This exemplar Unit 3 computing project was produced by a Year 13 student

Exemplar Project

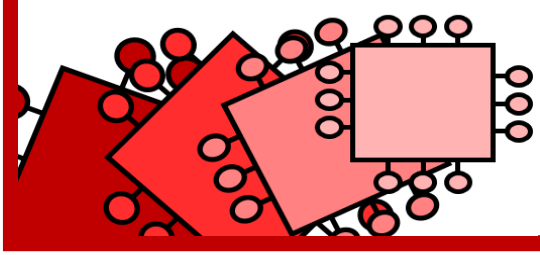
The project was marked at 68/70.

The mark was not adjusted during moderation.

Along with this document please see the following two files:

- Candidate 3 - (68 out of 70) - Additional comments.pdf
- Candidate 3 - (68 out of 70) - Mark Grid.pdf

These files show you, in detail how the best fit marking approach was applied to the actual marking criteria and provides an example of the additional commentary which was sent to the moderator to justify the teachers marks.



OCR GCE A

COMPUTER SCIENCE

PROJECT

H446-03

Name: [redacted]

Candidate Number: [redacted]

Archway School: [redacted]

Title of Project: Vampire Hunter

Candidate Name: [redacted] Candidate Number: [redacted]

H446-03 – PROJECT CONTENTS

TABLE OF CONTENTS

ANALYSIS

AN OUTLINE OF THE PROBLEM

For my project I am planning to create a game that is similar to the game 'Nintendo: Duck Hunt'. This is a retro styled shooting gallery where you are given moving targets to shoot which will give you a certain amount of points when shot. There may be a two player version. Furthermore, there are also moving objects that will deduct life/points if shot In this version of the game depending on the amount of targets you shot the game will take you to the next level/round or present a game over message. The amount of points collected will be added to the users score.

For this to work, the game will require: a score box to show the users score; an indicator for the moving targets being hit; a box to show the user how much life they have; a moving target that looks different from the others to remove points/life; a game over screen; a next level/round indicator for the user or a menu to select different game types.

STAKEHOLDERS

The nature of the graphics will most likely be retro meaning it will not be associated with real life and there won't be any speaking in the game allowing the target audiences ages to drop as there no language complications. The game will mainly be played using the mouse on a PC making the game easy to play. However, as it is a PC game it would mean you would have to take time out of your day to play it, and less of a game you can play on the go. The game can be easily adapted to be playable on a smart phone by using touch screen instead of clicking with the mouse to shoot the targets opening up the game to a wider range of target audience.
Due to the previous factors the game should be designed in a way that allows the target audiences ages to range between 3 and 18.
I have selected a user to personify my target audience, Ben Newton. He is a seventeen-year-old student in my computing class who has an interest in gaming and has also played the original duck hunt game so will know what to expect from the type of game being created. I will also have regular contact with Ben Newton as I share my computing class with him.

HOW THE PROBLEM CAN BE SOLVED BY COMPUTATIONAL METHODS

This problem is well suited for a computer as it can be solved using computational methods. This is due to how a game can be an easily abstracted from reality and how it can have multiple conditions that can be set or the fact that the game has simple algorithms that can be created for a game scenario. I have given the following examples to show how the computer is suited to be solved by a computer program due to the computational methods.

THINKING ABSTRACTLY

I need to decide what to include in the game/ what is necessary and what isn't. This is why abstraction is perfect for this.

Abstraction for 'Vampire hunter':

- Remove all 3 dimensions and restrict the game to 2 dimensions
- Add retro style
- Remove all unnecessary sounds and only have simplified sound effects to indicate certain things to the user
- Add a bar along the bottom of the screen when playing to help give the player an indication of how well they are doing
- Add a menu to give the user the ability too perhaps changes game modes or change some options.

THINKING AHEAD

This is a way of looking ahead and knowing what you want. It allows you to plan a bit giving the solution of your problem a bit of structure.

- I would plan to use monkey-X as this is good software to write games with
- Inputs for the game will most likely be the use of a mouse to play the game/navigate through the game and different buttons to activate e.g. game states like the 'pause' state
- The outputs for the game will consist of sound effects and various animations for the game
- Score system should be included to give the user something to aim towards e.g. reach a new high score

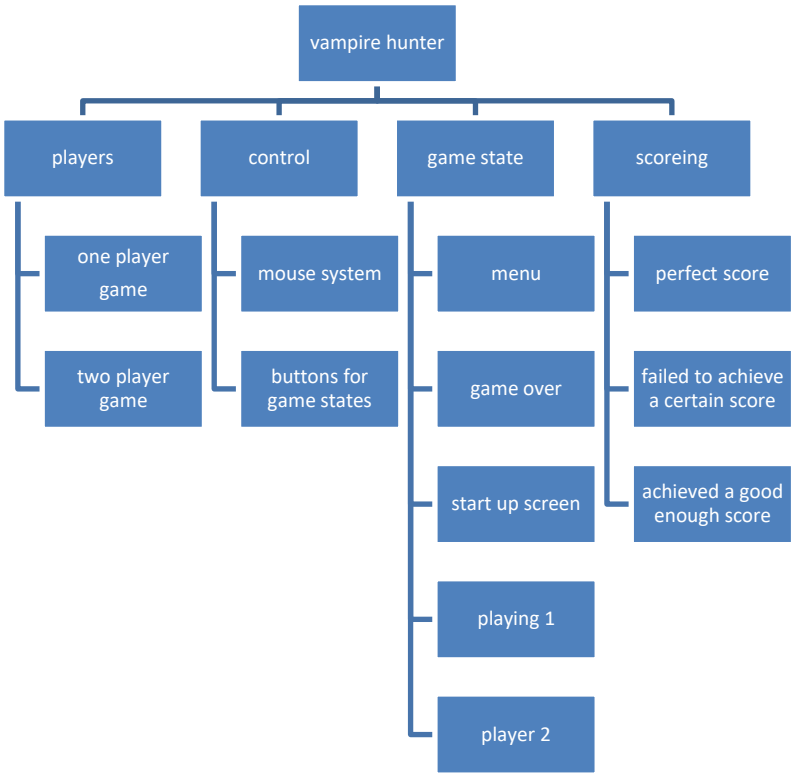
THINKING PROCEDURALLY

This is similar to the thinking ahead as this gives the problem structure making it easier to work with as now you can work at the problem bit by bit making the whole process more efficient.

Procedurally

Candidate Name: [redacted] Candidate Number: [redacted]

Breaking down of the problem



The game has been broken down into four main problems, by thinking procedurally, that should to be solved in order of the game to have basic functionality.

Players

This will determine how the game will be played and if there is a two-player version of the game. If the game is one player there will only one person aiming to shoot the moving targets. If the game is two player both people players could be trying to shoot the moving target or one person could be aiming and the other controlling the moving target.

Controls

These are necessary to interact with the game. The player should be able to use the mouse to click on things in the game to interact and a few buttons to allow the game state to change, e.g. playing to menu by pressing the key ‘m’.

Game states

These game states will create an illusion of depth for the game as it gives the game some structure. There will be a start-up screen to introduce the game; this will then switch to the menu game state this will then allow the user to choose between the ‘player 1’ game sate and the ‘player 2’ game state. Furthermore if certain conditions are meet the game state will change to game over.

Score

This will give the game some sense of achievement. Different amounts in score will trigger different events e.g. if it is perfect code will run to reward the player; if the score per round is too low a game over state will be triggered or the player has just achieved the right score to go to the next round.

THINKING LOGICALLY

This helps me think about what way I could solve the logical aspect of my problem.

When menu button (e.g. 'm') is selected the game will switch to the menu game state. If a target is hit a branch will be activated meaning points will be add to the players score board. As the game is running there will be a constant iteration running, checking certain conditions which will cause a branch changing the game state to the 'game over' game state. There will also be a constant iteration being run to check when a target is hit.

THINKING CONCURRENTLY

This is useful for thinking about how to make my solution more efficient when the solution is in use. The game will be updating the score and drawing the image to screen at the same time.

CONCLUSION

The previous examples have demonstrated that the features of my problem can be solved via computational methods making it suited for a computer program. These methods are useful for giving the problem some structure so it can me more easily solved hence making the problem suited to a computer program.

If the problem is successfully solved using a computer program the stakeholders will be able to play a game where they shoot moving targets, that appear to be flying, giving them a certain number of points. This is aimed to be a form of entertainment for the stakeholders.

RESEARCH

INTERVIEW WITH BEN NEWTON DO GET AN IDEA OF INITIAL REQUIREMENTS

INTERVIEW: 18/09/17 PLAN

This interview will be constructed to find the basic requirements form the user. This will give direction to what type of game will be created.

The main three headings for this are: what makes a good game; is a two-player mode wanted and what type of visuals are wanted for the game.

I have constructed a range of smaller questions under these headings to hit every aspect of the main questions so to make the requirements as specific as possible.

Question layout:

What makes a game good?

- Do sound effects affect your gaming experience?
- Do you think visuals effect gaming experience?
- Should a score system be included in the game?

- How realistic should the game be, taking the target audience into consideration?
- For the 'vampire hunter game', how many moving targets do you think there should be per-round?
- For the 'vampire hunter game', how would you want to interact with the game?

Is a two-player mode wanted?

- Do two player versions of a game make the original game better or worse and why?
- What are your thoughts on competitive games?
- What type of two player game would you want for the 'vampire hunter game'?

What sort of visuals would you want for the game?

- If any realistic features should be included for the 'vampire hunter game' what should they be?
- What type of game style would you like?
- What sort of bat theme would you want for the background of the game?
- Would you want the back grounds to change?
- Would you want any animations for the moving targets?

INTERVIEW SCRIPT: 18/09/17

WHAT MAKES A GOOD GAME?

Do sound effects affect your gaming experience?
Yes, because if its synchronous sound it amplifies the visual effects.
Do you think visuals effect gaming experience?
Yes, depending on the genre of game there will be different visuals. A game rated 16 will have an increased amount of gore compared to a game rated 3. Even though it doesn't affect the gameplay.
Should a score system be included in the game?
Yes, it gives you a goal and tells you how well you are doing in the game
How realistic should the game be, taking the target audience into consideration?
Due to the genre of game the graphics does not need to be realistic so the target audience know that you are not killing real bats. Again, it does not affect the gameplay.
For the 'vampire hunter game', how many moving targets do you think there should be per-round?
Per round I would say 5-10 per round.
For the 'vampire hunter game', how would you want to interact with the game?
Using a mouse to aim and click to shoot the bats.

IS A TWO PLAYER MODE WANTED?

Do two player versions of a game make the original game better or worse and why?
You can compete against your friend in real life to get the best score.
What are your thoughts on competitive games?
Competitive games require more time to set up and play. A one player game can be loaded quickly and played to waste time.
What type of two player game would you want for the 'vampire hunter game'?

One player can control the bat and try to get away from the other player trying to shoot them.

WHAT SORT OF VISUALS WOULD YOU WANT FOR THE GAME?

If any realistic features should be included for the 'vampire hunter game' what should they be?
When the bats appear, they have to go back towards the ground like gravity. Reloading takes time and is not instant.
What type of game style would you like?
I would like the game style to be in a retro pixel are format as it stops the game from becoming to realistic
Would you want the back grounds to change?
Different levels, change the background.
Would you want any animations for the moving targets?
Yes, as it makes the game more exciting e.g. when they get shot their eyes change to crosses and their tongues stick out.

A REVIEW OF THE INTERVIEW: 18/09/17 ANSWERS

This interview was created to see what things the stakeholders wanted from a game to make it enjoyable. The stakeholder and I ended up concluding that visuals are important when just looking at game play but in general it will make the game more unique.

For the visuals and sounds, we decided to include small animations along with some sound effects. Furthermore, all the art will be in the format of pixel art. The game will be realistic enough so that the moving targets appear to be following the laws of physic but remove gore.

For the game play, we decided to include a level system that will link in with a score system to give the player a sense achievement in the game. We also decided to include a two-player mode to make the game more appealing to a wider range of people as there is more content being given and they can play the game with their friends making the game more fun as it adds a competitive aspect. Finally, it was decided to use the mouse as the control.

RESEARCH IN TO NINTENDO: DUCK HUNT

'Nintendo: Duck Hunt' is a shooter where the users must shoot moving targets that move across the screen before they disappear. The game is a from first person perspective and requires a 'Zapper light gun' which is used to aim at the screen and shoot the moving targets.

The game has three game modes. For both game mode A and B the moving targets are flying ducks that have flown out from long grass; there is also a tree on the left of the screen which the ducks can fly behind. For game mode A, only one duck AI flies up from the grass and can be controlled by a second user. However, in game mode B two duck AI fly up from the grass but can't be controlled by a second user. The third game mode there are just clay pigeons that fly into the distance from the user's perspective.

Each round has a total of ten moving targets to shoot. The users must shoot a certain number of targets to proceed to the next round. The user will get points added to their score as the targets are hit and will receive bonus points for shooting all ten targets in one round.

Each time the targets appear, at any given time, the user is given three shots to shoot the targets; if the user runs out of bullets and doesn't shoot all the targets or if one of the targets disappear the user is presented with a game over message. As the user proceeds to higher round the difficulty increase; this done by the speed of the moving targets increasing and the minimum number of targets to shoot will increase.

Main features of the game include:

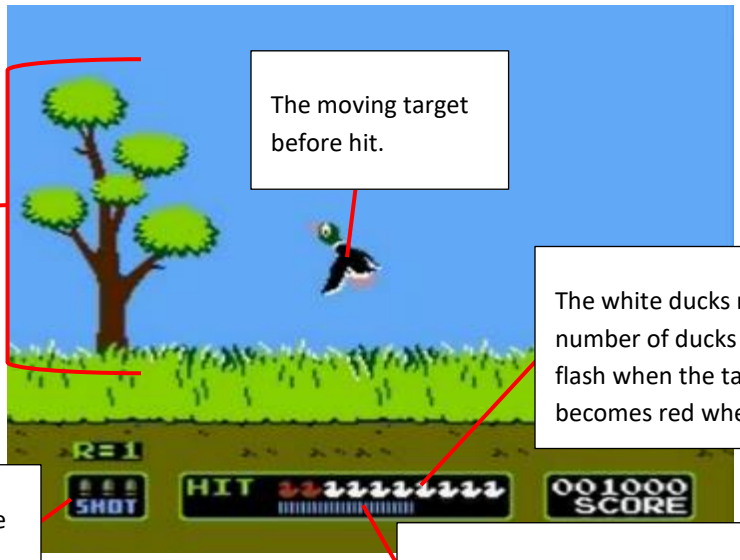
<u>Game mode A and B</u>

A dog animation is shown at the start of each round. The dog also barks.

Each time a new round appears the round is shown to the user in a black box along with some intro music that stops when the first duck is released.

The round the user is displayed when playing.

The score of the user.



A tree which the ducks can fly behind.

The moving target before hit.

The white ducks represents the total number of ducks per round. The white ducks flash when the target appears and the duck becomes red when successfully hit.

The amount of shots the user has. Furthermore, the 'Shot' text flashes when empty.

This represents the minimum number of ducks you have to hit to pass the round.

Candidate Name: [redacted] Candidate Number: [redacted]

This is the animation that is show when the users successfully hits the duck. When this happens a gunshot sound is played.

After the duck has been shot the number of points gained is briefly displayed above the duck and the duck falls with a falling animation. The falling animation is played with a whistling sound. The points are then added to the users score.

When the ducks are in the air they make a flapping sound and also quack from time to time.

When all the ducks are hit after each set there is a dog animation. For when two ducks are hit a dog appears with two ducks as shown on the left. If one duck is hit then the dog appears with one duck. Both are played with a short clip of victory music.

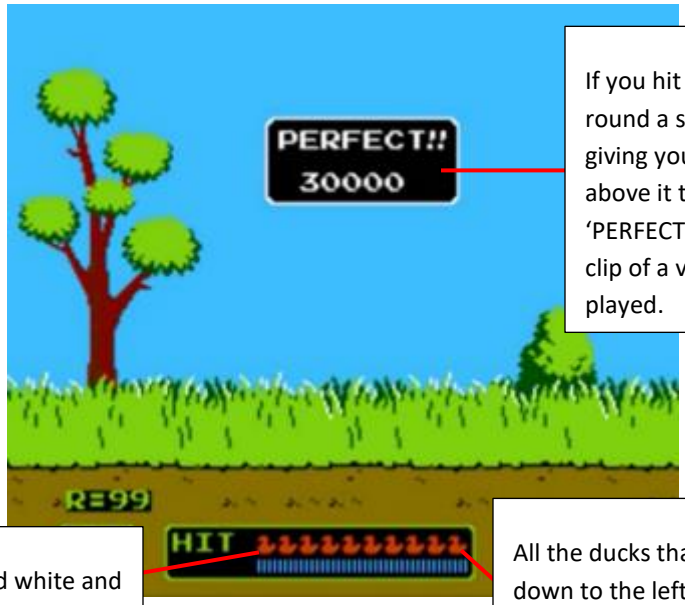
If you fail to shoot the duck in the given time an animation of a duck flying away is shown along with the sky briefly flashing from blue to pink.

A black box briefly appears with the text 'FLY AWAY'.

The shot box also flashes by appearing and becoming black.

If you run out of shots and don't hit all the ducks or all the ducks disappear a laughing dog animation appears with a sound clip of some sort of laughter.

If you fail to hit the minimum required number of ducks a game over sign appears with some game over music and then the game goes back to the menu.



If you hit all the ducks in one round a sign briefly appears giving your score and text above it that says 'PERFECT!!'. A short sound clip of a victory is also played.

The row of ducks flash red and white and then stays red until the next round if all the ducks are hit.

All the ducks that were hit slide down to the left hand side of the row of ducks to see if the user has shot enough ducks to pass to the next round.

Duck hunt game menu

The games title, 'Duck Hunt'.

There is menu music that is played on loop during the games menu state.

The selection icon.

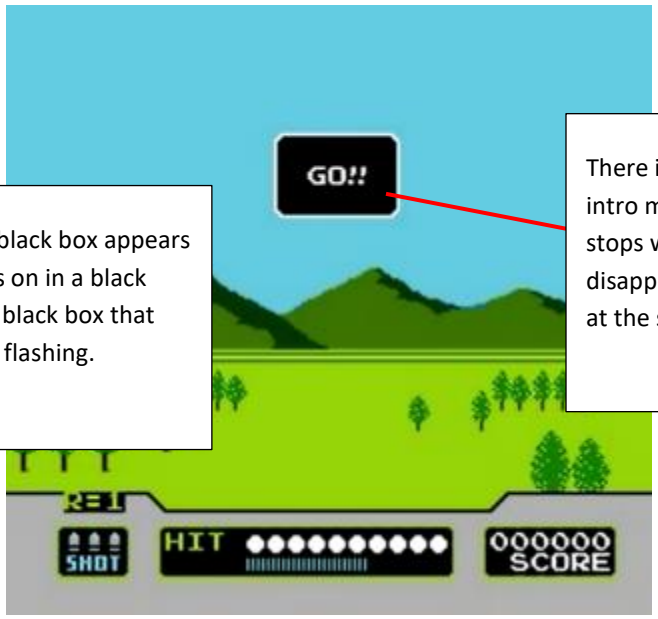
The three available game modes to play.

The games top score.

This shows the game is copyrighted by Nintendo.

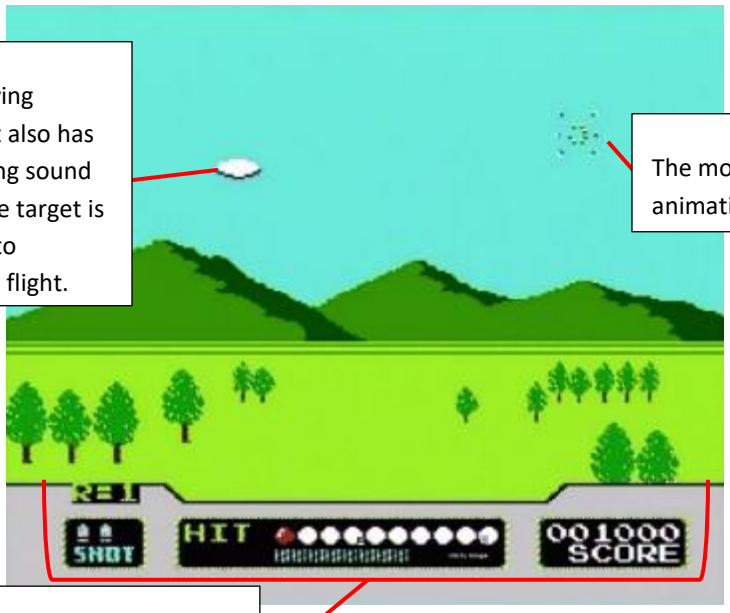
Game mode C

This game still belongs to ‘Duck Hunt’. However it lacks many features of the main game. This game type is less violent then the original as it doesn’t show the ducks getting killed and then being collected by a dog; it only shows the clay pigeons exploding.



When the round starts a black box appears with the round the user is on in a black box. This is followed by a black box that has the text ‘GO!!’ that is flashing.

There is a short piece of intro music played but stops when the ‘GO!!’ box disappears. This is played at the start of every round.



The moving target. It also has a whizzing sound when the target is moving to simulate flight.

The moving target animation when hit.

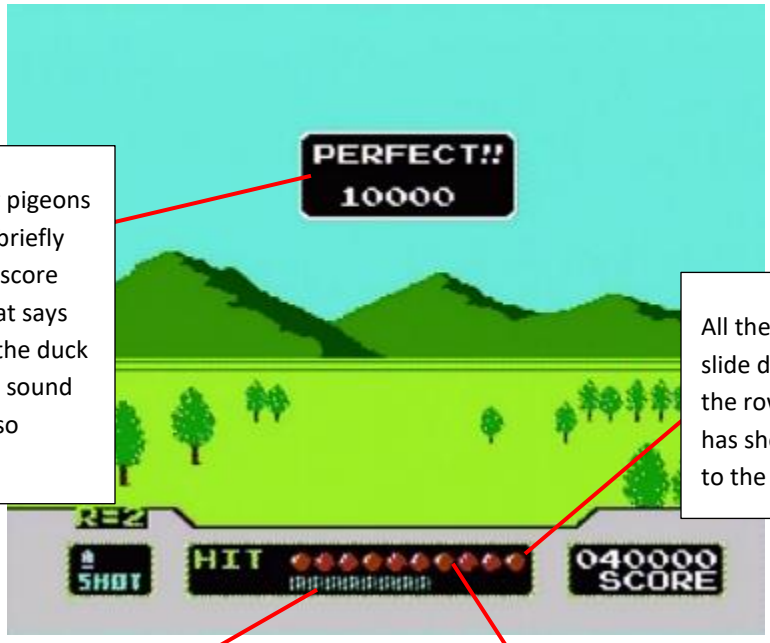
The round display; shot display; hit display and score display is identical to game mode A/B but has a different background and instead of a row of ducks it has a row of circles representing clay pigeons.

There is a sound similar to an explosion for when the clay pigeon is hit.

Candidate Name: [redacted] Candidate Number: [redacted]

There is a sound for launch of the clay pigeon and a sound for the flying of the clay pigeon.

After the clay pigeon's animation for being hit is played the score is then briefly displayed as white text where the target was hit. Then the points are added to the score.



If you hit all the clay pigeons in one round a sign briefly appears giving your score and text above it that says 'PERFECT!!'. Like in the duck hunt version a short sound clip of a victory is also played.

All the clay pigeons that were hit slide down to the left hand side of the row of ducks to see if the user has shot enough clay pigeons to pass to the next round.

This represents the minimum number of clay pigeons you have to hit to pass the round.

Like game mode A/B when all the targets are all hit in the same round the row clay pigeons flash red and white and then stays red until the next round.

If you run out of shots and don't hit all the ducks or all the ducks disappear a game over sign appears with some game over music and then the game goes back to the menu.

HOW DO THE DIFFERENT FEATURES EFFECT THE GAME PLAY OF 'DUCK HUNT'

Black boxes

The black boxes are used to display important information to the user such: whether a new round is starting; if the game is starting; if the user got a perfect score or if the game is over.

A failure in the game

They use loud sounds and harsh flashing images with bright colours that don't match with the colour scheme to show something is wrong.

How sound is used

The sounds are effectively used in this game by complementing the visuals. For example: if a duck is hit a gun sound is played; when the duck is flying a flap sound is played on loop or every time a new round starts a short intro song is played along with the black box with the text for which round you're going on to.

Lower half of the screen

The lower half of the screen contains a shot, hit, score box along with text saying which round the user is on. These are useful for the users game play as these are really good indicators for how the user is doing and where they are in the games progress.

Menu

This is an essential link between the different game types as it allows the user too easily switch between game types and boldly show the user the title of the game.

Visuals/animations

This is not essential to the game play but makes the game more exciting and unique that will help with marketing and 'story line'.

This gives me a good insight to how to create a similar game that has the potential to be just as successful as 'Duck Hunt'

THE FEATURES THAT WILL BE USED FOR THE GAME

INTERVIEW: 24/09/17 PLAN

This interview will give more specific requirements linking to the games visuals, sounds, game play, level system and two-player mode. This interview will allow me to create my first set of requirements suitable to allow me to start solving the problem.

