event.key == pygame.K_s:
                            #put choosing enemy here
                            while fighter.y < enemies[0].y:
                                fighter.y += 30
                            while fighter.y > enemies[0].y:
                                fighter.y -= 30
                            while fighter.x < (enemies[0].x - 30):
                                fighter.x += 30
                            #put animation here
                            damageGiven =  60 - (fighter.damage*(1 - enemies
                            enemies[0].health -= damageGiven
                            #print damage given to the screen
                        if event.key == pygame.K_s:
                            #distinguish between party for which spell
                            #change damage due to weaknesses
                            damageGiven = 60-(fighter.damage*(1 - enemies[0]
                            enemies[0].health -= damageGiven
                        moved = True
```

Test 4: Are NPC sidequests easily accessed?

Outcome: You can only talk to an NPC if you are on their left side and close to them, but once you hae talked to them, it is easy to know what you need to do for the sidequest to complete it.



```
def Encounter(self, MC, keyItems):
    global coinage
    if ((self.x - (MC.x + MC.width)) < 50) and ((self.x - (MC.x + MC.width)
        self.viscinity = True
        if self.done == False:
            print(self.dialogueA)
        if self.request in keyItems and self.done == False:
            print(self.dialogueB)
            keyItems.append(self.reward)
            coinage += self.coins
            self.done = True
    else:
        self.viscinity = False
```

Stakeholder Testing

How easy was traversal?

"The traversal was a little annoying since you had to press a key every time you want to move a step, so I would want to be able to hold a key down to move continuously."

How did you find switching from the game to the shell for dialogue?

"Given the relative lack of details in the game due to it being block-based, switching to the shell was not an issue as you could switch between them quickly since you can move the game window slightly to the side and can see part of the shell before clicking on it."

How easy were the battle controls to follow?

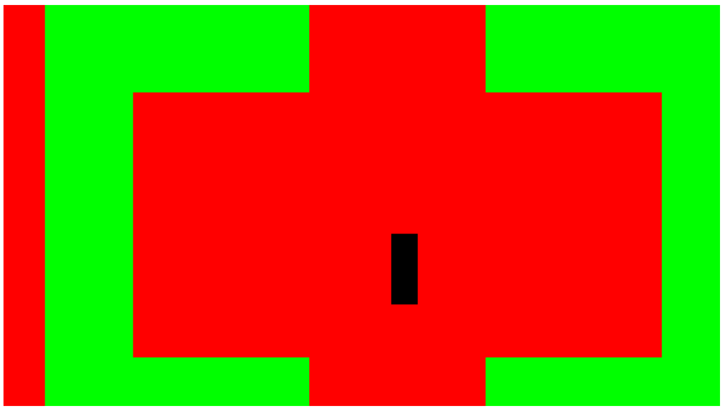
"Since the controls were given to you onscreen and are all a single key press, it was very easy to control the characters in-battle."

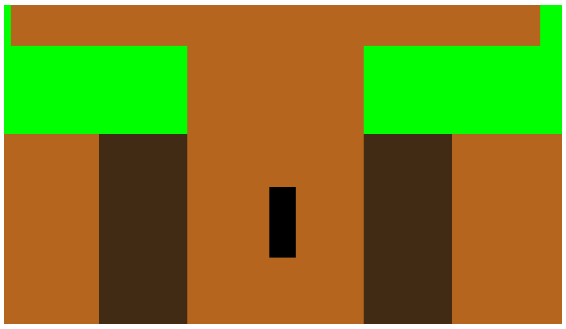
How easy was it to access sidequests?

"They were easy to access, but it is annoying that you can only talk to them when standing on their left. Later, it should be altered so that you can talk to them from any side"

Success Criteria evaluation from analysis

1. **Create all character and background sprites to be implemented into the game for the character to traverse the world while always being able to see where they can go. Each sprite for characters will be unique, and each area will be unique:** This criteria was not well met since, due to time constraints, I was unable to create sprites and animations for any characters or backgrounds. However, each character is a different colour from any other, and the areas are distinct as they have different background colours, walls and they have different side quests. For example, the first area has a fetch quest, the second area has a slot machine and the third area has a mini boss.





2. **Create a battle system to be used on the same window with random encounters:** This criteria was mostly met as each step the player takes has a random chance to trigger a battle, with a random number of 1 to 3 enemies (see test 7), the player can amass up to 4 party members throughout the game to aid them in battle. Also, players and enemies are able to buff themselves during the battle. However, the enemies do not have weaknesses or resistances to any spells.