Question layout:

Visuals/sounds

- What animations do you want for the bats?
- Do you want an animation for a win/lose?
    - If yes what would you like it to be?
- What sort of theme do you want for the game?
- Do you want the points to appear above the targets when hit?
- What type of sight do you want for the cursor?
- What types of sounds to you what the bats to make?
- What sounds do you want the clicks to make?
- What sounds do you want the win/lose to make?

Game play

- How many bats do you want to fly up at once?
- Do you want a limited amount of shots?
    - If yes, how many per rounds?
    - And do you want an indicator on screen?
- How would you lose the game?
- How would you like the game to get harder?
- How would like to shoot the bats?
- Would you like a life box at the bottom of the screen?
- Do you want a pause/menu button?
- Do you want a control description as an option on the games menu?

Level/score system

- Would you like a high score table?
- How would you like the score to link to a level system?
- Would you like a score board?
- How many points would you like bat to be worth?

Two player

- Would you want local or online two player?
- What type of two player do you want?

- How would a player win?

INTERVIEW SCRIPT: 24/09/17

VISUALS/SOUNDS

What animations do you want for the bats?

- I want a flapping animation for the bats when they are moving around
- I want the eyes to change to crossed eyes and their tongues stick out when they are shot
- After they are shot I want them to fall to the ground

Do you want an animation for a win/lose?

Yes

If yes what would you like it to be?

A vampire man to pop up from the bottom of the screen and laugh for a loss, similar to like in the duck hunt game

But nothing to happen for a win to give the game a flow

What sort of theme do you want for the game?

I would want different locations at night time that move away from a fictional village towards a mansion. E.g. there would be a forest at some point.

Do you want the points to appear above the targets when hit?

No, but I want a score box on screen

What type of sight do you want for the cursor?

A small red cross with a circle so it can be seen easily but not too big to obstruct the player's "vision" in the game

What types of sounds do you what the bats to make?

- I would like a flap sound for when they are moving around
- A type of scream when they are shoot
- And a falling sound after they have been hit

What sounds do you want the clicks to make?

A gun shot sound to give the illusion that the player is holding a gun

What sounds do you want the win/lose to make if a win/lose is included?

A laughing man for a loss but nothing for a win

GAME PLAY

How many bats do you want to fly up at once?

One

Do you want a limited amount of shots?

Yes

If yes, how many per rounds?

Three, similar to duck hunt

And do you want an indicator on screen?

Yes, a box at the bottom of the screen

How would you lose the game?

You would have three lives and every time a bat escapes you would lose a life. Once all lives have been lost the player has lost

How would you like the game to get harder?

As the game goes on the bats get faster

How would like to shoot the bats?

Using a mouse

Would you like a life box at the bottom of the screen?

Yes

Do you want a pause/menu button?

Yes

Do you want a control description as an option on the games menu?

Yes

LEVEL/SCORE SYSTEM

Would you like a high score table?

Yes, this will add a competitive aspect to the signal player mode

How would you like the score to link to a level system?

Different amounts of scores will change the background. So, the higher the score the closer you get to the mansion. This will happen every 100 points.

Would you like a score board?

There would be a score board in the game. If you get a high enough score you are added to the score board and saved.

How many points would you like bat to be worth?

10

TWO-PLAYER

Would you want local or online two-player?

Local

What type of two player do you want?

One would control the bats and the other would use the mouse to try to shoot the bats.

How would a player win?

The person controlling the bats have to survive a total of five times to win and for the hunter to win they would have to achieve a certain amount of points

A REVIEW OF THE INTERVIEW: 24/09/17 ANSWERS

Through this interview the main features of the game have been established.

Bats

- I want a flapping animation for the bats when they are moving around
- I want the eyes to change to crossed eyes and their tongues stick out when they are shot
- After they are shot I want them to fall to the ground
- I would like a flap sound for when they are moving around
- A type of scream when they are shoot
- A falling sound after they have been hit
- A vampire man will pop up from the bottom of the screen if a bat escapes
- One bat will fly up per mini round
- Each bat is worth 10 points
- The bats get faster as the game goes on

Game indicators

- There will be a score box at the bottom of the screen
- There will be shot box at the bottom of the screen, as you can only shoot three bullets per mini round
- There will be a life box with three lives inside at the bottom of the screen

Score system

- The longer it takes for the bat to be shot the less points you have added. If the player takes too long the bat will “escape”
- Different amounts of scores will change the background. So, the higher the score the closer you get to the mansion. There will be a level up sound alone with it.
- There would be a score board in the game. If you get a high enough score you are added to the score board and saved.
- Three bats would have to escape for the game over to occur
- Every 100 points that are added to the score box the back ground will change
- Include a high score system

Tw- players

- Two-player mode
- One would control the bats and the other would use the mouse to try to shoot the bats.
- The person controlling the bats must survive a total of five times to win and for the hunter to win they would have to achieve a certain amount of points

Game controls

- The game will be paused by pressing the “p” key
- The game will resume if paused when the “p” key is pressed
- The game will be paused by pressing the “m” key
- The mouse will be used to shoot the moving targets

FEATURES OF THE PROPSOED SOLUTION

These are the main proposed features for the game.

Score box	This acts as an indicator for the user as they play the game, so they have an idea of how well they are doing
Three shots per mini round	This was done to make the game more challenging, so the game wouldn’t become too easy. Easy games can become easily boring
Shot box, with three bits of ammo	This acts as an indicator for the user as they play the game, so they have an idea of how many shots they have already used
Three lives in total	This was done to make the game challenging as it gives the user a way to lose the game
Life box, has three lives inside	This acts as an indicator for the user as they play the game, so they have an idea of how many lives they have left
bat flies around for set amount of time then escapes	This gives the user time to react and shoot the bat and try to shoot the bat again if they miss there shot

Menu	This will create a link between the one player and two player modes
As game goes on the bats become faster	This will make the game increasingly difficult hence making the game more interesting
Pixel art	This was the decided theme as it allows the game to keep some realism but remove any major gore. This keeps the range of target audience large.
Sound effects	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.
Mouse controls	This was the decided control for the game as it felt the most appropriate for aiming and shooting the targets
Keys used for menu/pausing	This makes the game more convenient as now they can easily switch game modes or pause the game.
Changing back grounds	This will give the game a hint of “story telling” making it more unique and fun to play
Game over screen	This a strong indicator to the user that their game has finished due to certain conditions.
High score table	This will add a competitive aspect to the game

There is also a time restraint on the game being produced as I will have around six months to produce the game and I am studying two other subjects along with learning the theory for my computer Science course. Therefore, I have provided a second set of features that could be completed if the first set of features are completed.

IF ORIGINAL FEATURES ARE COMPLETED

Time limitations		
Requirement	Justification	Why is this a limitation?
Give the mouse cursor an aiming icon.	This would also make the game more unique due to its design and it will making the aiming easier for the user.	There is a time restriction created due to the fact I am taking other A level course and have less than a year working by myself.
Vampire guy pops up and laughs when you lose a life.	This would make the game visual more interesting	There is a time restriction created due to the fact I am taking other A level course and have less than a year working by myself.
Two player modes.	This would create a local two player competitive mode which will give the user more content	There is a time restriction created due to the fact I am taking other A level course and have less than a year working by myself. I haven't learnt multithreading and I don't have the time to learn about it
Key controls for the bats in the two-player mode.	This would create a two-player environment that removes the need for another controller	There is a time restriction created due to the fact I am taking other A level course and have less than a year working by myself.
Hardware limitations		
Requirement	Justification	Why is this a limitation?
Two mice can be used per computer	This will allow a two-player mode that has two people using their own mice to shoot at the bats	An extra piece of software and hardware will be required for this. I have neither of those things.
The game will have many objects interacting on the screen on screen	This will make the game more interesting to play as more is happing at any one time.	The computer will mostly likely struggle if a lot of things are going at the same time because I don't have a very powerful GPU
Software limitations		
Requirement	Justification	Why is this a limitation?
Impressive visuals	This will make the game more appealing to a wider audience.	I am not able to add lighting effects or shadow casting. This is due to not having access to software such as "Pyro"

HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS

Hardware	Justification
Monitor	To display the visuals of the game back to the user
Mouse	To allow the users to interact with the game by allowing them to click to interact
Touch screen	If the game becomes playable for a smart phone, a touch screens capability's will be required to play the game
Speakers/headphones	To play the sounds generated by the game e.g. intro music or sound effects
10Mb of secondary memory space	The program for the game will not be very big and the sprites will not take up very much memory
2Gb of RAM	An average computer holds 2Gb's of memory meaning most people will be able to play the game with no problems.
33MHz processor	This is the minimum required processing power to run desktop version of monkey-X

SOFTWARE REQUIREMENTS

Software	Justification
Monkey X run-time library	This allows a user to run a monkey-X game without having to download the monkey-X software
If a windows powered desktop is used:	
OpenAL Windows drivers	These applications allow the game to run on a windows platform
OpenAL API for audio	
OpenGL API for graphics rendering	
Microsoft Visual C++ Express 2010 or MinGW 4.8.1	
If an IOS powered device is used:	
Apple mac computer running OSX	These applications allow the game to run on a IOS platform
XCode developer with IOS SDK	
If an Android powered device is used:	
Android SDK. Only require 'SDK Tools' version	These applications allow the game to run on an Android platform
Java SE JDK 32bit version	
Apache Ant java library	
If HTML5 is used:	
HTML5 capable browser such as: google chrome, fire fox, edge or safari	These applications allow the game to run on a HTML5 platform
HTML5 capable browser such as: google chrome, fire fox, edge or safari	

SUCCESS CRITERIA

No.	Requirements	Justification	☑	Reference
1.	Start-up screen.	This is done to show the user what game they are about to play	☐	A feature from the Nintendo: Duck Hunt game
2.	The player gains 10 points when a bat is shot	This gives the user points that will be added on to their score	☐	Interview: 24/09/17
3.	Score box in the bottom right of the screen.	This places the indicator in a clear place for the user to see	☐	A feature from the Nintendo: Duck Hunt game
4.	Score board is shown at the end of each game.	This gives the user a goal to work towards making the game more interesting	☐	Interview: 24/09/17
5.	Three shots per mini round	This was done to make the game more challenging so the game wouldn't become too easy. Easy games can become easily boring	☐	A feature from the Nintendo: Duck Hunt game
6.	Ammo disappears as the player shoots at the bats until there is no ammo left so the player can no longer shoot.	This acts as an indicator for the user as they play the game so they have an idea of how many shots they have already used	☐	A feature from the Nintendo: Duck Hunt game
7.	Shot box, with three bits of ammo, in the bottom left of the screen.	This places the indicator in a clear place for the user to see	☐	A feature from the Nintendo: Duck Hunt game
8.	Three lives in total	This was done to make the game challenging as it gives the user a way to lose the game	☐	Interview: 24/09/17

9.	If a bat “escapes” the user loses a life	This acts as an indicator for the user as they play the game so they have an idea of how many lives they have left	☐	Interview: 24/09/17
10.	Life box, has three lives inside, in the middle of the bottom of the screen.	This places the indicator in a clear place for the user to see	☐	Interview: 24/09/17
11.	A counter is started when the bat starts to fly around. It decreases until 0. This will act as a timer	This gives the user time to react and shoot the bat and try to shoot the bat again if they miss there shot	☐	Interview: 24/09/17
12.	The bat flies around until score counter becomes 0 then the sprite disappears/ “escapes”	This will make the game challenging as you must try to shoot the bat before it disappears	☐	Interview: 24/09/17
13.	Menu with the title and three options, being the one player mode, two player mode and controls description.	This will create a link between the one player and two player modes	☐	A feature from the Nintendo: Duck Hunt game
14.	As the game goes on the bats become faster. This happens every multiple of 100	This will make the game increasingly difficult hence making the game more interesting	☐	A feature from the Nintendo: Duck Hunt game
15.	All sprites and backgrounds will be in a pixel art style	This was the decided theme as it allows the game to keep some realism but remove any major gore. This keeps the range of target audience large.	☐	Interview: 18/09/17
16.	Sound effect for the shooting. This will occur when bullets are used.	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.	☐	A feature from the Nintendo: Duck Hunt game

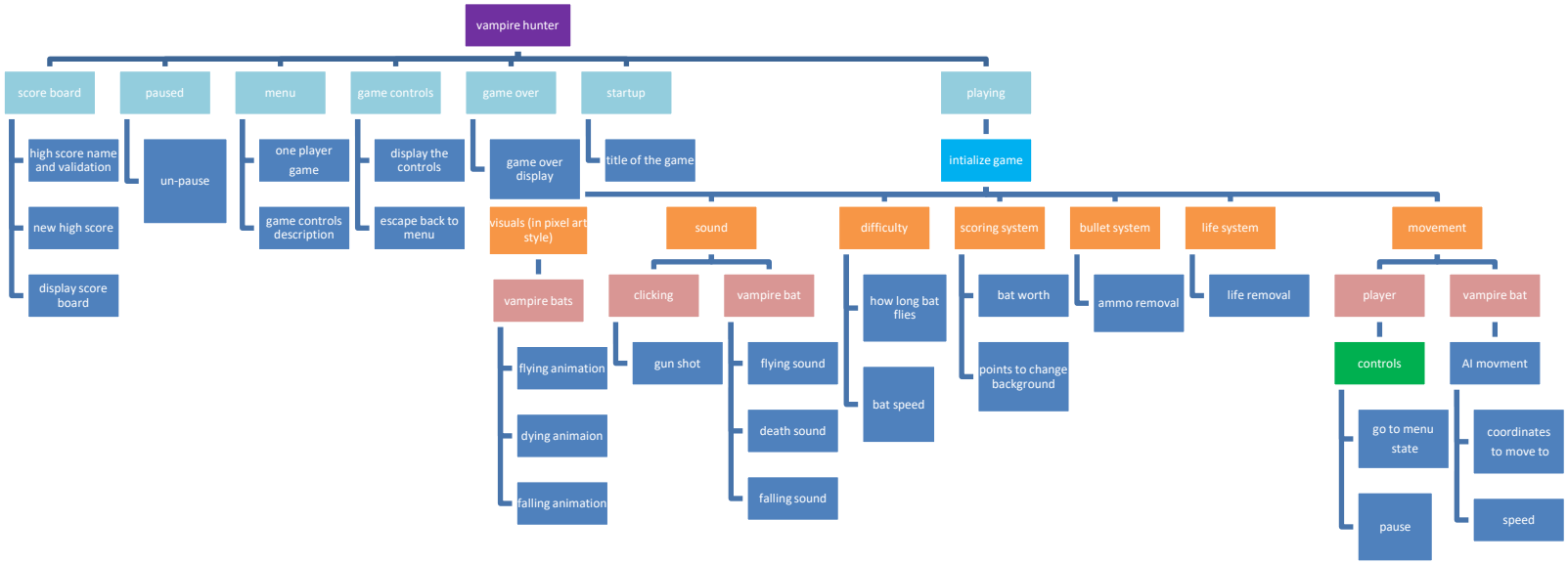
17.	Sound effect for the bat being “killed”. A hurling sound is played	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.	☐	A feature from the Nintendo: Duck Hunt game
18.	Sound effect for the bat flying around	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.	☐	A feature from the Nintendo: Duck Hunt game
19.	Sound effect for the bat falling	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.	☐	A feature from the Nintendo: Duck Hunt game
20.	The animation for the bat being “killed”.	This will make the game more unique and help as an indicator for the user when playing.	☐	A feature from the Nintendo: Duck Hunt game
21.	The animation for the bat flying around	This will make the game more unique and help as an indicator for the user when playing.	☐	A feature from the Nintendo: Duck Hunt game
22.	The animation for the bat falling	This will make the game more unique and help as an indicator for the user when playing.	☐	A feature from the Nintendo: Duck Hunt game
23.	Press the ‘m’ key to access the menu from the playing state.	This will allow the user to easily switch between the one and two player game modes	☐	Interview: 24/09/17

24.	Press the 'p' key to enter the paused game state from the playing state.	This makes the game more convenient as the players have the freedom to stop playing whenever and continue straight away with no difficulty	☐	Interview: 24/09/17
25.	Press the 'p' key to re-enter the playing game state from the paused state.	Allow the user to resume playing the game	☐	Interview: 24/09/17
26.	Press the escape key to re-enter the menu game state from the playing state.	This will allow the user to return to the menu state from the control state	☐	interview: 24/09/17
27.	Use the mouse to click on the moving targets in the game with left click.	This was the decided control for the game as it felt the most appropriate for aiming and shooting the targets	☐	Interview: 24/09/17
28.	Use the mouse to navigate on the menu screen	This was the decided as a control for navigating around the menu as it is already used else were in the game. This is done for continuity purposes.	☐	Interview: 24/09/17
29.	Back ground will change every time the score box's value increase by 100 points. These back grounds will be of different locations	This will give the game a hint of "story telling" making it more unique and fun to play	☐	Interview: 24/09/17
30.	When the player loses all their health a game over screen appears	It is a clear indicator for the user	☐	A feature from the Nintendo: Duck Hunt game
31.	A sound effect for when health is lost	It is a clear indicator for the user	☐	A feature from the Nintendo: Duck Hunt game
32.	Game music for some of the game states	Makes the game more interesting by giving the game a bit	☐	A feature from the Nintendo: Duck Hunt game

		more depth with sound		

DESIGN

SYSTEMS DIAGRAM



Why am I using this approach to help design my problem?

By using a top-down module design it helps me to identify what I need before I start coding as I am able to break the problem down into smaller parts allowing me to look at each part of the problem more specifically. This will allow me to use the computational thinking skill “thinking ahead” to determine the inputs and outputs; this also allows me to use the computational thinking skill “thinking logically” to look at the where decisions are required and what effects it will have on other parts of the solution. This will give my working solution some structure letting me work towards my solution systematically/ efficiently.

EXPLANATION OF EACH MODULE

Score board

new high score name and validation

This checks if the users final score is greater that then score boards lowest score, if yes then it checks whether a suitable name is given in. if a suitable name is given then it is used for the score board.

New high score

This places the users final score on the score board in the suitable place to correspond with the other final scores.

Display score board

This displays the score board for an amount of time then changes the game state for the score board state to the menu state.

Paused

Un-pause

When the "p" key is pressed the game will change from paused state to the playing state.

Menu

One player game

By selecting the “one player” option on the menu the game state will change from the menu state to the playing state.

Game controls description

By selecting the “games control” option on the menu the game state will change from the menu state to the game controls state.

Game controls

Back to menu

By pressing “m” the game state will change from the games controls state back to the menu state.

Game over

Game over display

If three bats “escape” (the life variable’s value is zero), when in the playing state, the state will change to the game over state and load up a game over screen.

Startup

Title of the game

The code will start in the start-up state displaying the title of the game as an image and then change to the menu state after a small amount of time.

Playing

Initialize game

At the start of every new game variables such as the life/bullet variable are reset to their starting values, the score variable is rest to zero, etc.

Visuals (in pixel art style)

Vampire bats

Flying animation

Three bat field sprites will be played in loop to give the illusion of flying when the target is moving around the screen.

Dying animation

When the cursor overlaps with the target and there is a click the target is “killed”. This results in the target’s field sprite changing in to a dead bat image.

Falling animation

After the dead bats field sprite is displayed it changes to the field sprite of a bat falling. The sprite will move straight down until out of site.

Sound

Clicking

Gun shot

When the user left mouse clicks within the screen a short gunshot sound will be played if the bullet if the user still has bullets left to “fire”.

Vampire bat

Flying sound

A flapping sound is played whenever a target is spawned on to the screen and moving around.

Dying sound

When the cursor overlaps with the target and the user left mouse clicks the target is “killed”. This results in a hurling sound being played to represent the bat dying.

Falling sound

After the target has been “killed” a whistling sound is played to represent the target falling.

Difficulty

How long bat flies

There will be a counter that is reset every time a new target is spawned. The counter will have a negative step count until it hits zero, when this happens the sprite will disappear/ “escapes”, this will also remove 1 from the life variable value. If the target is hit before the counter hits zero, the counter is reset anyway for the new target being spawned.

Bat speed

Every time the score variable value increases by a multiple of 100 the time between each line execution is reduced meaning the target is on screen for less time and moves more quickly. This makes the game more difficult.

Scoring system

Bat worth

When a bat is hit 10 points are added to the users score.

Points to change background

Every time the score variable value increases by a multiple of 100 the background changes to another image.

Bullet system

Bullet removal

Three bullets per mini round meaning the user has three clicks to use per mini round, in code this is a bullet variable with a value (3).

When the user clicks on the screen the bullet value is decrease by 1 until the value becomes zero. When the bullet variables value is zero the user will no longer be able to “kill” the targets as there are no more “bullets” left; also, the gun shot sound will no longer be played when the bullet variable is zero. The bullet sprites are removed every click until the three clicks have been used up. Then the box used to “contain” the bullets is just displayed plank

Life system

Life removal

Each time the user plays a game they are given three lives, in code this is life variable with a value (3).

The life sprites are removed every time a bat escapes until the three bats have escaped (the life's variable value is decreased by 1 until zero). When the life variable is zero the game state will change from playing to the game over state.

Movement

Player

Controls

Go to menu state

If the "m" key is pressed, when in the playing state, the game state playing changes to the menu state.

Pause

When the "p" key is pressed the game will change from the playing state to the paused state.

Vampire bat

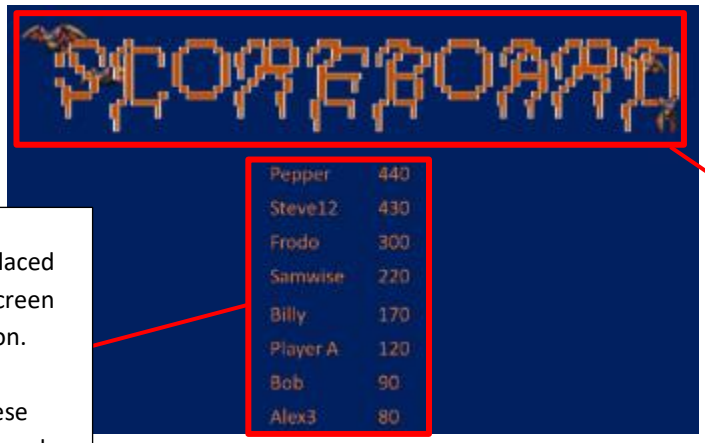
AI movement

- Coordinates to move to
- Speed

Candidate Name: [redacted] Candidate Number: [redacted]

USABILITY FEATURES

Score board



The high scores are placed in the center of the screen to draw most attention. The higher scores are higher up the list. These things where done to make the high scores as clear as possible.

The title of the game state is clearly stated at the top of the screen. This informs the user what part of the game they are at.

Paused

The title of the game state is clearly stated at the top of the screen. This informs the user what part of the game they are at.

This game state is just to pause the game so there is only an instruction, on screen, to inform the user how to get back to playing the game. This helps in making the game features easy to use.

Candidate Name: [redacted] Candidate Number: [redacted]

Menu

The title of the game state is clearly stated at the top of the screen. This informs the user what part of the game they are at.

The game options are in a big clear font so they can be clearly seen. This is done so the user knows what game options are and are not available to them.

The mouse and the left click will be used to select what game state the user will switch to in the game. Moving the mouse and left clicking is easy to do making the menu interface easy to use.

Game controls description

The title of the game state is clearly stated at the top of the screen. This informs the user what part of the game they are at.

The game controls are the only thing displayed along with the title on game controls screen. Keeping this screen simple is key to help the user understand what the game controls are.

The menu and paused states can be accessed by pressing a key meaning no buttons will have to be displayed on screen when playing the actual game. This will keep the gaming screen neat and only filled with stuff that are required to play. This will stop anything from being taken away from the gaming experience.

Candidate Name: [redacted] Candidate Number: [redacted]

Game over

The pixel art skull was include in the game over screen because it has a link to death. This helps bring the message of failure across.

The title game over acts an indicator as it lets the user know that there game has finished and not a glitch. The font was also selected as it has a link to death.

Start up

The screen that displays the title is shown at the start of the game to inform the user of what game they are playing. This works as a sort of marketing technique as people will be able to put a name on the game they are playing.

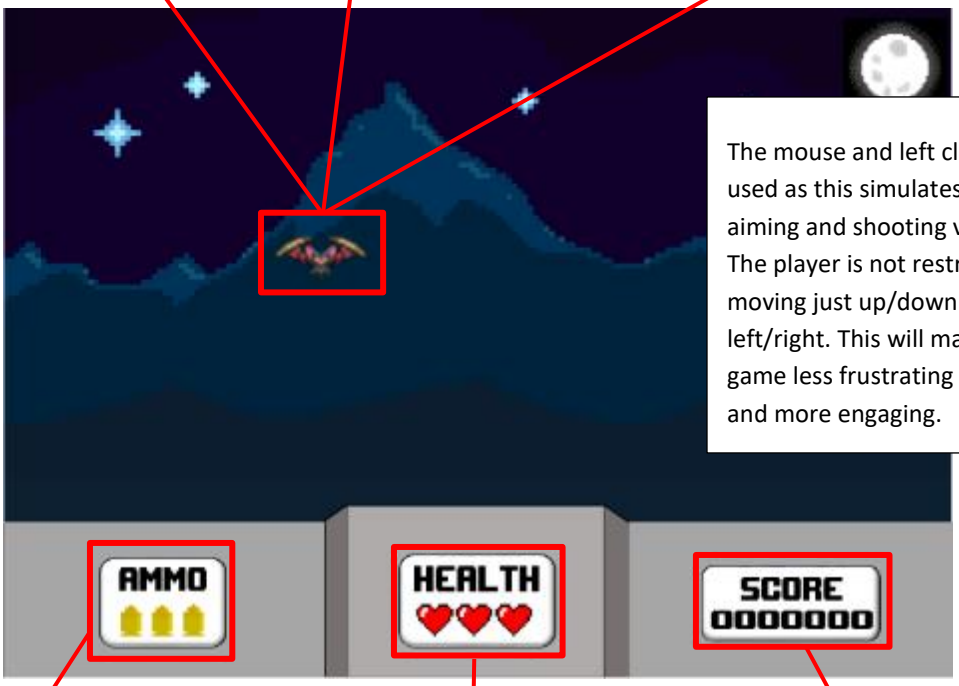
Candidate Name: [redacted] Candidate Number: [redacted]

Playing

These three filed sprites will play on loop to give the illusion that the target is flying. This will make the moving target seem more like a bat. This will have a flapping sound played along with the animation.

This will be displayed after the bat has been hit and the target is falling. This will make the game more interesting to play as the illusion of gravity is include. This will have a whistling sound played along with the animation.

These field sprite will be displayed when the bat has been hit. This acts as an indicator to let the user know when the bat has been hit. This will have a hurling sound played along with the animation.



The mouse and left click was used as this simulates the act of aiming and shooting very well. The player is not restricted to moving just up/down or left/right. This will make the game less frustrating to play and more engaging.

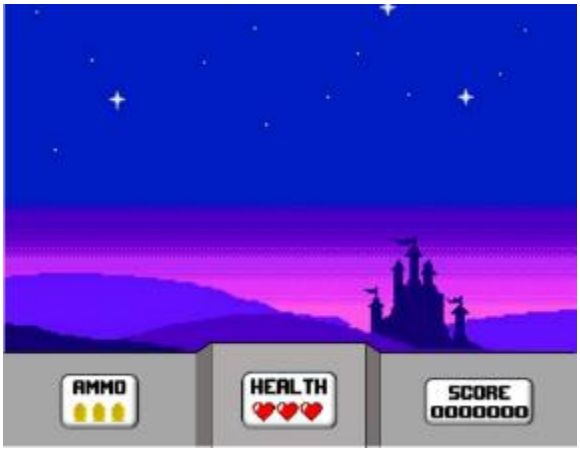
Every time the user left mouse clicks on the screen a bullet is removed until there are no bullets left. This acts as an indicator for when the user is playing the actual game.

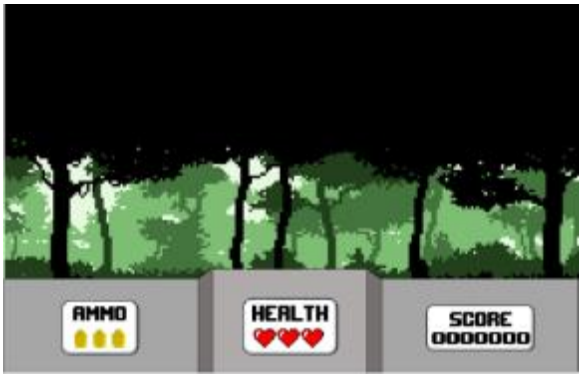
Every time a bat escapes a heart is removed until there are no more hearts left and it is games over. This acts as an indicator for when the user is playing the actual game.

Every time the user gains points the value of those points are added to the score. This box acts as an indicator for when the user is playing the actual game.

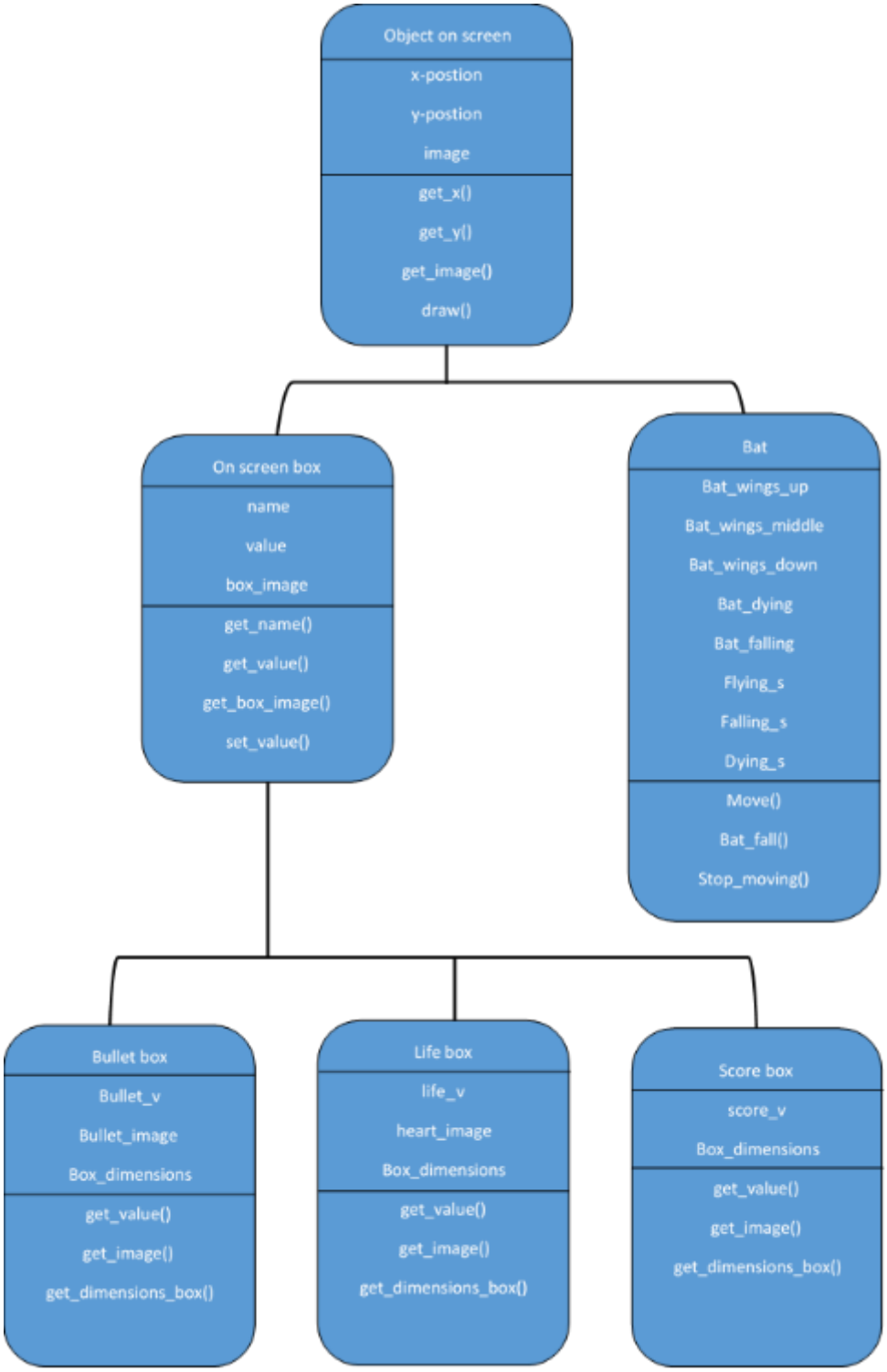
Candidate Name: [redacted] Candidate Number: [redacted]

The following screen designs are of the backgrounds that change for every multiple of 100. This was done so the users knows when the game is going to increase in difficulty.





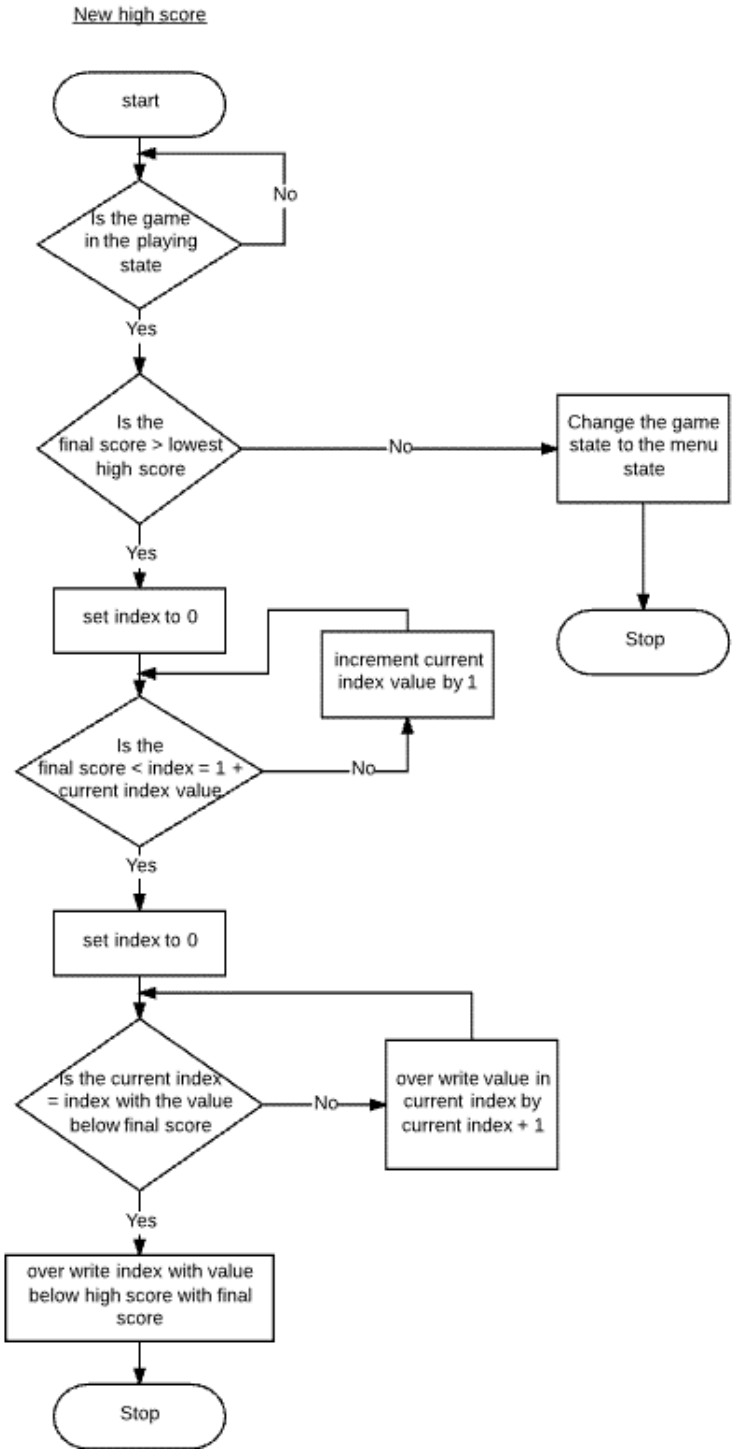
CLASSES


Object on screen
x-postion
y-postion
image
get_x()
get_y()
get_image()
draw()
On screen box
name
value
box_image
get_name()
get_value()
get_box_image()
set_value()
Bat
Bat_wings_up
Bat_wings_middle
Bat_wings_down
Bat_dying
Bat_falling
Flying_s
Falling_s
Dying_s
Move()
Bat_fall()
Stop_moving()
Bullet box
Bullet_v
Bullet_image
Box_dimensions
get_value()
get_image()
get_dimensions_box()
Life box
life_v
heart_image
Box_dimensions
get_value()
get_image()
get_dimensions_box()
Score box
score_v
Box_dimensions
get_value()
get_image()
get_dimensions_box()

Candidate Name: [redacted] Candidate Number: [redacted]

ALGORITHMS

Score board



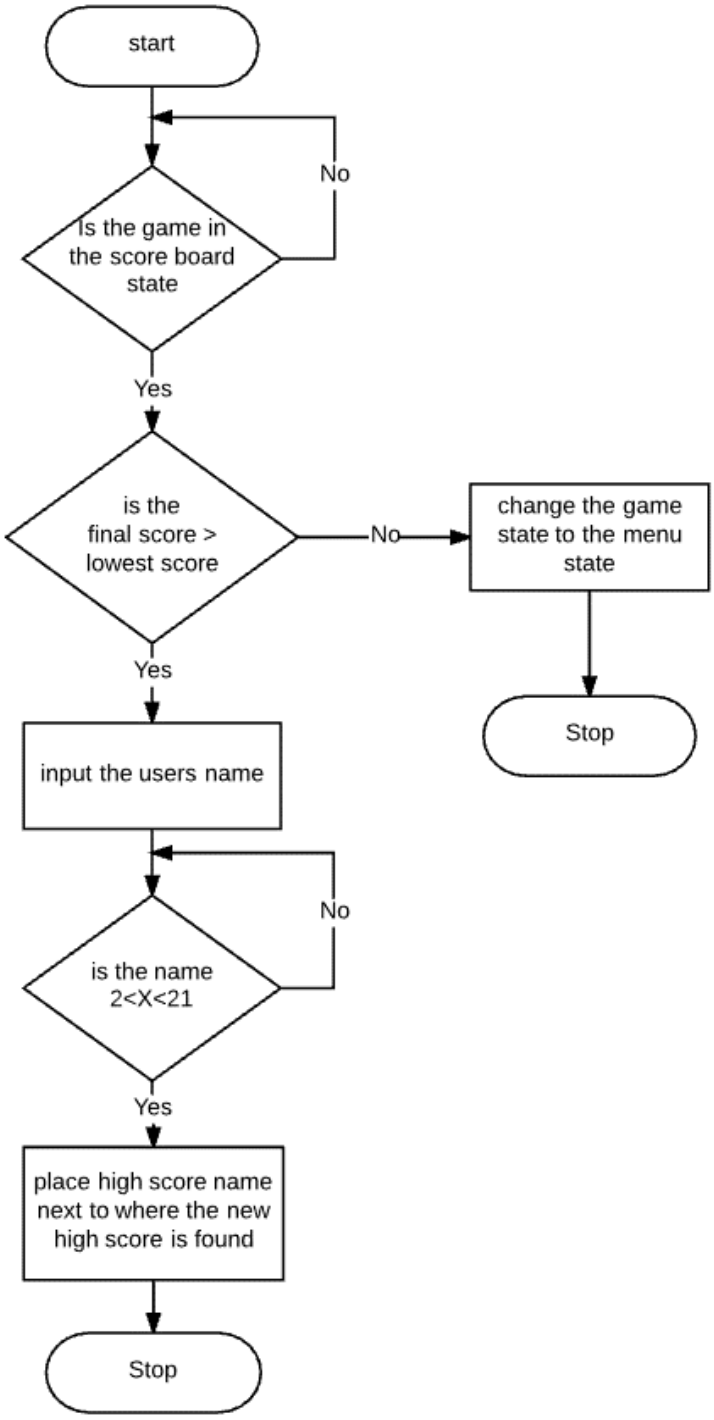
Array used for tracing execution

Index	name	score
7	Pepper	440
6	Steve12	430
5	Frodo	300
4	Samwise	220
3	Billy	170
2	Player A	120
1	Bob	90
0	Alex	80

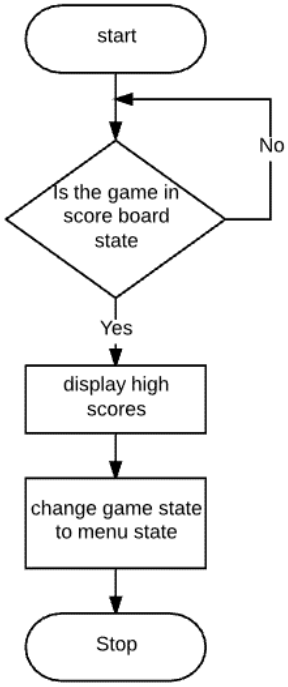
Tracing execution

Game state	Final score	Lowest score	Index
playing	200	80	Null
			0
			1
			2
			3
			0
			1
			2
end			

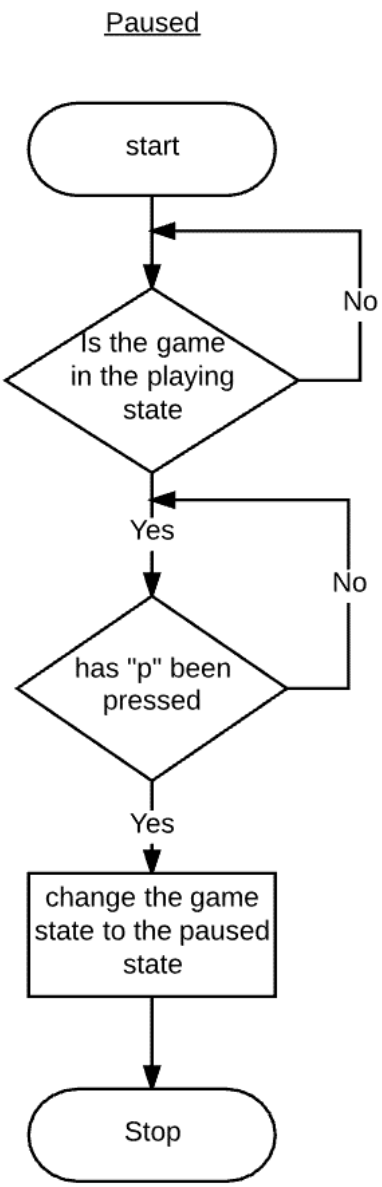
High score name and validation

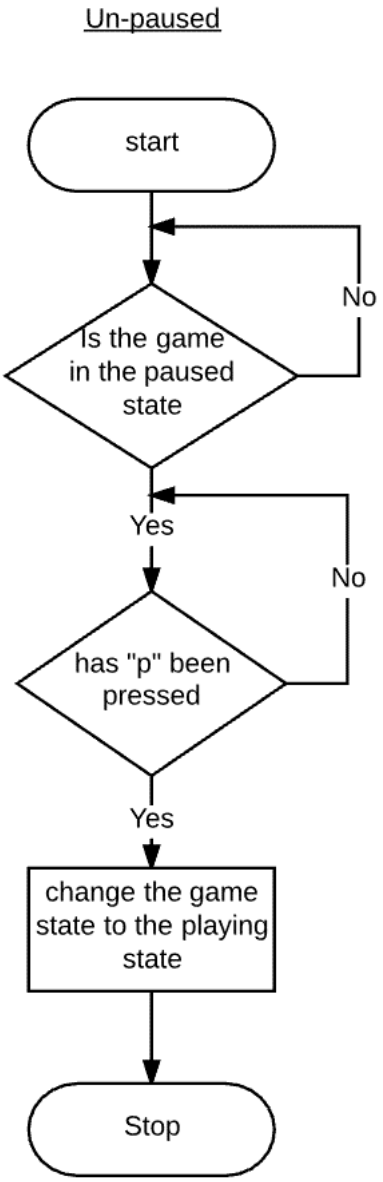


Display high scores

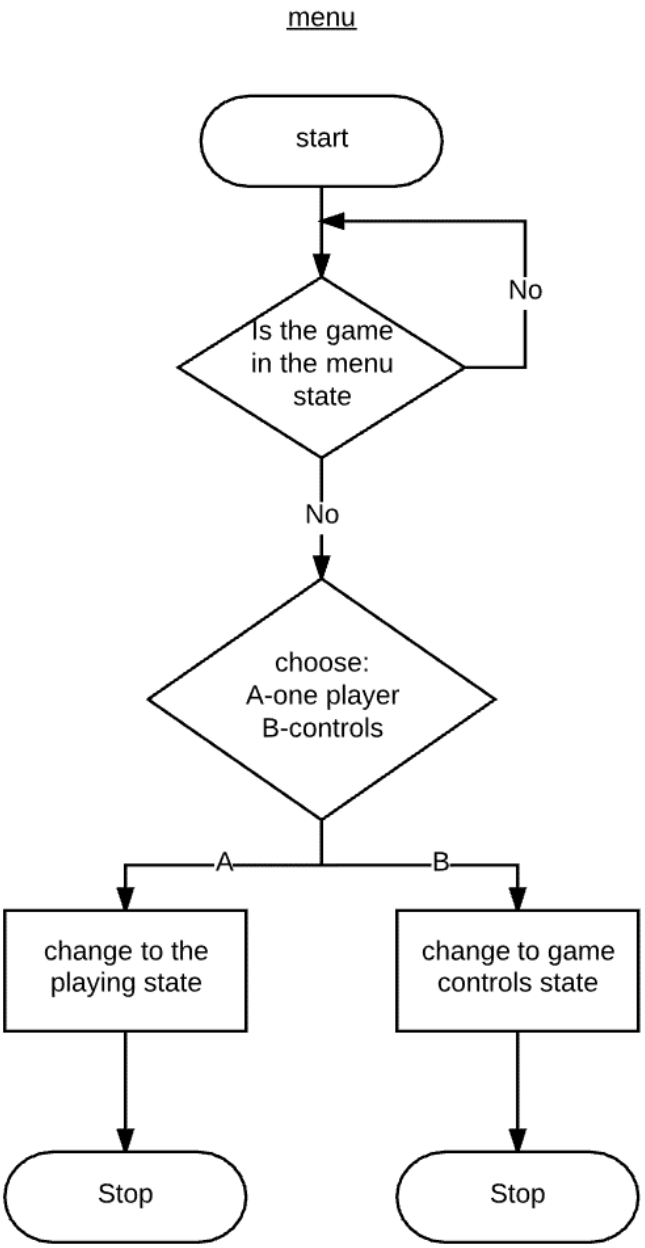


Paused

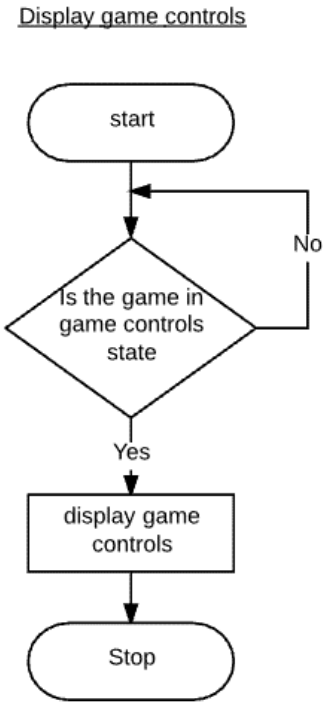

Paused
start
Is the game in the playing state
No
Yes
has "p" been pressed
No
Yes
change the game state to the paused state
Stop


Un-paused
start
Is the game in the paused state
No
Yes
has "p" been pressed
No
Yes
change the game state to the playing state
Stop

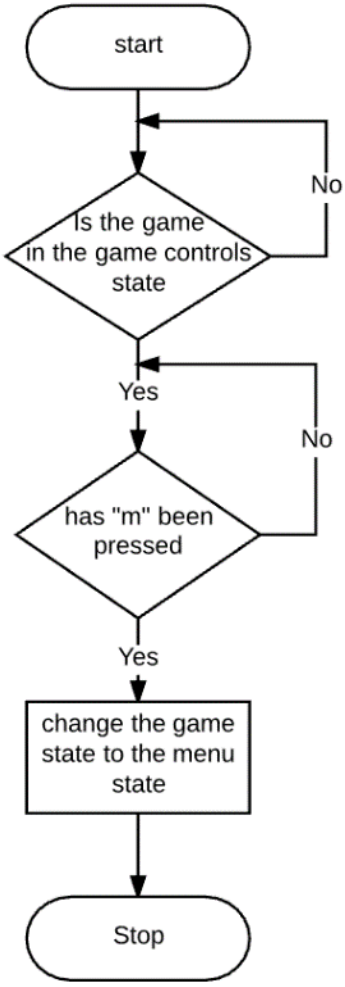
Menu


menu
start
Is the game in the menu state
No
No
choose:
A-one player
B-controls
A
B
change to the playing state
change to game controls state
Stop
Stop