3. **There will be optional content to reward the player for exploring well, like an optional area based on Bloodborne with a boss fight and good opportunities to gain experience and items:** This criteria was well met since the Bloodborne area is only required for the true ending and, to access it, the player must win the slots and find a path in the second area and it has 3 boss fights and a mini boss, with items to contribute to the true ending.(See test 23 in testing for development)

4. **Create multiple endings based on whether or not the player gained the necessary items throughout their playthrough to encounter the true final boss:** This criteria was well met as there are 4 endings based on how many of the key items the player received during their playthrough, and how many characters they have in their party. (See test 10)

5. **Create 2 sidequests in each area for the player to get invested into, if they want a break from the main story to do new things with interesting characters, fleshing out the world more and making each area have more life to it:** This criteria was not fully met, as there is only one side quest in each area as opposed to the two mentioned in the success criteria. However, these side quests are varied as one is a fetch quest, one is a slot machine and the last is a mini boss.

6. **Create a NG+ feature to add to the replayability of the game, making enemy and boss encounters harder, but the player keeps the equipment that they built up over the course**

of the previous playthrough. It will also have new bosses to encounter and 2 extra sidequests to incentivise the player to do another playthrough, keeping the player interested in the game for longer: This criteria was not met as, once the game has been completed, it is done so the player will only be able to go back to their last save and do side quests to get other endings if they haven't already.

7. **Create a save system that stores the state of all variables into a separate text file to be called to when the program is re-run to save the state of those variables back into the program, so the player can start exactly back from when they saved, unless they choose to start a new game, in which case, the values called will be the default values for each variable:** This criteria was very well met as, at any point in the game, they player is able to save the game to a text file and, if the game is reloaded and the player picks 'Continue Game', they will return to exactly the same place as they saved and, if they pick 'New Game', the player starts at the beginning. (See test 11)

8. **There will be 4 different areas for the player to explore throughout the story, 1 being optional, with 2 exclusive enemies for each area, and some that appear in multiple areas:** This criteria was somewhat met as there are 4 areas with one being optional and one to house the final boss, but there is only one enemy type in the game.

9. **The game will have a balanced difficulty, so it won't be too easy to plough through, but it won't be so hard that it is impossible to get through without grinding levels for hours:** This criteria was well met as the boss fights require strategy and good item usage, and there is no necessity to grind levels for a long time to be able to beat them. If you grow a few levels throughout the area, you should be able to beat each boss.

10. **The program needs to be efficient, taking up as little space as possible, with no load times as the world will be fully interconnected to keep the player as immersed as possible:** This criteria was well met as the full program is object-oriented, with reused functions as methods inside the classes, making it efficient, and the only load time is to load the title screen. Everything else, from loading the backgrounds to battles, is seamless.

Limitations and future development

The most important change for future for me would be making sprites and animations to make the game more visually appealing to hook new players into playing it. As it is currently, the game is not very visually appealing as it is only made up of coloured blocks, so adding unique sprites and animations would make it a lot easier for the player to get invested. I would also add in NG+ to add more replayability to the game (Shown in my stakeholder questionnaire) and allow the player to get more endings without reloading a previous save. It would also add extra difficulty to the game and a programmer would not find it hard to create new sidequests and potentially a new ending since everything is object-oriented so they could just add new objects for NPCs and items. They would also find it relatively easy to add new enemy types as the enemies are a class so they could make objects for the enemies and add them to the existing enemy list.

To add further enjoyment to the game, in the future, I will add more enemy types into the game with harder enemies in later areas. It won't be too hard to implement since the enemies are their own class so I can just make more instances of the class with higher stats for later areas.

The only criteria that a programmer may need some time to meet is adding enemy resistances and weaknesses as they would need to add a new set of variables and alter the battles to either increase or decrease damage taken by the enemies based on the spell they were hit by.

Since the only image file is for the title screen, as long as another computer has Pygame installed, another person is able to run this game quite easily on their computer, as long as they download the source code and the image file for the title screen.

Usability Evaluation

According to my stakeholder, the movement in the game was annoying due to having to press the arrow key to move each step. In the future, I would need to find a way to allow the player to hold down a key to move continuously without causing the game to crash to make it easier for the player to play the game without getting frustrated.

It would also make it a better experience for the player if all the dialogue and damage values in battle were displayed on the game window, so the player doesn't need to keep switching from the game window to the shell.