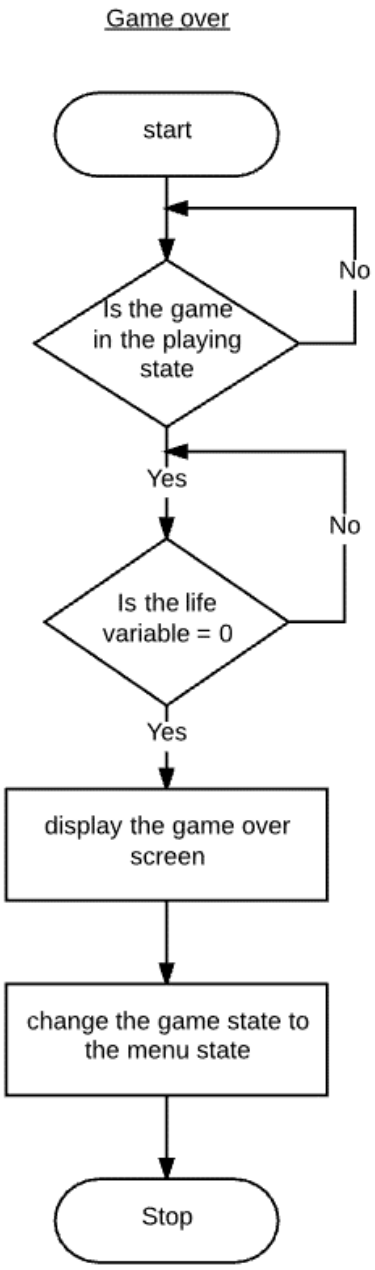
Game controls



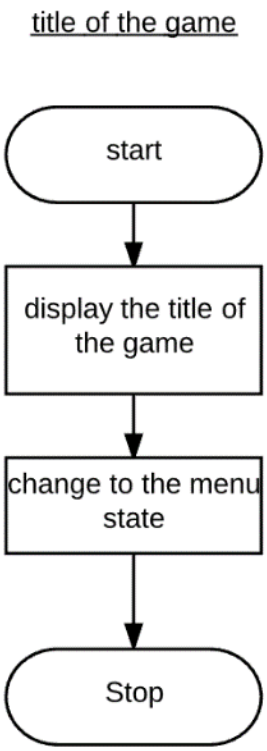
Escape to menu



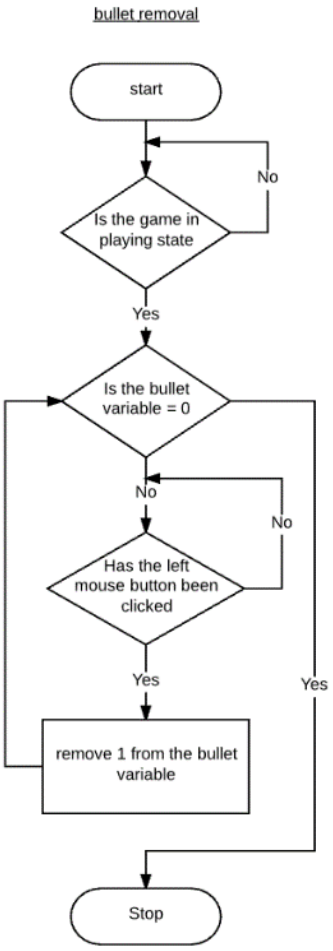
Game over

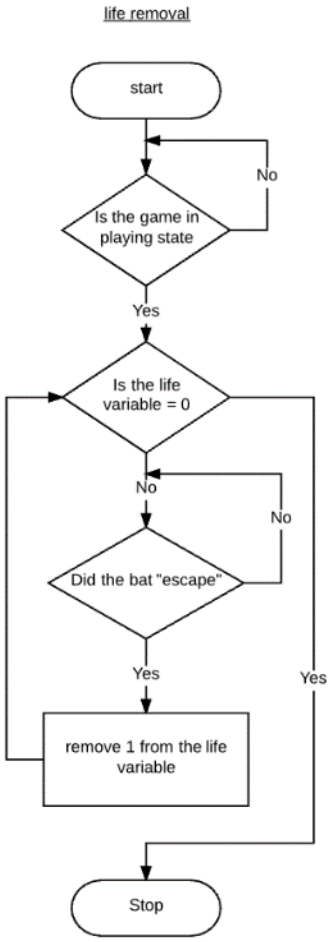

Game over
start
Is the game in the playing state
No
Yes
Is the life variable = 0
No
Yes
display the game over screen
change the game state to the menu state
Stop

Start up


title of the game
start
display the title of the game
change to the menu state
Stop

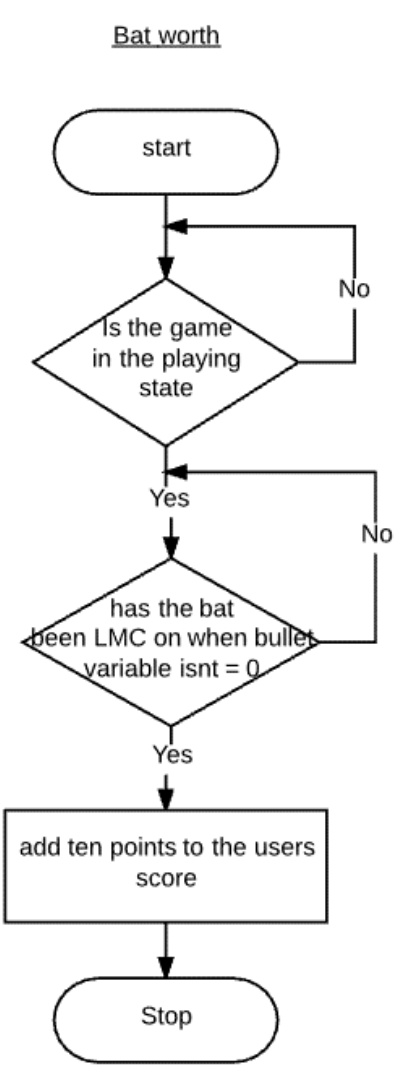
Playing

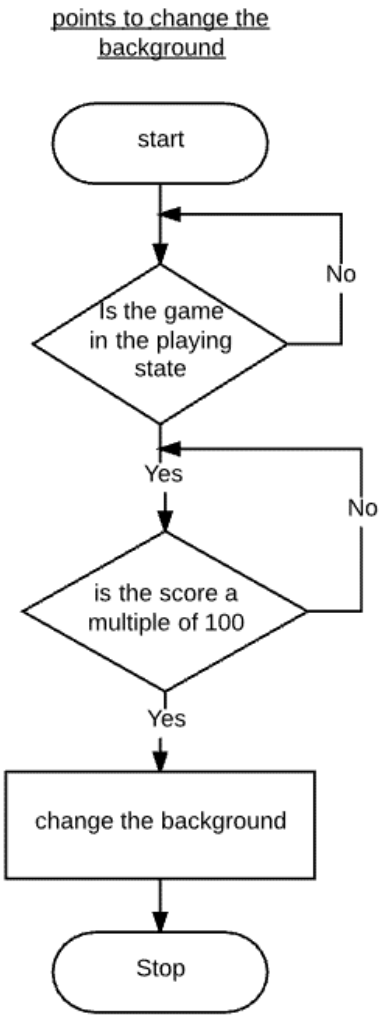


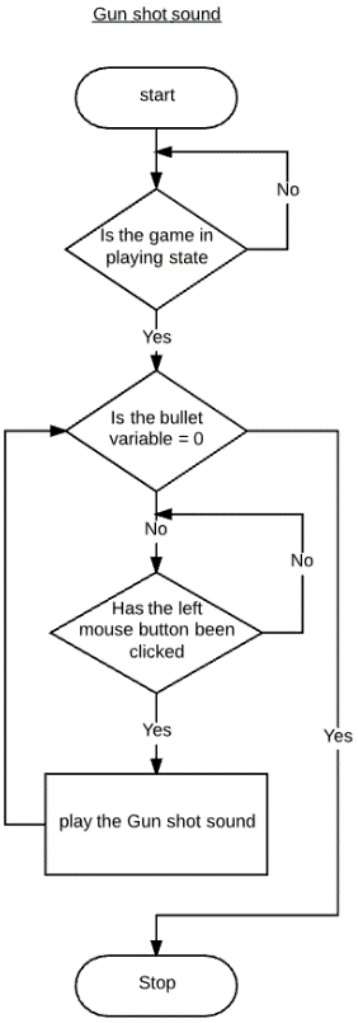


Tracing execution		
Game state	bullet variable	Bat has “escaped”
playing	2	True
	1	
	0	
end		

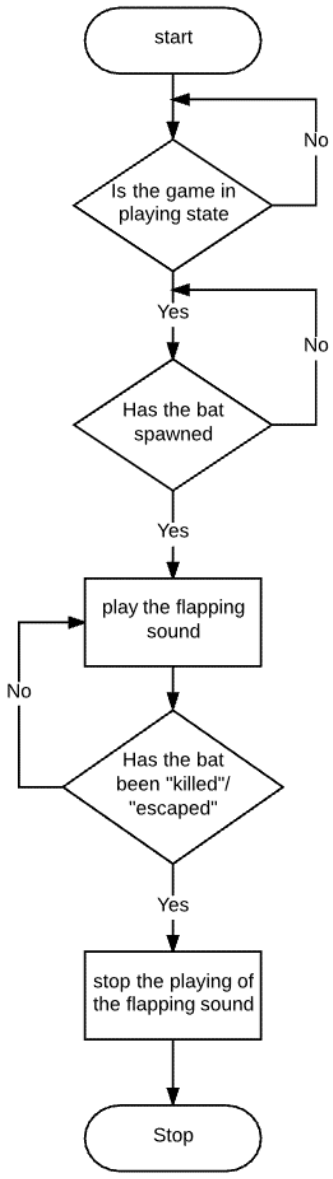
Tracing execution		
Game state	Life variable	Bat has “escaped”
playing	3	True
	2	
	1	
	0	
end		


Bat worth
start
Is the game in the playing state
No
Yes
has the bat been LMC on when bullet variable isnt = 0
No
Yes
add ten points to the users score
Stop

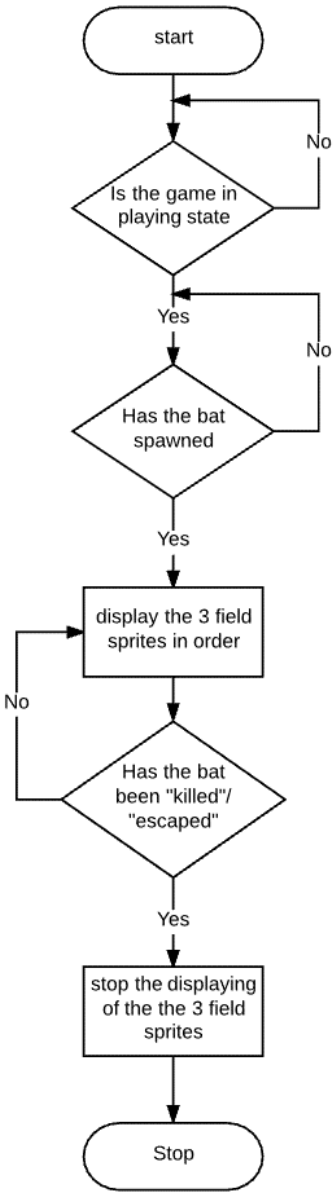

points to change the background
start
Is the game in the playing state
No
Yes
is the score a multiple of 100
No
Yes
change the background
Stop


Gun shot sound
start
No
Is the game in playing state
Yes
Is the bullet variable = 0
No
No
Has the left mouse button been clicked
Yes
Yes
play the Gun shot sound
Stop

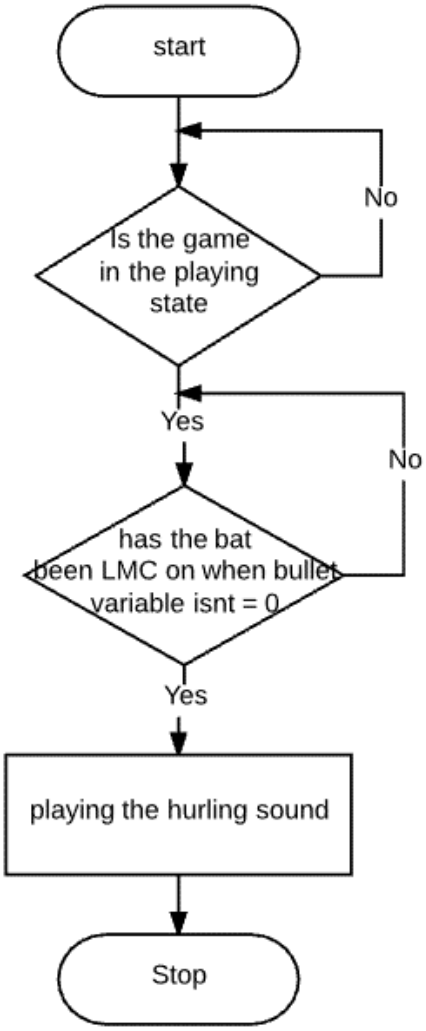
flying sound



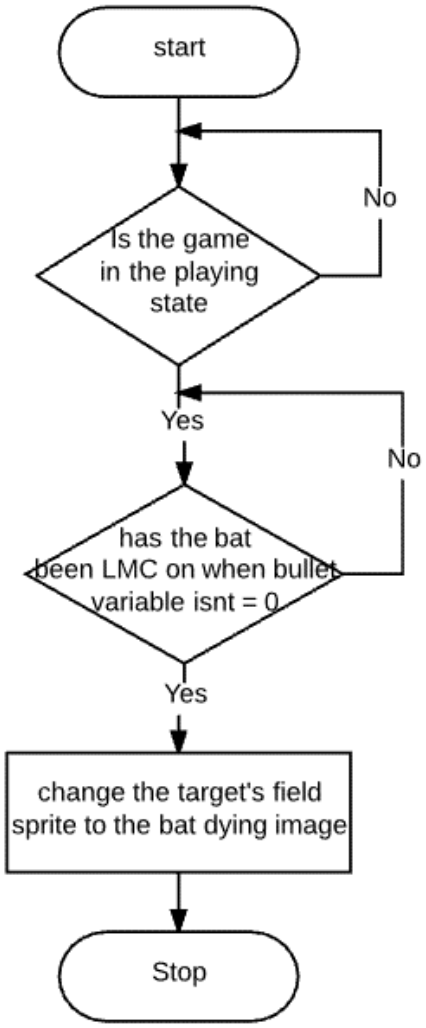
flying animation



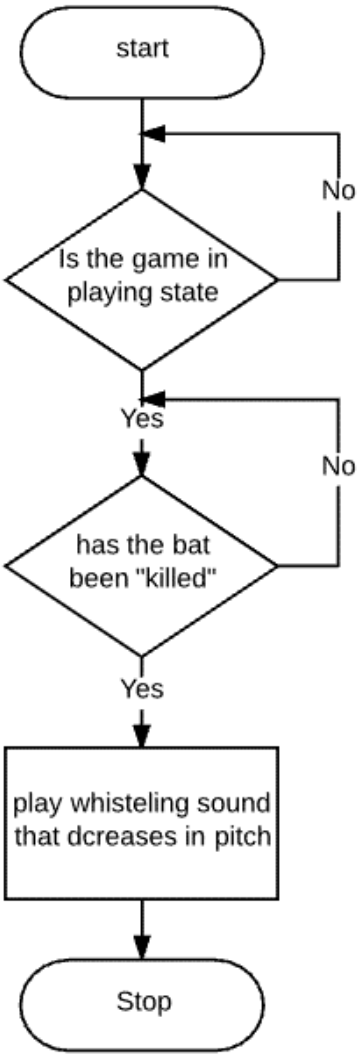
dying sound



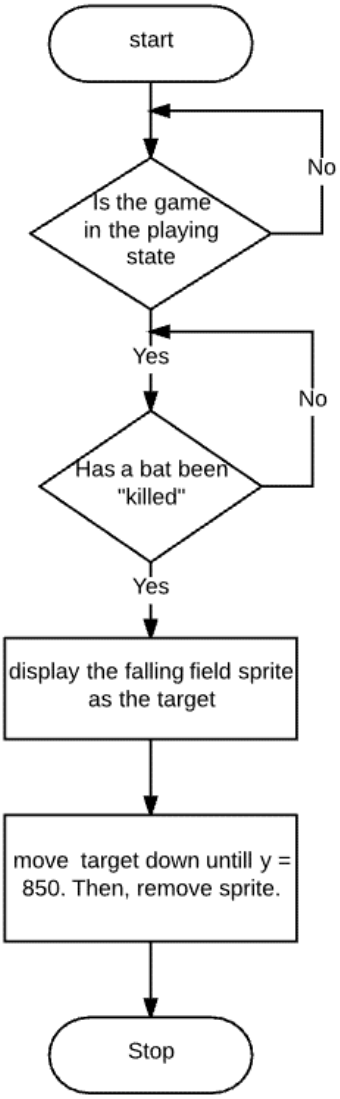
dying animation



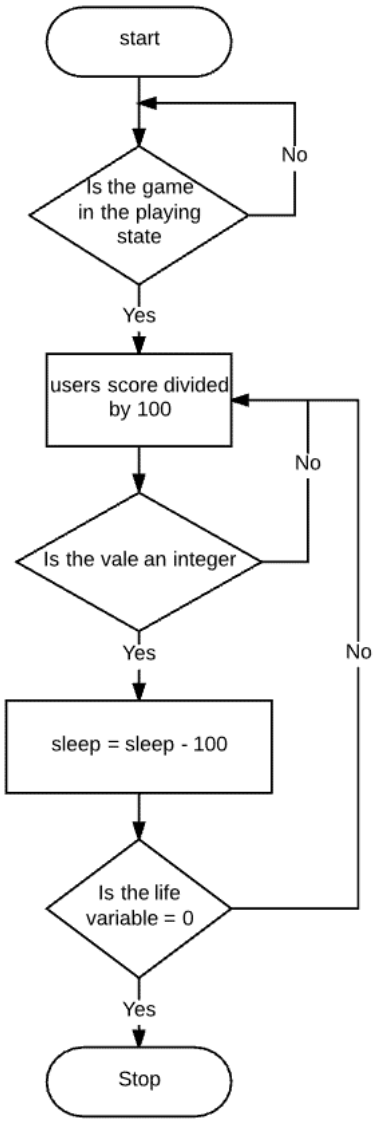
falling sound



falling animation



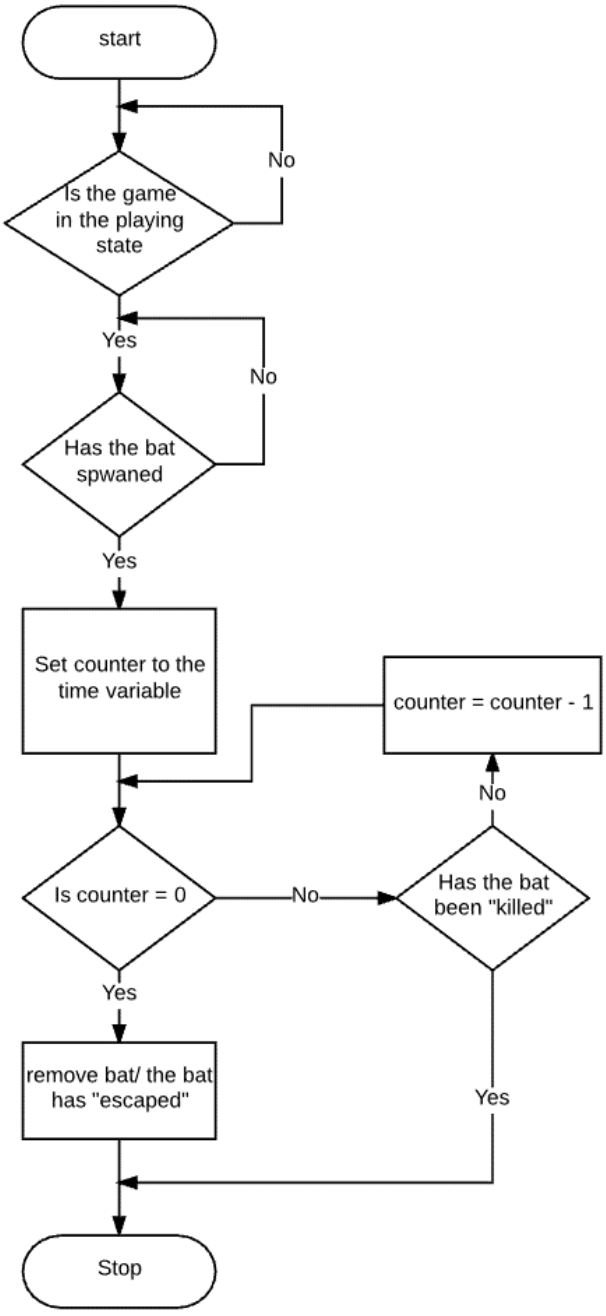
Speed increases for the bat



Tracing execution

Game state	Users score	Sleep variable	Life variable
Playing	300	600	0
		500	
end			
playing	400	500	0
		400	
end			

How long the bat flies

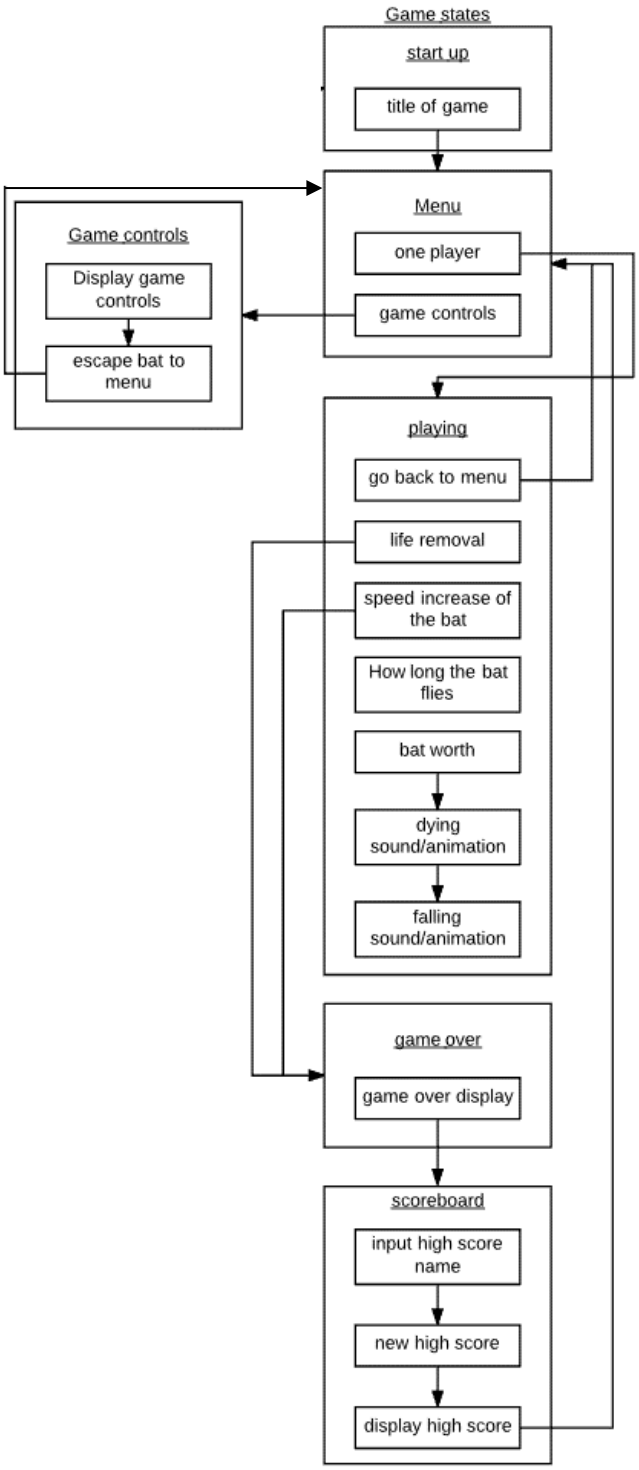


Ai algorithms should be inserted here

I’m not sure how to tackle this problem yet so I will have to do some more research before I can design this algorithm.

Candidate Name: [redacted] Candidate Number: [redacted]

Linking the algorithms



KEY VARIABLES AND DATA STRUCTURES

Method	Name	Data type	Explanation	Justification
Score board				
Game state	Score_board	N/A	The game is at a part where the score board is displayed and new high scores can be added	This is where all the code related to the score board is
Procedure	hs_score	N/A	A new high score is added to the score board	This will give the game a competitive aspect because as players getter better more new high scores will be created
variable	Score_v	integer	This is the users score	This will allow the value of the users score to be used for game logic
procedure	hs_name_and_validation	N/A	This will only allow high score names that are 2<X<21 to be input	This will stop large names being input which will disturb the screen layout of the score board
Array	hs_array	N/A	This is the structure used to organize the scoreboard data	This will allow the new high scores to be rearranged
Text file	hs_file	.txt	This is to store the high scores	This will allow the high scores to be saved
paused				
Game state	Paused	N/A	The game is at a part where it is paused	This is where all the code related to the pausing is

Procedure	Un-pause	N/A	The game state changes from playing to paused	This will allow the user to resume playing the game
Menu				
Game state	Menu	N/A	The game is at a part where the user can choose game options	This is where all the code related to the menu is
Procedure	One_player	N/A	The game state changes from menu state to playing state	This creates a link between the menu state and playing state
Procedure	Game_controls	N/A	The game state changes from menu state to the games controls state	This creates a link between the menu state and game controls state
Game controls				
Game state	Game_controls	N/A	The game is at a part where the controls are displayed to the user	This is where all the code related to the games controls is
Procedure	Controls	N/A	The controls are loaded as an image	This informs the user what the controls are
Procedure	Escape_menu	N/A	The game state changes from the games controls state to the menu state	This allows the user to exit the game controls state and return to the menu state
Game over				
Game state	Game_over	N/A	The game is at a part where the game ends and game over is displayed	This is where all the code related to the game over is
Procedure	Game_over_display	N/A	A game over image is loaded up	This clearly shows the user that the game has ended
Start up				

Game state	Start_up	N/A	The game is at a part where the game is initialised and the title of the game is displayed	This is where all the code related to the games start up is
Procedure	Title	N/A	Game title is loaded up as an image	This tells the user what game they are playing
Variable	sleep	Integer	Value with the time between each line execution	This is the value that will be changed to increase the speed of the bats
Variable	time	integer	The value that the targets counter is reset to	The bats fly around for a consistent amount of time
Playing				
Game state	Playing	N/A	The game is at a part where the actual game is being played	This is where all the code related to the playing of the actual game is
Procedure	Initialize	N/A	The game is initialized	This is to set all the variables to there
Procedure	Pause	N/A	The game state is changed from the playing state to the menu state	This makes the game more convenient as the players have the freedom to stop playing whenever and resume
Procedure	B_menu	N/A	The game state changes from the playing state to the menu state	This will allow the user to easily switch between the one and two player game modes as well as looking up the game controls
procedure	Bat_time_flies	N/A	The bat “flies” around until the counter reaches 0	This gives the user a certain amount of time to shoot the bat. This

				makes the game more challenging
Variable	Counter_value	Integer	The value the counter is rest to	This value will determine how long the bat will "fly around"
Variable	Life_v	Integer	A variable with the value 3	This gives the user a finite amount of lives making the game more challenging
Variable	Bullet_v	Integer	A variable with the value 3	This gives the user a finite amount of bullet making the game more challenging
Procedure	Bat_speed	N/A	This will increase the speed of the bat every multiple of 100	This will make the game more challenging for the user
Procedure	Bat_worth	N/A	10 points are added to the users score	This gives the user points that will be added on to their score
procedure	Bg_change	N/A	The back ground changes ever multiple of 100 for the users score	This makes the game more interesting to look at and acts as an indicator for when the bat increases in speed
Procedure	bullet_removal	N/A	1 is removed from the bullet variable and one of the bullet field sprites are removed	This makes the game more challenging as every time the user misses the target a bullet is removed
Procedure	Life_removal	N/A	1 is removed from the life variable and one of the life field sprites are removed	This makes the game more challenging as every time a bat "escapes" the user loses a life

Sound file	Flapping_s	.wav	Sound effect of flapping wings are played	This will help as an indicator for the user as sometimes visuals can go unnoticed.
Sound file	Dying_s	.wav	Sound effect of a hurling sound is played	This will help as an indicator for the user as sometimes visuals can go unnoticed.
Sound effect	Falling_s	.wav	Sound effect of a whistling sound that decreases in pitch is played	This will help as an indicator for the user as sometimes visuals can go unnoticed.
Sound effect	Gun_shot	.wav	Sound effect of a gunshot sound is played	This will help as an indicator for the user as sometimes visuals can go unnoticed.
Field sprite	Bat_wings_up	.jpeg	Pixel art image of a bat with wings up	This makes the game unique and help as an indicator for the user.
Field sprite	Bat_wings_middle	.jpeg	Pixel art image of a bat with wings middle height	This makes the game unique and help as an indicator for the user.
Field sprite	Bat_wings_down	.jpeg	Pixel art image of a bat with wings down	This makes the game unique and help as an indicator for the user.
Field sprite	Bat_dying	.jpeg	Pixel art image of a bat “dying”	This makes the game unique and help as an indicator for the user.
Field sprite	Bat_falling	.jpeg	Pixel art image of a bat “falling”	This makes the game unique and help as an indicator for the user.

Field sprite	Bullet_image	.jpeg	Pixel art image of a bullet	This is an indicator for the user.
Field sprite	Heart_image	.jpeg	Pixel art image of a heart	This is an indicator for the user.
Field sprite	Ammo_title	.jpeg	Pixel art font of “AMMO” in a white box	This is an indicator for the user.
Field sprite	health_title	.jpeg	Pixel art font of “HEALTH” in a white box	This is an indicator for the user.
Field sprite	score_title	.jpeg	Pixel art font of “SCORE” in a white box	This is an indicator for the user.
Field sprite	Metal_display	.jpeg	Pixel art metal strip to cut of where bats can “fly”	This gives a place to but the shot box, life box and score box.
Field sprite	Forest_bg	.jpeg	Pixel art background of a forest	This makes the game unique.
Field sprite	Mountain_bg	.jpeg	Pixel art background of a mountain range	This makes the game unique.
Field sprite	Hills_bg	.jpeg	Pixel art background of hills	This makes the game unique.
Field sprite	Castle_bg	.jpeg	Pixel art background with a castle in in it	This makes the game unique.
AI stuff				
AI stuff				
AI stuff				

Candidate Name: [redacted] Candidate Number: [redacted]

TEST DATA

Score board

New high score

Test data	Type
(New score) > (scores on score board)	Valid
(New score) < (scores on score board)	Invalid

High score name and validation

Test data	Type
GameBoy32	Valid
12345678910takingawalk	Invalid
54	Invalid

Menu

Player one

Test data	Type
“1”	Valid
“*”	Invalid
“4”	Invalid

Controls

Test data	Type
“c”	Valid
“1”	Invalid
“}”	Invalid

Controls

Escape to menu state

Test data	Type
“escape key”	Valid
“4”	Invalid
“(”	Invalid

Paused

Un-paused

Test data	Type
“p”	Valid
“*”	Invalid
“4”	Invalid

Playing

Flying animation

Test data	Type
Bat spawned within in the parameters of (1250,850)	Valid
Bat spawned on the perimeter of (1250,850)	Borderline
Bat spawned outside the parameters of (1250,850)	Invalid
Bat can “fly” within in the parameters of (1250,850)	Valid
Bat can “fly” on the perimeter of (1250,850)	Borderline
Bat can “fly” outside the parameters of (1250,850)	Invalid

Flying sound

Test data	Type
Bat spawned within in the parameters of (1250,1090)	Valid
Bat spawned outside the parameters of (1250,1090)	Invalid

Dying animation

Test data	Type
“LMC”(left mouse click) on the target	Valid
“RMC”(right mouse click) on the target	Invalid
“LMC”(left mouse click) within the parameter of (1250, 850)	Invalid
“RMC”(right mouse click) within the parameter of (1250, 850)	Invalid

Dying sound

Test data	Type
“LMC”(left mouse click) on the target	Valid
“RMC”(right mouse click) on the target	Invalid
“LMC”(left mouse click) within the parameter (1250,850)	Invalid
“RMC”(right mouse click) within the parameter of (1250,1090)	Invalid

Falling animation

Test data	Type
“LMC”(left mouse click) on the target	Valid
“RMC”(right mouse click) on the target	Invalid
“LMC”(left mouse click) within the parameter of (1250, 850)	Invalid
“RMC”(right mouse click) within the parameter of (1250, 850)	Invalid

Falling sound

Test data	Type
“LMC”(left mouse click) on the target	Valid
“RMC”(right mouse click) on the target	Invalid
“LMC”(left mouse click) within the parameter of (1250,850)	Invalid
“RMC”(right mouse click) within the parameter of (1250,850)	Invalid

Gun shot

Test data	Type
“LMC” (left mouse click) within the parameter of (1250, 610)	Valid
“RMC” (right mouse click) within the parameter of (1250, 610)	Invalid
“LMC”(left mouse click) on the target	Valid
“RMC”(right mouse click) on the target	Invalid
“LMC” (left mouse click) outside the parameter of (1250, 610)	Invalid
“RMC” (right mouse click) outside the parameter of (1250, 610)	Invalid
“LMC” (left mouse click) on the parameter of (1250, 610)	Borderline
“RMC” (right mouse click) on the parameter of (1250, 610)	Borderline

How long the bat flies

Test data	Type
Counter resets after new target spawns	Valid
Counters value is a non-integer/negative value	Invalid

Bat speed

Test data	Type
Score becomes a multiple of 100	Valid
Score is a integer that is non-integer/negative value	Invalid

Bat worth

Test data	Type
“LMC”(left mouse click) on the target	Valid
“RMC”(right mouse click) on the target	Invalid
“LMC”(left mouse click) within the parameter of (1250,850)	Invalid
“RMC”(right mouse click) within the parameter of (1250,850)	Invalid

Points to change the background

Test data	Type
Score becomes a multiple of 100	Valid
Score is a integer that is not a multiple of 100	Invalid

Players score

Test data	Type
Points are added to the players score	Valid
Non-integers/negative values are added to the score	Invalid

Bullet removal

Test data	Type
“LMC”(left mouse click) within the parameter of (1250,850)	Valid
“RMC”(right mouse click) within the parameter of (1250,850)	Invalid

“LMC”(left mouse click) on the target	Valid
“RMC”(right mouse click) on the target	Invalid
“LMC”(left mouse click) outside the parameter of (1250,850)	Invalid
“RMC”(right mouse click) outside the parameter of (1250,850)	Invalid
“LMC”(left mouse click) on the parameter of (1250,850)	Valid
“RMC”(right mouse click) on the parameter of (1250,850)	Invalid

Life removal

Test data	Type
Counter value is 0	Valid
The target disappears from screen	Valid
The target is still on the screen	Invalid
“LMC”(left mouse click) on the target	Invalid

Go to menu state

Test data	Type
“m”	Valid
“4”	Invalid
“(”	Invalid

Pause

Test data	Type
“p”	Valid
“9”	Invalid
“=”	Invalid

AI stuff

Insert AI test tables here

I’m not sure how to tackle this problem yet so I will have to do some more research before I can design the expected inputs.

ACCEPTANCE TESTING

No.	Requirements	Input	Expected outcome
1.	The users final score is added to the score board if it is greater than the score boards lowest score	**Valid:** a final score of 1000 **Invalid:** the string "quickshot2"	**Valid:** the score is placed on the score board next to the user's name **Invalid:** the name "quickshot2" is displayed twice, once as the name and once as the score
2.	The users name can be used for the score board if the name is greater than 2 characters and less than 21 characters	**Valid:** the string "duck321" **Invalid:** the string "ya"	**Valid:** the name "duck321" is placed next to the users final score on the score board **Invalid:** the name "ya" is placed next to the users final score on the score board

3.	The player gains 10 points when a bat is shot	**Valid:** left mouse click on the target when the bullet variable isn't 0 **Invalid:** left mouse click within the parameter of (1250,850) when the bullet variable isn't 0	**Valid:** 10 points are added to the users score variable **Invalid:** 10 points are taken away from the users score variable
4.	Score box in the bottom right of the screen.	**N/A**	The score box is in the bottom right of the screen
5.	Score board is shown at the end of each game	**N/A**	Score board is shown at the end of each game
6.	Three shots per mini round	**Valid:** the bullet variable will be set to 3	**Valid:** when the game is initialised the bullet variable is set to 3 **Invalid:** when the game is initialised the bullet variable is set to a number other that 3
7.	Ammo disappears as the player shoots at the bats until there is no ammo left so the player can no longer shoot.	**Valid:** left mouse click within the parameter of (1250,850) when the bullet variable isn't 0 **Invalid:** left mouse clicks outside the parameter of (1250,850) when the bullet variable isn't 0	**Valid:** a bullet is removed from the ammo box **Invalid:** no bullets are removed from the ammo box
8.	Shot box, with three bits of ammo, in the bottom left of the screen.	**N/A**	The ammo box Is in the bottom left corner
9.	Three lives in total	**Valid:** the life variable will be set to 3 **Invalid:** the life variable will be set to a number other than 3	**Valid:** when the game is initialized the life, variable is set to 3 **Invalid:** when the game is initialized the life, variable is set to a number other that 3
10.	If a bat "escapes" the user loses a life	**Valid:** a bat has "escaped"	**Valid:** 1 is removed from the life variable
11.	Life box, has three lives inside, in the middle of the bottom of the screen.	**N/A**	The ammo box is in the bottom left corner
12.	A counter is started when the bat starts to fly around. It	**Valid:** Counter resets after new target spawns	**Valid:** the counter decrease until 0 **Invalid:** the counter will become negative

	decreases until 0. This will act as a timer	**Invalid:** Counters value is a non-integer/negative value	
13.	The bat flies around until score counter becomes 0 then the sprite disappears/ "escapes"	**Valid:** the counter has decrease to 0 **Invalid:** the counter becomes negative	**Valid:** the bat "escapes" **Invalid:** the bat will just keep flying around with no end
14.	Menu with the title and three options, being the one and two player modes and the game controls option.	**N/A**	The menu will have three options, being the one and two player modes and the game controls option.
15.	As game goes on the bats become faster. This happens every multiple of 100	**Valid:** Score becomes a multiple of 100 **Invalid:** Score is a integer that is non-integer/negative value	**Valid:** the time between each line of execution is decreased **Invalid:** the speed at which the bat moves doesn't change
16.	All sprites and backgrounds will be in a pixel art style	**N/A**	All sprites and backgrounds will be in a pixel art style
17.	Sound effect for the shooting. This will occur when left mouse button is clicked	**Valid:** left mouse click within the parameter of (1250, 1090) when the bullet variable isn't 0 **Invalid:** left mouse click within the parameter of (1250, 1090) when the bullet variable is 0 **Borderline:** left mouse click on the perimeter of (1250, 850) when the bullet variable isn't 0	**Valid:** a gunshot sound is played **Invalid:** no sound is played **Borderline:** a gunshot sound is played
18.	Sound effect for the bat being "killed". A hurling sound is played	**Valid:** the bat has been left mouse clicked on when the shot variable isn't 0 **Invalid:** the bat has been clicked on when the shot variable is 0	**Valid:** A hurling sound is played **Invalid:** No sound is played
19.	Sound effect for the bat flying around	**Valid:** the bat has been spawned within the game screen (1250,850)	**Valid:** a flapping sound is played when the bat is "flying" around the screen **Invalid:** hurling sounds are played on loop

		Invalid: the bat has been left mouse clicked on when the shot variable isn't 0	
20.	Sound effect for the bat falling	**Valid:** bat hurling sound has been played **Invalid:** "m" has been pressed	**Valid:** a whistling sound is played **Invalid:** the game state changes from playing to the menu state
21.	The animation for the bat being "killed". A hurling sound is played	**Valid:** the bat has been left mouse clicked on when the shot variable isn't 0 **Invalid:** the bat has been clicked on when the shot variable is 0	**Valid:** A "dying" bat animation is played **Invalid:** No animation is played
22.	The animation for the bat flying around	**Valid:** the bat has been spawned within the game screen (1250,850) or the bat can "fly" within in the parameters of (1250,850) **Invalid:** Bat spawned outside the parameters of (1250,850) or Bat can "fly" outside the parameters of (1250,850) **Borderline:** Bat spawned on the perimeter of (1250,850) or Bat can "fly" on the perimeter of (1250,850)	**Valid:** a flapping animation is played when the bat is "flying" around the screen **Invalid:** the bat doesn't appear on screen **Borderline:** a flapping animation is played when the bat is "flying" around the screen
23.	The animation for the bat falling	**Valid:** bat "dying" animation has been played **Invalid:** "3" has been pressed	**Valid:** bat "falling" animation is played **Invalid:** no animation is played
24.	Press the 'm' key to access the menu	**Valid:** when in the playing state "m" is pressed **Invalid:** when in the playing state "r" is pressed	**Valid:** the game state will change from playing to the menu state **Invalid:** the game will continue in the playing state

25.	Press the 'p' key to enter the paused game state	**Valid:** when in the playing state "p" is pressed **Invalid:** when in the playing state "y" is pressed	**Valid:** the game state will change from playing to the paused state **Invalid:** the game will continue in the playing state
26.	Press the 'p' key to re-enter the playing game state	**Valid:** when in the paused state "p" is pressed **Invalid:** when in the playing state "m" is pressed	**Valid:** the game state will change from paused to the playing state **Invalid:** the game state will change from paused to the menu state
27.	Use the mouse to click on the moving targets in the game with left click.	**Valid:** LMC on the bat when there is still ammo in the ammo box **Invalid:** enter is pressed to "shoot"	**Valid:** when the user clicks on the bat when they have ammo it is removed of screen **Invalid:** a gunshot sound is played when enter is pressed
28.	Use the mouse to navigate on the menu screen	**Valid:** the user is able to left mouse click on the different game options to select **Invalid:** enter is pressed to select a game option	**Valid:** the user can select different game options using the mouse **Invalid:** nothing will happen
29.	Back ground will change every time the score box's value increase by 100 points. These back grounds will be of different locations	**Valid:** Score becomes a multiple of 100 **Invalid:** Score is a integer that is not a multiple of 100	**Valid:** the background changes according to the multiple of 100 **Invalid:** the background doesn't change
30.	A splash screen is displayed for a while then changes to the menu	**N/A**	A splash screen is displayed for a while then changes to the menu

Insert AI acceptance testing here

I'm not sure how to tackle this problem yet so I will have to do some more research before I can design the acceptance testing.

DEVELOPING THE CODED SOLUTION ("THE DEVELOPMENT STORY")

This is section of the project will be treated like a diary. It will be a series of dates followed by what I completed in this day.

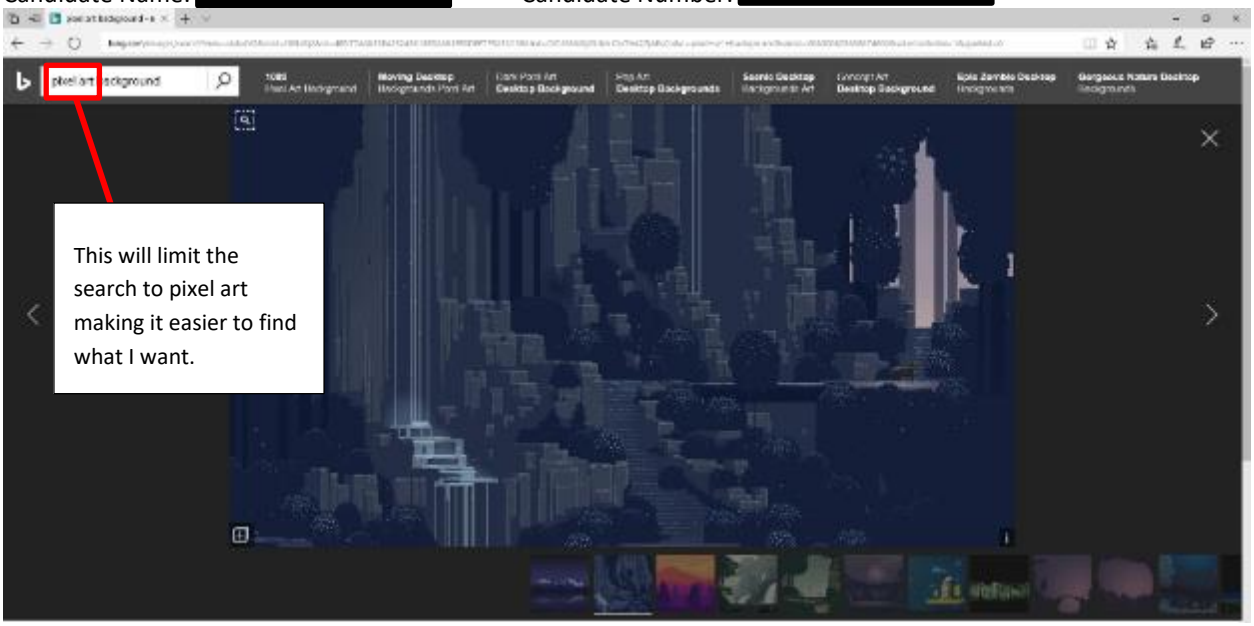
SPRITE CREATION

DATE: 06/12/17

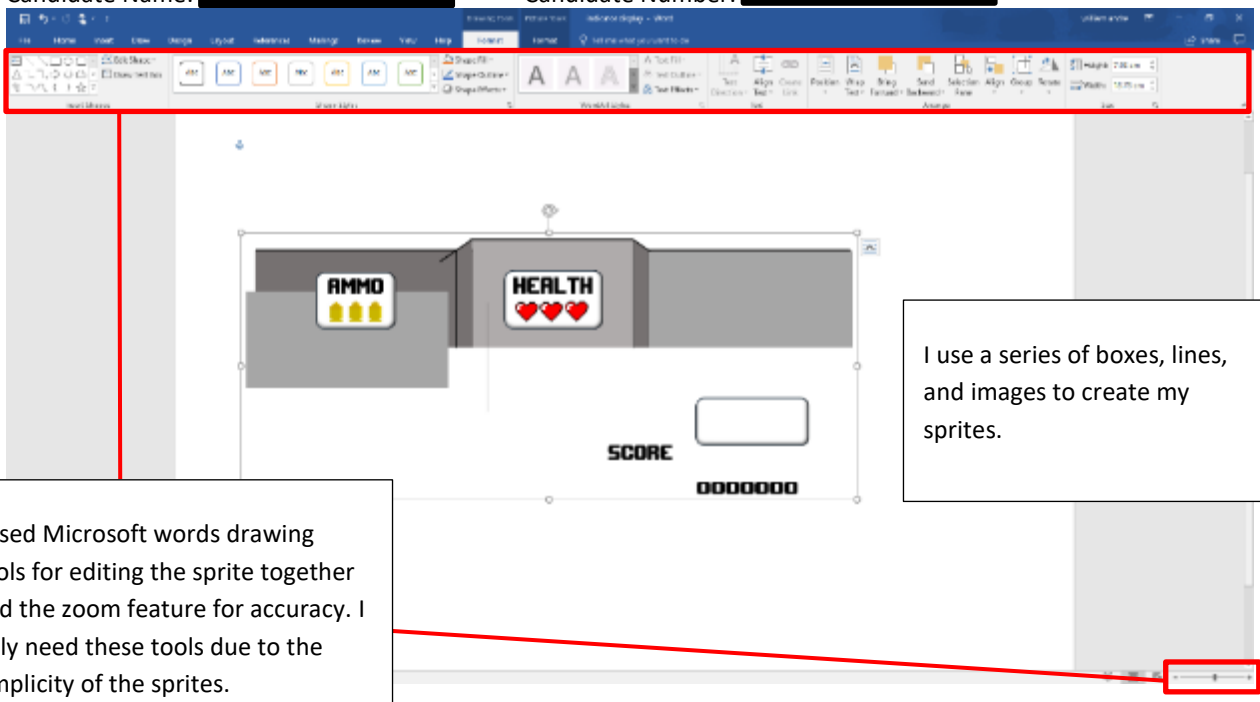
I first decided to create the sprites required for the coded solution. I started off by using an internet browser to search for relevant pixel styled images e.g. pixel art bat, forest background, bullet, etc. Next, I used Microsoft word to edit together the selected images from the internet browser search to create my sprites e.g. the indicator display; this was done due to the simplicity of the sprites meaning no specialist software was required for editing. Once this was done all I had to do was re-size the sprites, so they would all be in the correct scaling for my game, I did this using Microsoft paint because it had a system that would easily allow you to re-size images down to the resolution of one pixel giving me a reasonable amount of creative freedom.

The following screen shots are the different stages of this process.

Candidate Name: [redacted] Candidate Number: [redacted]

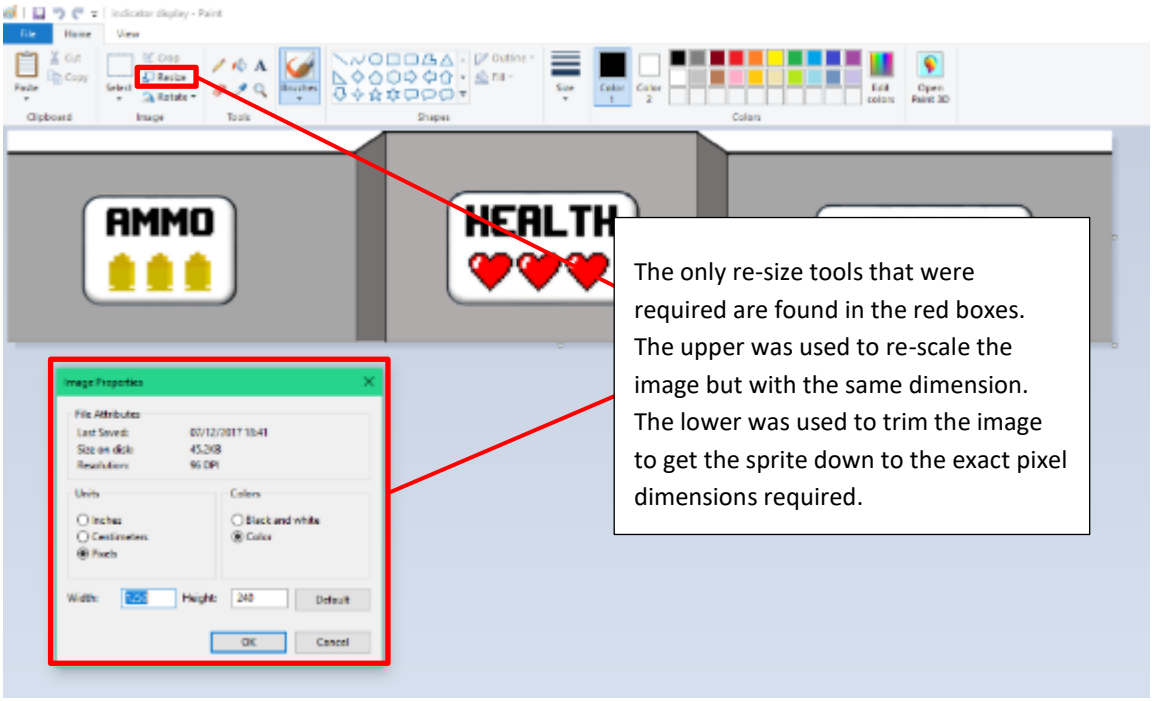


Candidate Name: [redacted] Candidate Number: [redacted]



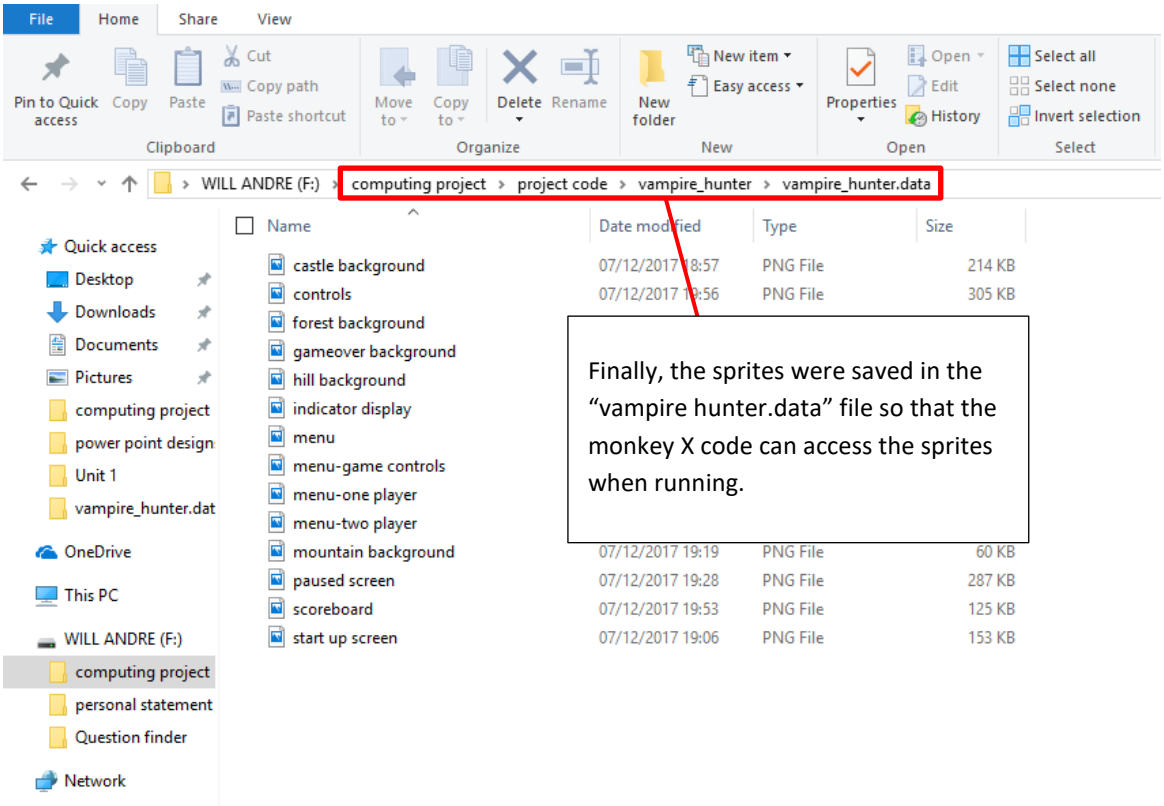
I use a series of boxes, lines, and images to create my sprites.

I used Microsoft words drawing tools for editing the sprite together and the zoom feature for accuracy. I only need these tools due to the simplicity of the sprites.



The only re-size tools that were required are found in the red boxes. The upper was used to re-scale the image but with the same dimension. The lower was used to trim the image to get the sprite down to the exact pixel dimensions required.

Candidate Name: [redacted] Candidate Number: [redacted]



Code deconstruction

CODE DECONSTRUCTION

DATE: 08/12/17

This section of my solution will be following the development of the game code. The development will be shown in a series of screen shots with annotations of the code that I have written along with an appropriate amount of testing to ensure that what I have written works.

To start off my code development I have researched other code to get an idea of what techniques to use when coding the solutions to the individual modules. So, far I have found an internet resource created by "Craig n Dave" that provides basic tutorials for creating simple games in Monkey X.

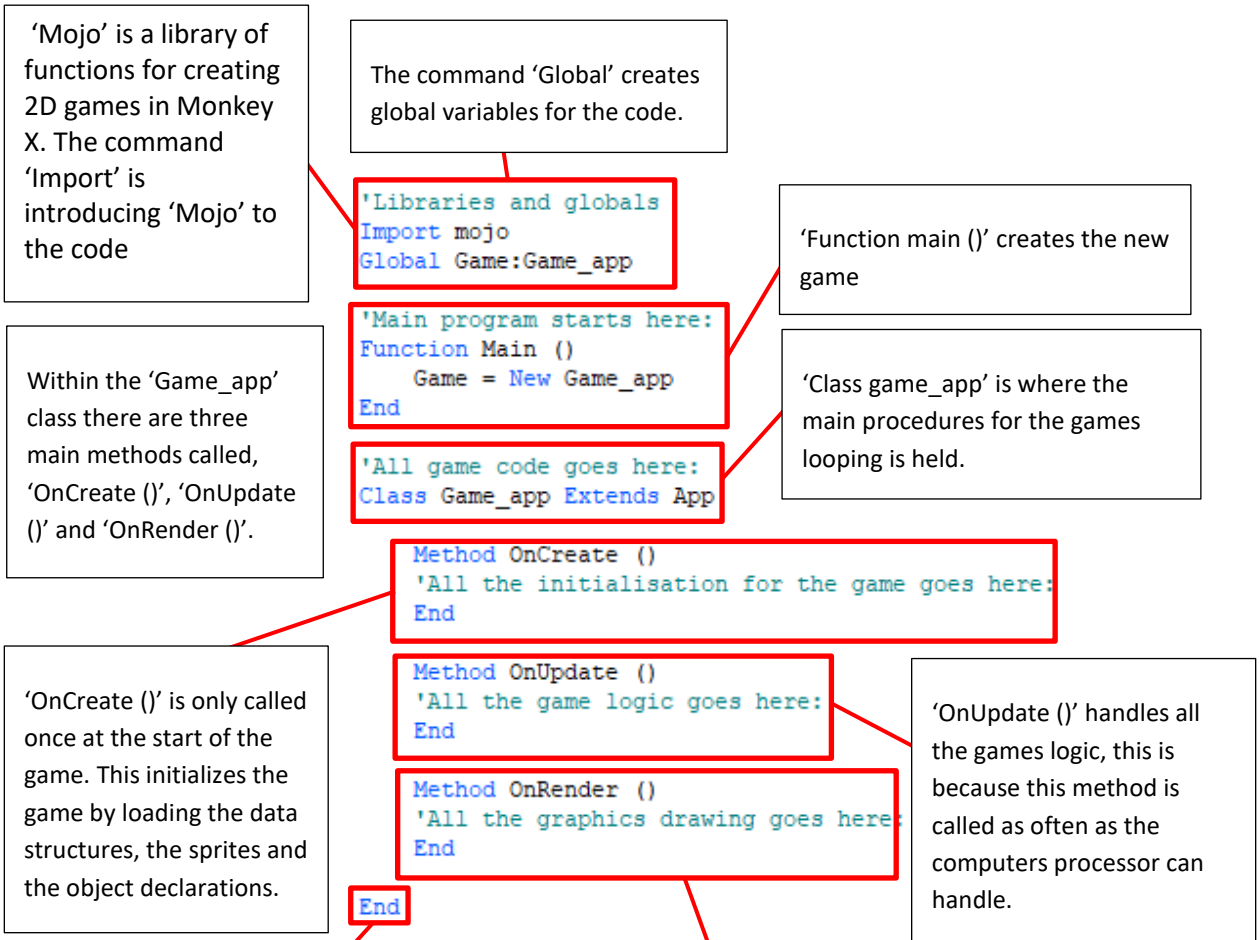
The internet resource:
file:///T:/Computing/A%20Level%20Computing/Year%2013%20Project/Monkey-X/website/02-tutorials.html

Candidate Name: [redacted] Candidate Number: [redacted]

Structure

This is the basic framework for the game in Monkey X

The green code with a single quote at the beginning of the line is quote



This code for the structure will be very useful to me as this will allow me to create the back bone of my project and then build upon it. The way the structure is laid out will keep my code neat and easy to follow.

Game States

```
'Libraries and globals
Import mojo
Global Game:Game_app

'Main program starts here:
Function Main ()
    Game = New Game_app
End

'All game code goes here:
Class Game_app Extends App

Field menu:Image
Global GameState:String = "MENU"

    Method OnCreate ()
    'All the initialisation for the game goes here:
        SetUpdateRate 60
        menu = LoadImage ("menu.png")
    End

    Method OnUpdate ()
    'All the game logic goes here:
        Select GameState
            Case "MENU"
                If KeyHit (KEY_SPACE) Then GameState="PLAYING"
            Case "PLAYING"
                If KeyHit (KEY_ESCAPE) Then GameState="MENU"
        End
    End

    Method OnRender ()
    'All the graphics drawing goes here:
        Select GameState
            Case "MENU"
                DrawImage menu, 0,0
            Case "PLAYING"
                Cls 123, 123, 123
        End
    End

End
```

The field command is used to declare the variable menu with the data type image. This is going to hold the image.

'Global' is the keyword to declare a global variable that can be accessed anywhere in the program.

A global variable is declared, this can be accessed from anywhere in the code, called 'GameState' as a string and its value has been set to "MENU". This is now the variable that will hold the game state.

Using a combination of if statements, select case and key presses allows the user to switch between the game states playing and menu. E.g. by pressing the space when in the menu state the game state will change to the playing state.

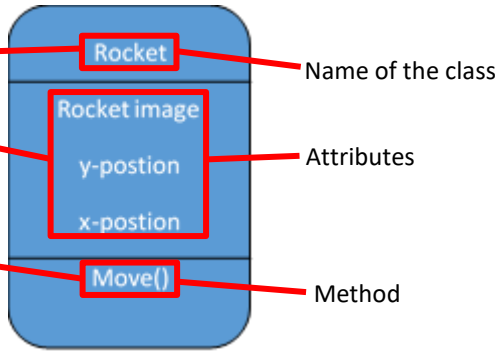
When the game state changes this section of code will change what the screen looks like. For example, when the game state is the menu game state a menu image is drawn.

I will be creating similar code to this however more game states will be implemented. Furthermore, I will use a wider range of keys to allow the user to change through game states. This has also shown me how to load images into the game which I could also use when loading my sprites into the game.

Candidate Name: [redacted] Candidate Number: [redacted]

Creating a class

```
Class Rocket
    Field sprite:Image = LoadImage ("player.png")
    Field x:Float = 300
    Field y:Float = 420
    Method Move(x_distance:Int)
        x+=x_distance
        If x<0 Then x=0
        If x>590 Then x=590
    End
End
```



I will be following the same structure when creating the classes for my bats, but the methods and attributes will be exchanged with appropriate code.

DATE: 02/01/18

Moving object

This is just the speed of the object

```
For Local missile:=Eachin missile_collection
    missile.Move(5)
    If missile.y<10 Then missile_collection.Remove missile
Next
```

If the object is below a certain value on the y axis the '.Remove' command is used to remove the object from the screen

The important bits of this section of code is being able to set the speed of the object and removing the object go the screen. I could use the code for changing the speed of the object to increase the speed of the bat making the game harder and use the '.remove' command to remove the bat of screen when it has been 'shot' by the user.

Candidate Name: [redacted] Candidate Number: [redacted]

Mouse clicking on object

This is an 'if statement' triggered by the user pressing the left mouse button.

The coordinates of the mouse are stored as local variables, x and y. The new local variables are both divided by 100 because the object the user is clicking on is 100 pixels wide and tall.

```
If KeyHit (KEY_LMB) Then
Local x:Int = (MouseX/100)
Local y:Int = (MouseY/100)
Local square:Int = y*5+x
```

This converts the x and y coordinates into a square which can be used with a 1D array.

I will use code similar to this so that I have a mechanism allowing me to click on the moving targets, bats, to 'kill' them. When the user clicks on the screen the coordinates are compared to the location of the bat. If the coordinates match then the bat will be 'killed'.

Creating two dimensional arrays

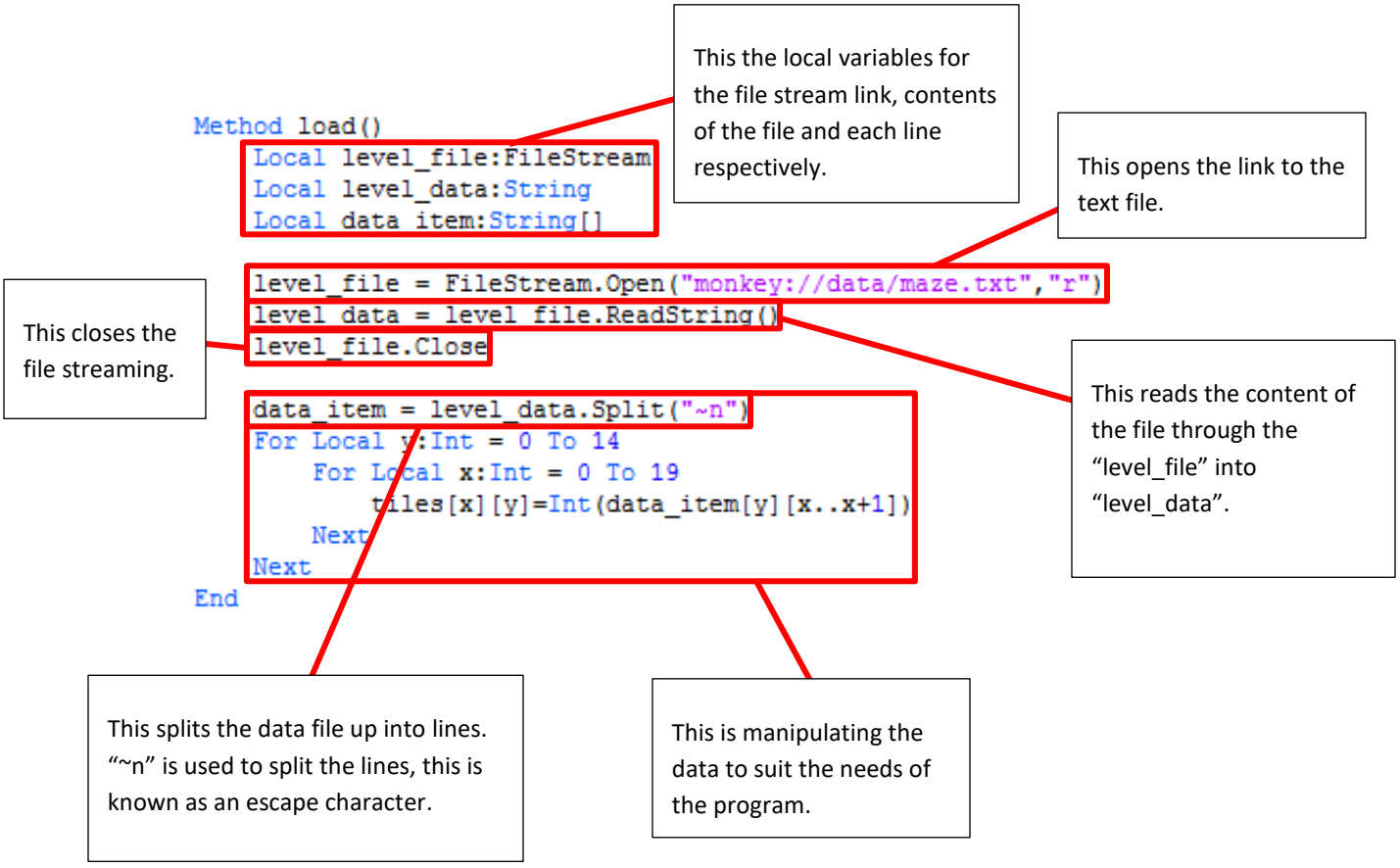
```
Local stuff:[][]
```

The number of open and closed brackets determines the number of dimensions the array will contain.

I will use the array to create a high score table and movement location for the AI bats.

Candidate Name: [redacted] Candidate Number: [redacted]

File streaming; could be used for high score table



When I close my game I don't want the high scores to be lost so they will be saved in a text file. During the game the high score table is read from the text file and stored as an array. At the end of each game the array is written into the text file with all the new changes. Hence, all changes to the high scores in the high score table are saved.

Candidate Name: [redacted] Candidate Number: [redacted]

GAME STATES

DATE: 06/01/18

Next step:

I am going to start coding. As suggested by the 'Craig and Dave' I will start of the coded solution by setting up the structure of the code first. Then, I will code the game states that can be navigated through by the user.

This is the main structure of the code.

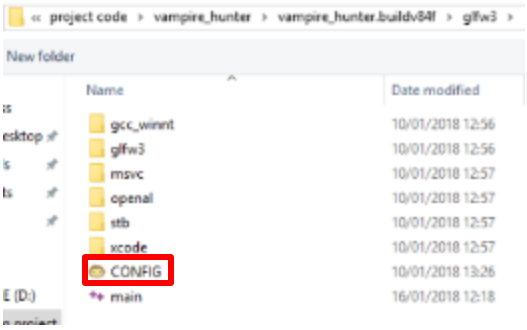
'OnCreate ()' is only called once at the start of the game. This initializes the game by loading the data structures, the sprites and the object declarations.

'OnUpdate ()' handles all the games logic, this is called as often as the computers processor can handle to check.

'OnRender' does all the screen drawing for the game, this is also called as often as the processor can handle.

Candidate Name: [redacted] Candidate Number: [redacted]

Here I opened the 'vampire_hunter.buildv84f' and accessed the 'CONFIG.monkey' file. Within this file I changed the dimensions of the window which will display the game screen to 1250 pixels wide and 850 pixels high then saved the changes. This is because I design my sprites and background to match these dimensions. Now when I run the program the dimensions of the window that pops up will be 1250 by 850 pixels.




```
#GLFW_USE_MINGW=True

#GLFW_WINDOW_TITLE="Monkey Game"
#GLFW_WINDOW_WIDTH=1250
#GLFW_WINDOW_HEIGHT=850
#GLFW_WINDOW_SAMPLES=0
#GLFW_WINDOW_RESIZABLE=False
#GLFW_WINDOW_DECORATED=True
#GLFW_WINDOW_FLOATING=False
#GLFW_WINDOW_FULLSCREEN=False
```

What's being tested?	input	justification	Expected outcome	Actual outcome
Did the dimensions change to the required dimensions	Width-1250(pixels) Height-850(pixels)	Larger playable area for the user to move there mouse	The window on which the game is run has a width of 1250(pixels) and a height 850(pixels)	The width and height of the window are 1250(pixels) and 850(pixels) respectively

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 08/01/18

My next step when building upon the back bone was to create the game states which the user can navigate between and the links between them. I had a bit of difficulty starting of as I had to get used to the new programming language. I was confused to why some command words where being marked as syntax errors but then realized this was due to the fact the language is case sensitive.

```
'All game code goes here:
Class Game_app Extends App

    Field menu:Image
    Field controls:Image
    Field paused:Image
    Field gameover:Image
    Global gamestate:String = "MENU"

    Method OnCreate ()
    'All the initialisation for the game goes here:
        SetUpdateRate 60
        menu = LoadImage ("menu.png")


    End

    Method OnUpdate ()
    'All the game logic goes here:
    'This is the code to allow the user to change between gamestates
        Select gamestate
            Case "MENU"
                If KeyHit (KEY_C) Then gamestate="CONTROLS"
                If KeyHit (KEY_1) Then gamestate="PLAYING"
            Case "PLAYING"
                If KeyHit (KEY_P) Then gamestate="PAUSED"
                If KeyHit (KEY_M) Then gamestate="MENU"
            Case "CONTROLS"
                If KeyHit (KEY_ESCAPE) Then gamestate="MENU"
            Case "PAUSED"
                If KeyHit (KEY_U) Then gamestate="PLAYING"
        END
    End
```

Here I am declaring the variables that will be used to load the images for the game states.

Here I created a game state variable that will allow the change in game states to occur. I first set it to the menu game state.

This sets the frame rate of the game to 60 fps.

This loads the image for the menu as the game state was initially set to menu.

This section of code will allow the user to switch between game states using keys, this may be changed in the future so that the user can click on the options in the menu to change between game states. I used a select case instead of an 'if' 'statement

```
    Method OnRender()
'   All the graphics drawing goes here:
        Select gamestate
            Case "MENU"
                DrawImage menu, 0,0
            Case "PLAYING"
                Cls 123, 123, 123
            Case "CONTROLS"
                controls = LoadImage ("controls.png")
                DrawImage controls, 0,0
            Case "PAUSED"
                paused = LoadImage ("paused.png")
                DrawImage paused, 0,0
        End

    End
end
```

This is drawing the images for the games states. Depending on what the 'gamestate' variable is set to a certain image is loaded up.

What's being tested?	input	justification	Expected outcome	Actual outcome
Does the game state change from menu to playing	Key '1'	This allows the user to select the player one game mode from the menu	A grey screen will be loaded	A grey screen was loaded

What's being tested?	input	justification	Expected outcome	Actual outcome
Does the game state change from playing to menu	Key 'm'	This allows the use to return to the menu to select different options	The menu image will load	A menu image was loaded

What's being tested?	input	justification	Expected outcome	Actual outcome
Does the game state change from menu to control	Key 'c'	This lets the users access the control descriptions from the menu	The control descriptions will be loaded up as an image	The control descriptions image is loaded up

What's being tested?	input	justification	Expected outcome	Actual outcome
Does the game state change from control to menu	Key 'Esc'	This is allowing the user to exit back to the menu	The menu image will load	A menu image was loaded

What's being tested?	input	justification	Expected outcome	Actual outcome
Does the game state change from playing to paused	Key 'p'	This gives a display for the paused game	The pause image will load	A pause image was loaded

What's being tested?	input	justification	Expected outcome	Actual outcome
Does the game state change from paused to playing	Key 'p'	This will resume the game	A grey screen will be loaded	A grey screen was loaded

TESTING INPUT DATA

Menu

Player one

Test data	Expected	Actual
“1”	Valid	Valid
“*”	Invalid	Invalid
“4”	Invalid	Invalid

Controls

Test data	Expected	Actual
“c”	Valid	Valid
“1”	Invalid	Invalid
“}”	Invalid	Invalid

Controls

Escape to menu state

Test data	Expected	Actual
“escape key”	Valid	Valid
“3”	Invalid	Invalid
“(”	Invalid	Invalid

Playing state

Go to menu state

Test data	Expected	Actual
“m”	Valid	Valid
“4”	Invalid	Invalid
“(”	Invalid	Invalid

Pause

Test data	Expected	Actual
“p”	Valid	Valid
“9”	Invalid	Invalid
“=”	Invalid	Invalid

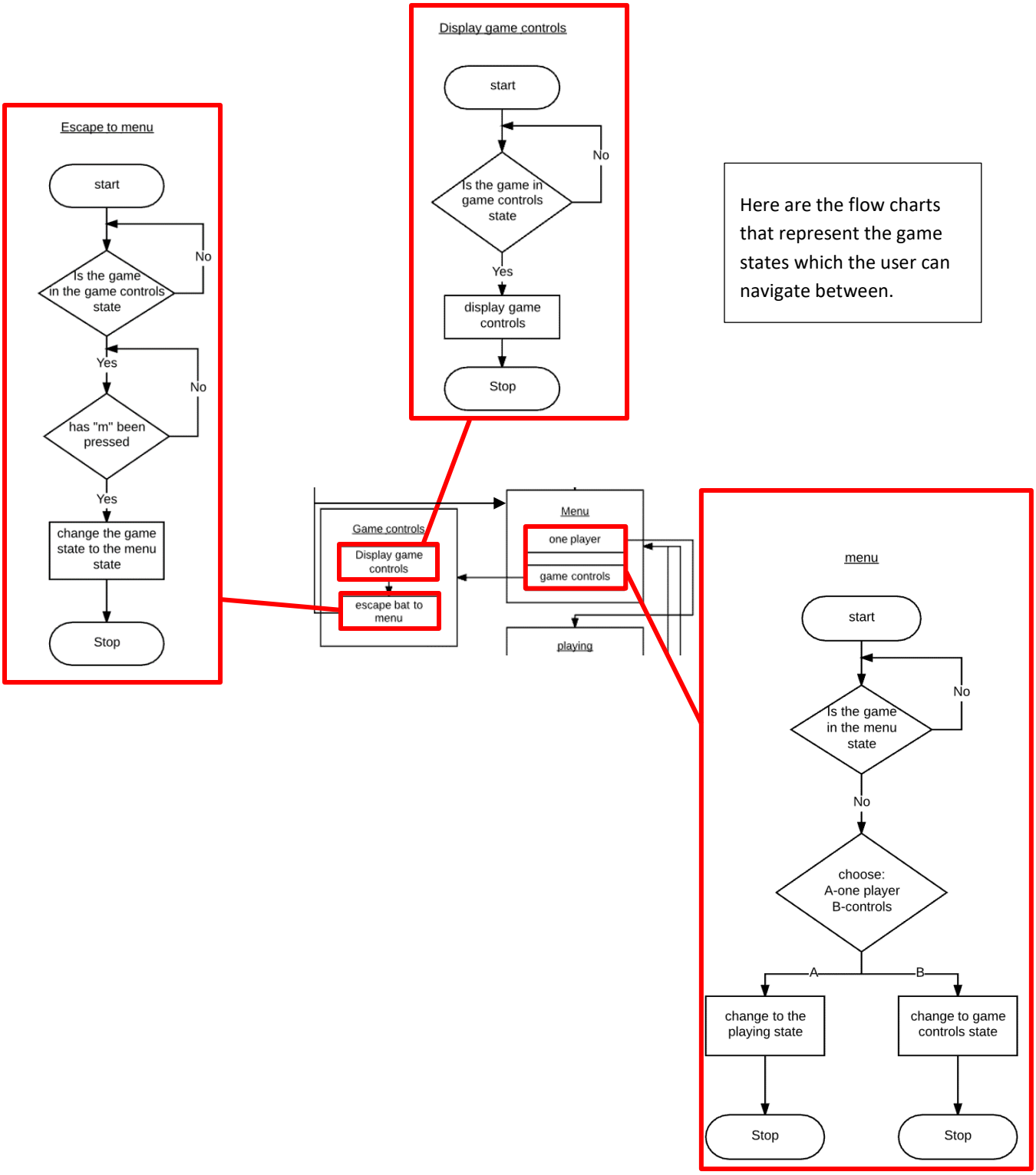
Paused state

Un-paused

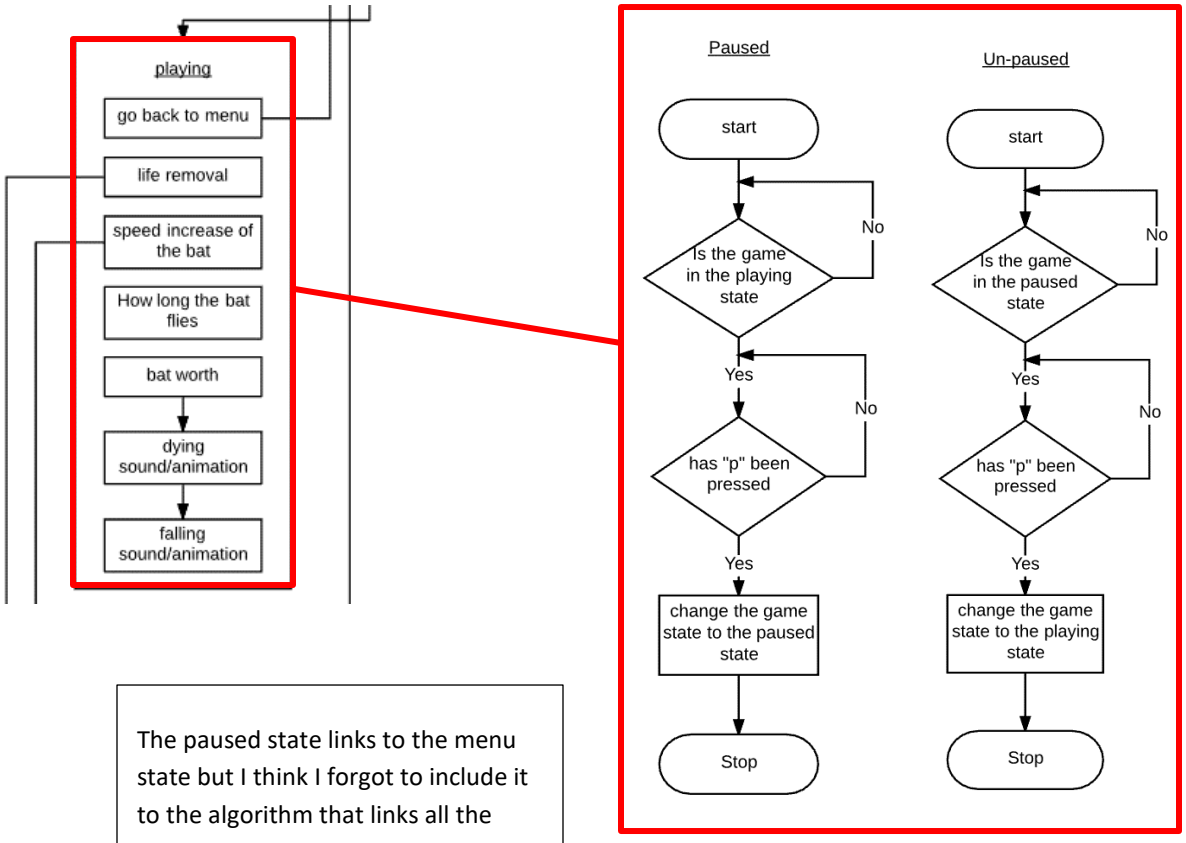
Test data	Type
“p”	Valid
“”	Invalid
“4”	Invalid

Candidate Name: [redacted] Candidate Number: [redacted]

FLOW CHART

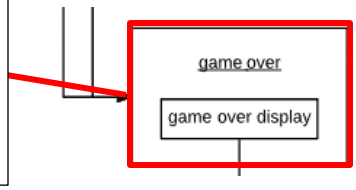


Here are the flow charts that represent the game states which the user can navigate between.



The paused state links to the menu state but I think I forgot to include it to the algorithm that links all the game states to it.

This game states are effected by the user's game play so I will implement this later on in my development stage.



INTERVIEW: 12/01/18

Here are all the requirements that were looked at this stage of the development.

No.	Requirements	Justification
13.	Menu with the title and three options, being the one player mode, two player mode and controls description.	This will create a link between the one player and two player modes
23.	Press the 'm' key to access the menu	This will allow the user to easily switch between the one and two player game modes
24.	Press the 'p' key to enter the paused game state	This makes the game more convenient as the players have the freedom to stop playing whenever and continue straight away with no difficulty
25.	Press the 'p' key to re-enter the playing game state	Allow the user to resume playing the game
26.	Press the escape key to re-enter the menu game state	This will allow the user to return to the menu state from the control state
28.	Use the mouse to navigate on the menu screen	This was the decided as a control for navigating around the menu as it is already used else were in the game. This is done for continuity purposes.

Me:
I have completed most of the requirements I wanted to complete for this stage of my development. I have successfully implemented the game states which the user can manually switch between. This consists of being able to access the playing state from the menu using the '1' key; Within the playing state using the 'p' key the user can pause and resume the game. The 'm' key in the playing state allows you to return to the menu. Furthermore, pressing the 'c' key opens the controls description and to exit you press the escape key. I have also created and implemented all the screen designs for the game states except for the playing state.
We did agree that we should include the capability that the user can navigate through the menu using the mouse. But, due to time restraints and my unfamiliarity of how to implement this I decided to use keys to navigate the menu instead.

Ben Newton:

I am happy with how the game state system works, and I do think it is better to just navigate the menu with keys rather that to navigate the menu with a mouse as it is not crucial, and we have limited time. I would like to see I developed playing state screen for the next prototype but otherwise the progress you have made is good.

Next step:

Following the interview, I have decided that for my next steps in developing my code I will implement a splash screen and put together my playing state screen with its elements e.g. health, ammo and score.

SPLASH SCREEN

DATE: 16/01/18

For my splash screen to work as needed it needs to appear for a certain amount of time and then change to the menu screen.

To do this I thought I could change the initial game state to the 'start_up' game state and then load the splash screen up, this would then be kept of the screen using a sleep function. After this the game state would just switch to the 'menu' game state.

So, first I got the startup image to load first instead of the menu image.

```
'All game code goes here:
Class Game_app Extends App

    Field menu:Image
    Field controls:Image
    Field paused:Image
    Field gameover:Image
    Field start_up:Image
    Global gamestate:String = "START_UP"

    Method OnCreate ()
    'All the initialisation for the game goes here:
        SetUpdateRate 60
        start_up = LoadImage ("start_up.png")
```

This is declaring the startup image

This is changing the game state to the 'start_up' game state.

When the game is initialized the 'start_up' image is loaded.

Now I just need the code to pause to display the image for a short time and then switch to the menu game state. This seemed simple. However, I didn't know what the sleep function was for monkey so I started searching for the function. I couldn't find this function so I decided to use a 'for' loop instead.

Whenever the game state is 'start_up' a 'for' loop would run which would just count to 90000 to "pause" the running of the code and then change to the 'menu' game state.

```
Method OnCreate ()
'All the initialisation for the game goes here:
    SetUpdateRate 60
    start_up = LoadImage ("start_up.png")
End

Method OnUpdate ()
'All the game logic goes here:
'This is the code to allow the user to change between gamestates
    Select gamestate
        Case "START_UP"
            For Local x:Int = 0 To 90000
                x = x + 1
            Next
            gamestate = "MENU"
```

What's being tested?	input	justification	Expected outcome	Actual outcome
The splash screen is displayed for roughly 3 seconds	N/A	This gives the user enough time to take in the splash screen	The splash screen is displayed for 3 seconds	The splash screen was displayed for 1 second.

The only issue with this is that the game wouldn't pause long enough. So, I tried to make the value of x count to a higher number.

```
Method OnUpdate ()
'All the game logic goes here:
'This is the code to allow the user to change between gamestates
    Select gamestate
        Case "START UP"
            For Local x:Int = 0 To 900000000000000000000000000
                x = x + 1
            Next
            gamestate = "MENU"
```

What's being tested?	input	justification	Expected outcome	Actual outcome
The splash screen is displayed for roughly 3 seconds	N/A	This gives the user enough time to take in the splash screen	The splash screen is displayed for 3 seconds	An error is indicated

```
Console
gcc -O0 -Wno-free-nonheap-object -I../glfw3/include -I../glfw3/src -I../openal/include -I../stb -I../zlib-1.2.8 -I../lpng1610 -c -o
build/Debug/stb_vorbis.o ../stb/stb_vorbis.c
gcc  -O0 -Wno-free-nonheap-object -I../glfw3/include -I../glfw3/src -I../openal/include -I../stb -I../zlib-1.2.8 -I../lpng1610 -c -o
build/Debug/stb_image.o ../stb/stb_image.c
g++ -O0 -Wno-free-nonheap-object -I../glfw3/include -I../glfw3/src -I../openal/include -I../stb -I../zlib-1.2.8 -I../lpng1610  -c -o
build/Debug/main.o ../main.cpp
../main.cpp:4730:22: warning: integer constant is too large for its type [enabled by default]
  for(int t_x=0;t_x<=900000000000000000000000;t_x=t_x+1){
                    ^
g++ -Wl,--subsystem,windows -L../openal/libs/Win32 -L../openal/libs/Win64 -o Debug/MonkeyGame build/Debug/context.o build/Debug/init.o
build/Debug/input.o build/Debug/monitor.o build/Debug/wgl_context.o build/Debug/win32_init.o build/Debug/win32_monitor.o
build/Debug/win32_time.o build/Debug/win32_tls.o build/Debug/win32_window.o build/Debug/window.o build/Debug/winmm_joystick.o
build/Debug/stb_vorbis.o build/Debug/stb_image.o build/Debug/main.o -lcomdlg32 -lgdi32 -lopengl32 -lOpenAL32 -lws2_32
Done.
```

The only problem with this was that the integer was too big. To solve this problem I decide to use a nested loop because this would allow me to keep the numbers relatively small but the loop would take longer to loop through.

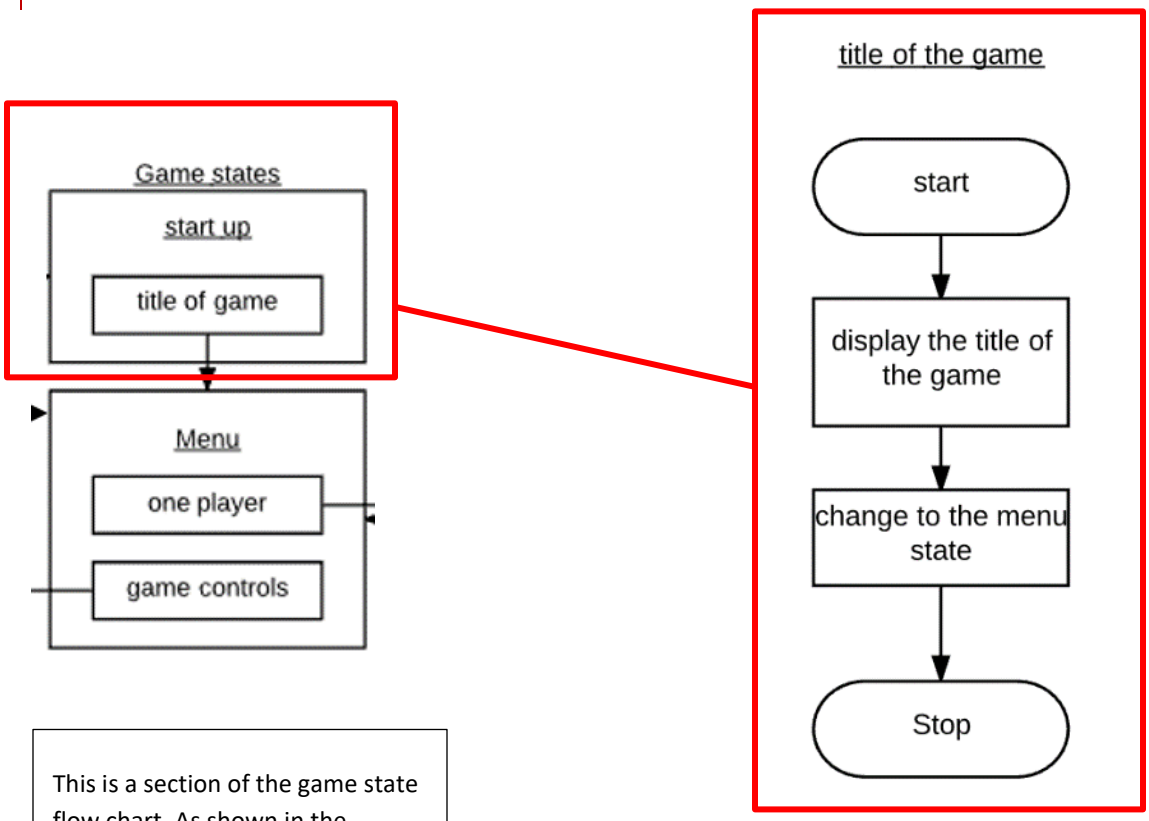
```
Method OnCreate ()
'All the initialisation for the game goes here:
    SetUpdateRate 60
    start_up = LoadImage ("start_up.png")
End

Method OnUpdate ()
'All the game logic goes here:
'This is the code to allow the user to change between gamestates
    Select gamestate
        Case "START UP"
            For Local x:Int = 0 To 90000
                x = x + 1
                For Local y:Int = 0 To 20000
                    y = y + 1
                Next
            Next
            gamestate = "MENU"
```

The nested 'for' loop

This is changing the game state from the 'start_up' game state to the 'menu' game state.

What's being tested?	input	justification	Expected outcome	Actual outcome
The splash screen is displayed for roughly 3 seconds	N/A	This gives the user enough time to take in the splash screen	The splash screen is displayed for 3 seconds	The splash screen is displayed for 3 seconds

FLOW CHART



This is a section of the game state flow chart. As shown in the diagram it has no link other than to the menu game state as this will only occur once in the game.

PLAYING STATE SCREEN

DATE: 22/01/18

I realised that I didn't have to load the sprites into the game directly before they were drawn to the screen in the 'on render' method and that I could load all the sprites that aren't linked to classes in the 'on create' method. This will also help keep my code tidy and easier to follow.

```
Method OnCreate ()
'All the initialisation for the game goes here:
    'frame rate at which the game runs
    SetUpdateRate 60
    'game state images
    start_up = LoadImage ("start_up.png")
    paused = LoadImage ("paused.png")
    controls = LoadImage ("controls.png")
    menu = LoadImage ("menu.png")
```

Next, I loaded all the different possible backgrounds in for the different levels of difficulties of the game

```
Method OnCreate ()
'All the initialisation for the game goes here:
    'frame rate at which the game runs
    SetUpdateRate 60
    'game state images
    start_up = LoadImage ("start_up.png")
    paused = LoadImage ("paused.png")
    controls = LoadImage ("controls.png")
    menu = LoadImage ("menu.png")
    'diffrent backgrounds for playing state
    backf = LoadImage ("forest background.png")
    backc = LoadImage ("castle background.png")
    backh = LoadImage ("hill background.png")
    backm = LoadImage ("mountain background.png")
```

Temporally, I have only loaded in the forest background so that I am able to position the heart, bullet and score sprites correctly.

```
Method OnRender()
All the graphics drawing goes here:
    Select gamestate
    Case "START_UP"
            DrawImage start_up, 0,0
        Case "MENU"
            DrawImage menu, 0,0
        Case "PLAYING"
            DrawImage backf, 0,0
```

Now I wanted to code the screen elements. I first started with the players health. I gave each of the three hearts their own classes as then I could remove them specifically according to how much health the play has. Then, I declared them as images and using the 'DrawImage' command in the 'on render' method to draw them on screen.

```
'*********************************************
Class heart1

Field sprite:Image = LoadImage ("heart.png")
Field x:Int = 60
Field y:Int = 60
End
'*********************************************
Class heart2

Field sprite:Image = LoadImage ("heart.png")
Field x:Int = 60
Field y:Int = 90

End
'*********************************************
Class heart3

Field sprite:Image = LoadImage ("heart.png")
Field x:Int = 60
Field y:Int = 150

End
```

These are the positions of all the hearts. They have not been lined up correctly yet as I first want to make sure they load properly.

This is loading the heart image for each class.

```
Class Game_app Extends App
    'declaring images for game states
    Field menu:Image
    Field controls:Image
    Field paused:Image
    Field gameover:Image
    Field start_up:Image
    'declaring images for playing state backgrounds
    Field backf:Image
    Field backc:Image
    Field backh:Image
    Field backm:Image
    'declaring playing state sprites
    Field health1:heart1
    Field health2:heart2
    Field health3:heart3
```

Here the classes are being declared

```
Method OnRender()
All the graphics drawing goes here:
    Select gamestate
    Case "START_UP"
            DrawImage start_up, 0,0
        Case "MENU"
            DrawImage menu, 0,0
        Case "PLAYING"
            DrawImage backf, 0,0
            DrawImage health1.sprite, health1.x, health1.y
            DrawImage health2.sprite, health2.x, health2.y
            DrawImage health3.sprite, health3.x, health3.y
```

This will draw the sprites to the screen.

Candidate Name: [redacted] Candidate Number: [redacted]

What's being tested?	input	justification	Expected outcome	Actual outcome
Do the heart sprites load in the playing state	N/A	These will act as an indicator for the user's health	The forest background with three hearts	The program was stopped, and an error message came up saying "memory access violation"

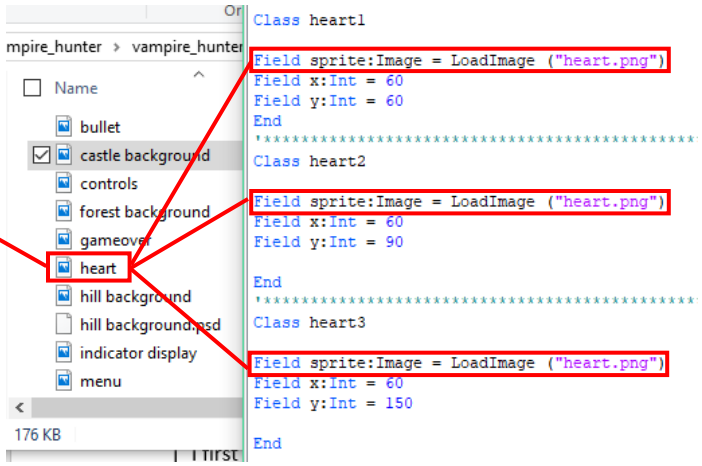
```
        'objects being initialised
        'health1 = New heart1
        'health2 = New heart2
        'health3 = New heart3

    End
'*****************************************************************************************
    Method OnUpdate ()
    'All the game logic goes here:
    'This is the code to allow the user to change between gamestates
        Select gamestate
            Case "START_UP"
                'this is code to pause the display of the splash screen
                For Local x:Int = 0 To 90000
                    x = x + 1
                    For Local y:Int = 0 To 20000
                        y = y + 1
                    Next
                Next
                gamestate = "MENU"
            Case "MENU"
                If KeyHit (KEY_C) Then gamestate="CONTROLS"
                If KeyHit (KEY_1) Then gamestate="PL
            Case "PLAYING"
                If KeyHit (KEY_P) Then gamestate="PA
                If KeyHit (KEY_M) Then gamestate="ME
            Case "CONTROLS"
                If KeyHit (KEY_ESCAPE) Then gamestat
            Case "PAUSED"
                If KeyHit (KEY_P) Then gamestate="PL
        END
    End
'*****************************************************************************************
    Method OnRender()
    ' All the graphics drawing goes here:
        Select gamestate
        Case "START_UP"
            DrawImage start_up, 0,0
        Case "MENU"
            DrawImage menu, 0,0
        Case "PLAYING"
            DrawImage back, 0,0
            DrawImage health1.sprite, health1.x, health1.y
```

Monkey Runtime Error

Memory access violation

OK

I first checked that the image names were all spelt correctly, which they were.

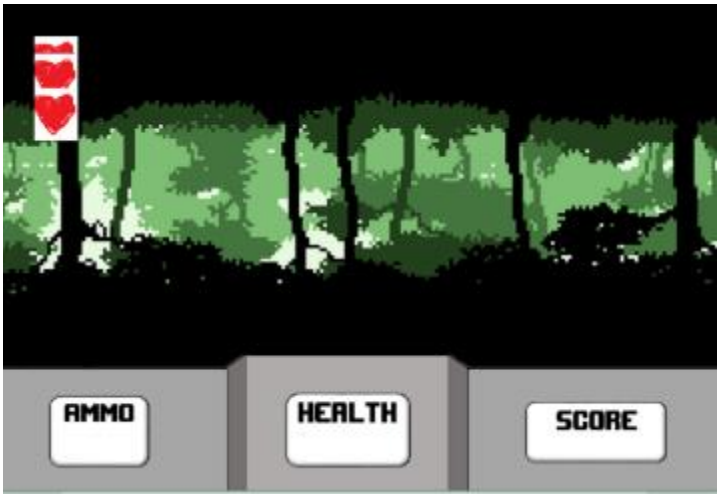


So, I knew I must have missed something out in the code. After tracing my steps back and searching through the code I realised that I had missed out a crucial step, I needed to initialize the objects.

I initialised the objects in the 'on create' method

```
Method OnCreate ()
'All the initialisation for the game goes here:
    'frame rate at which the game runs
    SetUpdateRate 60
    'game state images
    start_up = LoadImage ("start_up.png")
    paused = LoadImage ("paused.png")
    controls = LoadImage ("controls.png")
    menu = LoadImage ("menu.png")
    'diffrent backgrounds for playing state
    backf = LoadImage ("forest background.png")
    backc = LoadImage ("castle background.png")
    backh = LoadImage ("hill background.png")
    backm = LoadImage ("mountain background.png")

    'objects being initialised
    health1 = New heart1
    health2 = New heart2
    health3 = New heart3

End
```

What's being tested?	input	justification	Expected outcome	Actual outcome
Do the heart sprites load in the playing state	N/A	These will act as an indicator for the user's health	The forest background with three hearts	The forest background with three hearts



Now that the sprites load correctly I will reposition the hearts to the right location and repeat the process for the bullets.

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 27/01/18

The three bullet classes being created. I estimated the position of the bullets based on the location of the hearts.

I also changed the data type for the x coordinate to a float so the sprite could be lined up exactly.

```
Class bullet1
    Field sprite:Image = LoadImage ("bullet.png")
    Field x:Float = 206.5
    Field y:Int = 740
End
'*************************************************
Class bullet2
    Field sprite:Image = LoadImage ("bullet.png")
    Field x:Float = 156.5
    Field y:Int = 740
End
'*************************************************
Class bullet3
    Field sprite:Image = LoadImage ("bullet.png")
    Field x:Float = 106.5
    Field y:Int = 740
End
```

```
'declaring playing state sprites
Field health1:heart1
Field health2:heart2
Field health3:heart3
Field ammo1:bullet1
Field ammo2:bullet2
Field ammo3:bullet3
```

Declaring the sprites.

The bullet objects being initialized.

```
'objects being initialised
health1 = New heart1
health2 = New heart2
health3 = New heart3
ammo1 = New bullet1
ammo2 = New bullet2
ammo3 = New bullet3
```

What's being tested?	input	justification	Expected outcome	Actual outcome
Do the bullet sprites load in the playing state	N/A	These will act as an indicator for how many times the user has clicked	The forest background with three hearts and bullets	The forest background with three hearts and bullets

INTERVIEW: 27/01/18

These are the requirements that are being looked at

No.	Requirements	Justification	☑	Reference
1.	Start-up screen.	This is done to show the user what game they are about to play	☐	A feature from the Nintendo: Duck Hunt game
3.	Score box in the bottom right of the screen.	This places the indicator in a clear place for the user to see	☐	A feature from the Nintendo: Duck Hunt game
4.	Score board is shown at the end of each game.	This gives the user a goal to work towards making the game more interesting	☐	Interview: 24/09/17
7.	Shot box, with three bits of ammo, in the bottom left of the screen.	This places the indicator in a clear place for the user to see	☐	A feature from the Nintendo: Duck Hunt game
10.	Life box, has three lives inside, in the middle of the bottom of the screen.	This places the indicator in a clear place for the user to see	☐	Interview: 24/09/17

Me:
I have successfully completed all but one of the requirements shown above. I have managed to get the splash screen to pause for around two seconds and then switch to the menu allowing the user to interact with the game. I have also developed the playing state screen by adding a background and including the screen elements e.g. bullets and hearts. However, I have not done this for the score yet as this will be constantly changing so it will be trickier to implement.

Ben Newton:
You have made good progress. This prototype is looking more like a game. However, I do think that having the score appear on the playing screen is necessary, so I would like you to get around to that. Now that you have gotten the game to look like a game I now want you to get the game elements to work.

Next step:

Following the interview, I want the hearts to disappear when a life is lost and the bullets to disappear when the ammo is used up. I also want to implement to game over game state.

SCREEN ELEMENTS

DATA: 30/01/18

I realized that I could just create one heart and one bullet class and then just make copies of the object making the six classes I created into just two. I could do this because the only difference in the classes are the coordinates.

```
'*****************************************************************
Class heart
'sprite image
    Field sprite:Image = LoadImage ("heart.png")
'variable for the location of the sprite
    Field x:Int
    Field y:Int
' this is giving parmeters for the location of the sprites
    Method New(x_spawn:Int,y_spawn:Int)
        x = x_spawn
        y = y_spawn
    end

End
'*****************************************************************
Class bullet
'sprite image
    Field sprite:Image = LoadImage ("bullet.png")
'variable for the location of the sprite
    Field x:Float
    Field y:Float
' this is giving parmeters for the location of the sprites
    Method New(x_spawn:float,y_spawn:float)
        x = x_spawn
        y = y_spawn
    End
End
'*****************************************************************
```

This will allow the different objects to have different positions when they are

```
'declaring playing state sprites
Field health_collection:List<heart>
Field ammo_collection:List<bullet>
```

This is creating a list that contains a specific type of class

Here I am initializing objects from the heart and bullet classes adding them to lists called health and ammo respectively

```
'objects being initialised
health_collection = New List<heart>
health_collection.AddLast(New heart(640, 745))
health_collection.AddLast(New heart(580, 745))
health_collection.AddLast(New heart(520, 745))

ammo_collection = New List<bullet>
ammo_collection.AddLast(New bullet(206.5, 740))
ammo_collection.AddLast(New bullet(156.5, 740))
ammo_collection.AddLast(New bullet(106.5, 740))
```

```
'this is drawing all the hearts to the screen
For Local health:=Eachin health_collection
    DrawImage health.sprite, health.x, health.y
Next
'this is drawing all the bullets to the screen
For Local ammo:=Eachin ammo_collection
    DrawImage ammo.sprite, ammo.x, ammo.y
Next
```

Two separate for loop are running through both lists and drawing all the objects to the screen.

One of the end goals of this section is to get the heart and bullet sprites to be removed off screen as an indicator for the user when certain things happen whilst playing e.g. the player has lost health or they have used up their ammo.

```
'*************************************************
'All game code goes here:
Class Game_app Extends App
    'declaring the health and bullet variables
    Field life_v:Int = 3
    Field bullet_v:Int = 3
```

I started off by declaring the variables “life_v” and “bullet_v”. The game logic will change the values of these variables due to certain conditions, this will affect what happens to the heart and bullet sprites on screen.

```
If KeyHit (KEY_L) Then life_v = life_v -1
```

To get the player to lose health in the game a bat must escape, but I have not coded this yet. So, instead I have written a line of code that will reduce the "life_v" variable by one every time I press the key "L".

```
If life_v = 2 Then
For Local health:=Eachin health_collection
    health_collection.Remove health
Next
```

Using the code, I deconstructed previously I knew how to remove a list of sprites using a for loop, so I implemented this into my code to test if when I press the "L" key the "life_v" would decrease resulting in the sprites disappearing.

What's being tested?	input	justification	Expected outcome	Actual outcome
When the "L" key is pressed all the heat sprites will be removed of screen	"L" key	This is a test harness for removing hearts when health is lost	All three hearts will be removed	All hearts were removed

My next step was to figure out how to remove the hearts separately. I knew that the objects were inside a list, but I didn't know how to remove only one object out of the list at a time. I look at multiple forums and tried to find certain commands, but I had no success. Due to the time limit I must create the coded solution I decide to create a separate list for each object. This may not be the most efficient code but for now it will do.

```
'declaring playing state sprites
Field health1_collection:List<heart>
Field health2_collection:List<heart>
Field health3_collection:List<heart>
```

I created three different lists with the same class.

Here I initialized each of the objects from the separate lists.

```
'objects being initialised
'three diffrent lists are being created
health1_collection = New List<heart>
health1_collection.AddLast(New heart(640, 745))
health2_collection = New List<heart>
health2_collection.AddLast(New heart(580, 745))
health3_collection = New List<heart>
health3_collection.AddLast(New heart(520, 745))
```

Candidate Name: [redacted] Candidate Number: [redacted]

```
If KeyHit (KEY_L) Then life_v = life_v -1

' this is removing the contents of "health1" given "life_v"=2
If life_v = 2 Then
For Local health1:=Eachin health1_collection
    health1_collection.Remove health1
Next
' this is removing the contents of "health2" given "life_v"=1
Endif
If life_v = 1 Then
For Local health2:=Eachin health2_collection
    health2_collection.Remove health2
Next
' this is removing the contents of "health2" given "life_v"=0
Endif
If life_v = 0 Then
For Local health3:=Eachin health3_collection
    health3_collection.Remove health3
Next
Endif
```

This is the game logic that will remove each sprite at a time depending on the value of the "life_v" variable. For example, when "life_v" equals 2 the first heart is removed.

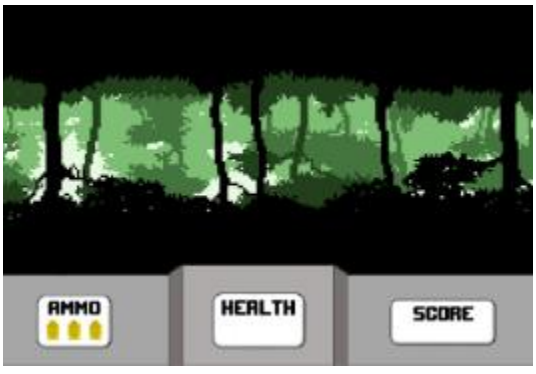
This is drawing each of the lists to the screen for the heart sprites.

```
'this is drawing all the hearts to the screen
'this is drawing all the objects in the list "health1"
For Local health1:=Eachin health1_collection
    DrawImage health1.sprite, health1.x, health1.y
Next
'this is drawing all the objects in the list "health1"
For Local health2:=Eachin health2_collection
    DrawImage health2.sprite, health2.x, health2.y
Next
'this is drawing all the objects in the list "health1"
For Local health3:=Eachin health3_collection
    DrawImage health3.sprite, health3.x, health3.y
Next
'this is drawing all the bullets to the screen
For Local ammo:=Eachin ammo_collection
    DrawImage ammo.sprite, ammo.x, ammo.y
Next
```

What's being tested?	input	justification	Expected outcome	Actual outcome
When the "L" key is pressed once the right heart is removed	"L" key	This is a test harness for removing hearts when health is lost	The right heart in the health box is removed	The right heart in the health box was removed

What's being tested?	input	justification	Expected outcome	Actual outcome
When the "L" key is pressed twice the right and middle hearts are removed	"L" key	This is a test harness for removing hearts when health is lost	The right and middle hearts in the health box are removed	The right and middle hearts in the health box were removed

What's being tested?	input	justification	Expected outcome	Actual outcome
When the "L" key is pressed three times all hearts are removed	"L" key	This is a test harness for removing hearts when health is lost	There are no hearts in the health box	There are no hearts in the health box



Now I will repeat this process for the bullets in the ammo box.

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 01/02/18

```
'declaring the three lists for ammo
Field ammo1_collection:List<bullet>
Field ammo2_collection:List<bullet>
Field ammo3_collection:List<bullet>
```

I created three different lists with the same class.

Here I initialized each of the objects from the separate lists.

```
'a bullet object is being added to each of the lists
ammo1_collection = New List<bullet>
ammo1_collection.AddLast(New bullet(206.5, 740))
ammo2_collection = New List<bullet>
ammo2_collection.AddLast(New bullet(156.5, 740))
ammo3_collection = New List<bullet>
ammo3_collection.AddLast(New bullet(106.5, 740))
```

```
If KeyHit (KEY_A) Then bullet_v = bullet_v -1
' this is removing the contents of "ammo1" given "bullet_v"=2
If bullet_v = 2 Then
For Local ammo1:=Eachin ammo1_collection
    ammo1_collection.Remove ammo1
Next
endif
' this is removing the contents of "ammo2" given "bullet_v"=1
If bullet_v = 1 Then
For Local ammo2:=Eachin ammo2_collection
    ammo2_collection.Remove ammo2
Next
endif
' this is removing the contents of "ammo3" given "bullet_v"=0
If bullet_v = 0 Then
For Local ammo3:=Eachin ammo3_collection
    ammo3_collection.Remove ammo3
Next
Endif
```

This is the game logic that will remove each sprite at a time depending on the value of the “bullet_v” variable.

This is drawing each of the lists to the screen for the heart sprites.

```
'this is drawing all the objects in the list "ammo1"
For Local ammo1:=Eachin ammo1_collection
    DrawImage ammo1.sprite, ammo1.x, ammo1.y
Next
'this is drawing all the objects in the list "ammo2"
For Local ammo2:=Eachin ammo2_collection
    DrawImage ammo2.sprite, ammo2.x, ammo2.y
Next
'this is drawing all the objects in the list "ammo3
For Local ammo3:=Eachin ammo3_collection
    DrawImage ammo3.sprite, ammo3.x, ammo3.y
Next
```

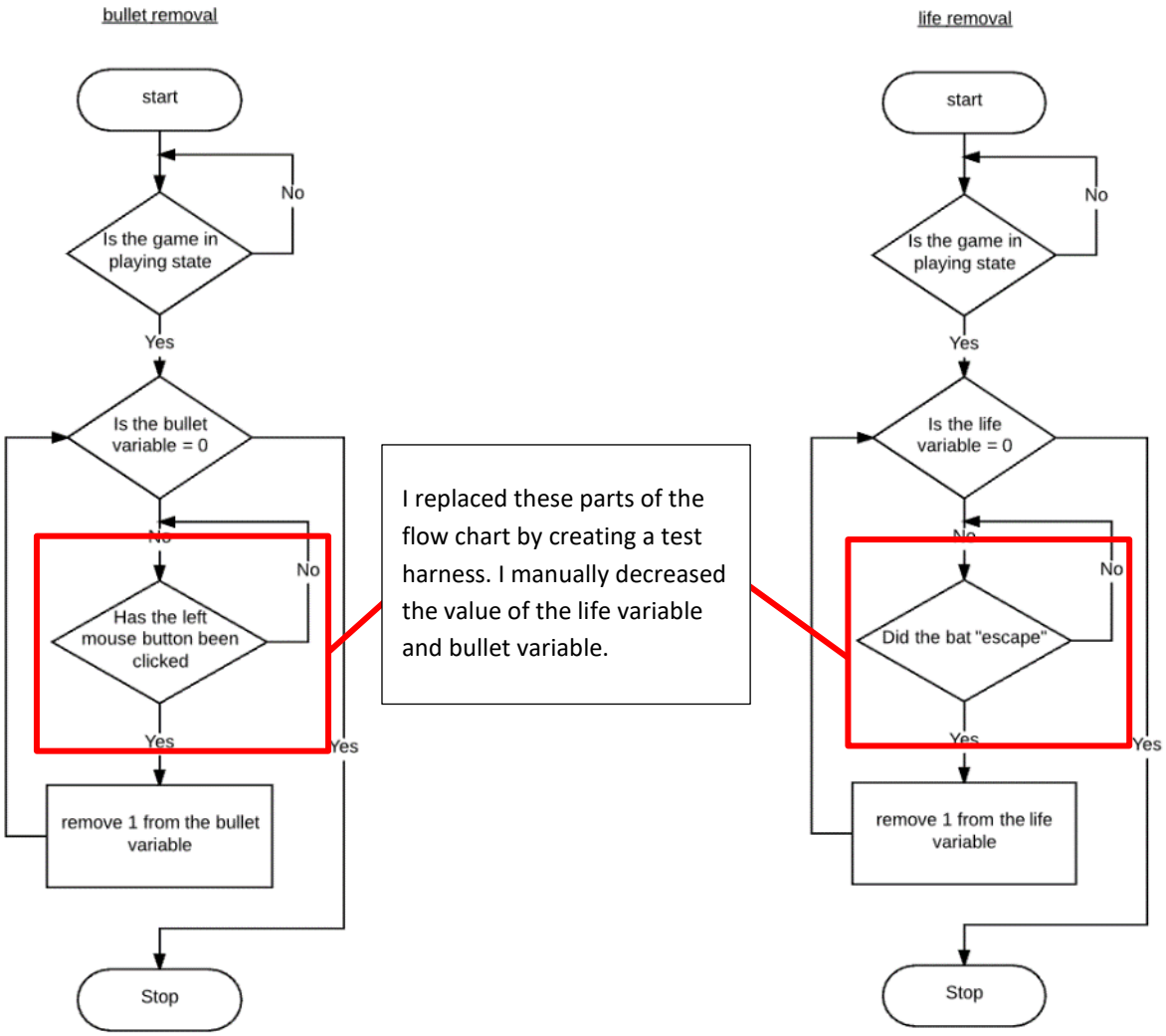
What's being tested?	input	justification	Expected outcome	Actual outcome
When the "A" key is pressed once the right bullet is removed	"A" key	This is a test harness for removing ammo when the bullets are used	The right bullet in the ammo box is removed	The right bullet in the ammo box was removed

What's being tested?	input	justification	Expected outcome	Actual outcome
When the "A" key is pressed twice the right and middle bullets are removed	"A" key	This is a test harness for removing ammo when the bullets are used	The right and middle bullets in the ammo box are removed	The right and middle bullets in the ammo box were removed

What's being tested?	input	justification	Expected outcome	Actual outcome
When the "A" key is pressed three times all bullets are removed	"A" key	This is a test harness for removing ammo when the bullets are used	There are no bullets in the ammo box	There are no bullets in the ammo box


AMMO
HEALTH
SCORE

FLOW CHART


bullet removal
start
Is the game in playing state
No
Yes
Is the bullet variable = 0
No
Has the left mouse button been clicked
No
Yes
Yes
remove 1 from the bullet variable
Stop
I replaced these parts of the flow chart by creating a test harness. I manually decreased the value of the life variable and bullet variable.
life removal
start
Is the game in playing state
No
Yes
Is the life variable = 0
No
Did the bat "escape"
No
Yes
Yes
remove 1 from the life variable
Stop

GAME OVER

DATE: 03/02/18

Now that I have successfully implemented the features of the hearts and the bullets I want to add the game over state. The game over state will only activate when the player is dead, so I will link it to the loss of health.

```
'declaring images for game states
Field menu:Image
Field controls:Image
Field paused:Image
Field gameover:Image
Field start_up:Image
```

I first declared the image that will be used for the game over game state.

I then loaded the game over image into the game.

```
'game state images
start_up = LoadImage ("start_up.png")
paused = LoadImage ("paused.png")
controls = LoadImage ("controls.png")
menu = LoadImage ("menu.png")
gameover = LoadImage("gameover.png")
```

```
Case "GAMEOVER"
    DrawImage gameover, 0,0
```

This will draw the game over image to the screen.

```
' this is changing playing state to the gameover state when no health is left
If life_v = 0 Then
gamestate = "GAMEOVER"
endif
```

When the player loses their last life the game state switches from playing state to the game over state.

What's being tested?	input	justification	Expected outcome	Actual outcome
When the "L" key is pressed three times all hearts are removed	"L" key	This is to test when all health is lost that the game state changes to game over	The game over image is displayed	The game over image was displayed

As I have not yet created the high score game state after the game over image is displayed for a certain amount of time it should just switch back to the game state.

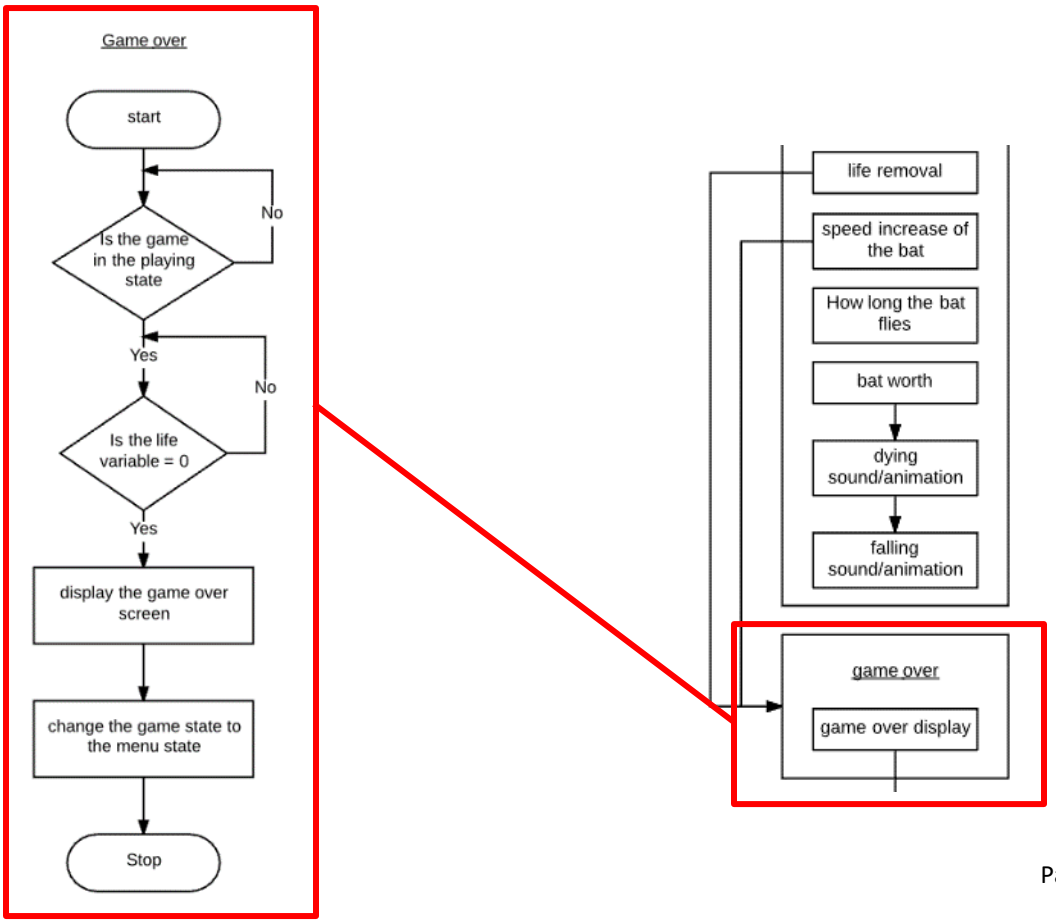
```
Case "GAMEOVER"
    For Local x:Int = 0 To 90000
        x = x + 1
        For Local y:Int = 0 To 20000
            y = y + 1
        Next
    Next
    gamestate = "MENU"
```

I used a nested for loop to pause the image for the game over state.

This is changing the game state from the game over state to the playing state.

What's being tested?	input	justification	Expected outcome	Actual outcome
If the game state switches from the game over state to the menu state	N/A	This will allow the user to play another round of the game	The game over image is displayed for a short time then switches to the game state	The game over image was displayed for a short time then switched to the game state

FLOW CHART



Candidate Name: [redacted] Candidate Number: [redacted]

When going through my prototype I realised that when I returned to the playing state after having "played" a game that the screen elements were not reset. This is most likely easily fixed by including an initialise state.

INTERVIEW: 04/02/18

These are the requirements that are being looked at

No.	Requirements	Justification	☑	Reference
5.	Three shots per mini round	This was done to make the game more challenging, so the game wouldn't become too easy. Easy games can become easily boring	☐	A feature from the Nintendo: Duck Hunt game
6.	Ammo disappears as the player shoots at the bats until there is no ammo left so the player can no longer shoot.	This acts as an indicator for the user as they play the game, so they have an idea of how many shots they have already used	☐	A feature from the Nintendo: Duck Hunt game
30.	When the player loses all their health a game over screen appears	It is a clear indicator for the use	☐	A feature from the Nintendo: Duck Hunt game
8.	Three lives in total	This was done to make the game challenging as it gives the user a way to lose the game.	☐	Interview: 24/09/17
9.	If a bat "escapes" the user loses a life	This acts as an indicator for the user as they play the game, so they have an idea of how many lives they have left	☐	Interview: 24/09/17

Me:
I have coded most of the aspects of the given requirements. This includes the removal of health and ammo given certain conditions of the game. I used a test harness for this as I have not yet coded the bat AI or LMC in the games perimeter. When the players health is gone the game state changes to the game over game state. Furthermore, the game over image is displayed for about two seconds and then the game state changes to the menu state. However, I did notice a problem when re-entering the playing state, the screen elements did not reset. This means I'll have to code another game state that will initialise all the necessary things for the game.

Ben Newton:
I am happy with what you have done for this prototype. But, it is important that you get the games initialization right because the game would not be very good if you could only play it once without having to restart the program. So, I would like you to have this fixed for the next prototype.

After the interview new requirements were added.

New requirements

No.	Requirements	Justification	☑	Reference
33.	Re-playability of game	This will allow the user to replay a game as many times as they want without having to re start the game	☐	Interview: 04/02/18

Next step:

Following the interview, I want to make sure that the game can be replayed without having to restart the program so I will start of by creating an initialization state. Furthermore, I want to add sound effects and music to the game.

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 07/02/18

GAME INITIALIZATION

All I had to do was add another case in the “on update” method. I reset the “life_v” and “bullet_v” to the value 3 so the playing state would go straight to the game over state.

```
Case "INITIALISE"
    life_v = 3
    bullet_v = 3
    'objects being initialised
    'a heart object is being added to each of the lists
    health1_collection = New List<heart>
    health1_collection.AddLast(New heart(640, 745))
    health2_collection = New List<heart>
    health2_collection.AddLast(New heart(580, 745))
    health3_collection = New List<heart>
    health3_collection.AddLast(New heart(520, 745))

    'a bullet object is being added to each of the lists
    ammo1_collection = New List<bullet>
    ammo1_collection.AddLast(New bullet(206.5, 740))
    ammo2_collection = New List<bullet>
    ammo2_collection.AddLast(New bullet(156.5, 740))
    ammo3_collection = New List<bullet>
    ammo3_collection.AddLast(New bullet(106.5, 740))

    gamestate = "PLAYING"
```

After everything has been initialized the game state switches to the playing state.

I also moved the adding of the objects to the lists to the “initialise” game state so that the removed objects from the playing state are added back to the list at the start of every new game.

```
Case "MENU"
    if KeyHit (KEY_C) Then gamestate="CONTROLS"
    If KeyHit (KEY_1) Then gamestate="INITIALISE"
```

So, from the menu state if the user wanted to play a game the game state would first change to the initialised state so everything can be rest for the game.

I now want to add music to the menu and sound effects to the screen elements, splash screen and game over state. I will begin by adding sound effects to the bullet screen elements.

GAME SOUNDS

```
'declaring sound for the game
Field gunshot:Sound
```

Here I declared the variable gunshot and gave it a data type 'sound'.

The gun shot sound is being loaded into the game.

```
'loading sounds
gunshot = LoadSound ("sfx_weapon_singleshot21.wav")
```

I want the sound to play when a bullet sprite is removed so I placed it within the 'if' statements that include the code that removes the sprites.

```
If KeyHit (KEY_A) Then bullet_v = bullet_v -1

' this is removing the contents of "ammo1" given "bullet_v"=2
If bullet_v = 2 Then
    PlaySound(gunshot,1,0)
    For Local ammo1:=Eachin ammo1_collection
        ammo1_collection.Remove ammo1
    Next
Endif

' this is removing the contents of "ammo2" given "bullet_v"=1
If bullet_v = 1 Then
    PlaySound(gunshot,1,0)
    For Local ammo2:=Eachin ammo2_collection
        ammo2_collection.Remove ammo2
    Next
Endif

' this is removing the contents of "ammo3" given "bullet_v"=0
If bullet_v = 0 Then
    PlaySound(gunshot,1,0)
    For Local ammo3:=Eachin ammo3_collection
        ammo3_collection.Remove ammo3
    Next
Endif
```

What's being tested?	input	justification	Expected outcome	Actual outcome
If the gun shot sound effect will play when a bullet sprite is removed.	N/A	This will act as an indicator for the user.	The sound effect should play once when one sprite is removed.	The sound effect is played on a loop

DATE: 11/02/18

I found this quite confusing to begin with, but after a while I realised that because the sound would play when the bullet variable is a certain value and combined with this the 'on update' class is looping as fast as the CPU is allowing. This means that the line of code making the sound play is being run over and over again as long as the bullet variable is a certain value.

To solve this problem, I increased the 'bullet_v' variable from 3 to 6. The idea behind this was that the sound effect playing, and the removing of the bullet sprites are triggered by different values. So, when a bullet is "shot" the 'bullet_v' value decrease by 1 causing a sound effect to play then the value is decreased by 1 again causing the sprite to be removed. This results in the sound playing once and the sprite being removed. Each sprite is in its own list and the remove function for the sprite is removing everything from that list. The loop is causing this to happen repeatedly but makes no difference as removing nothing does nothing.

Furthermore, I didn't let the sound effect play when the key hit was triggered as this would mean the sound effect would play even when there are no sprites left. The game would then be indicating false information to the user.

I assigned the 'bullet_v' and 'life_v' variable values of 6.

```
Case "INITIALISE"
    life_v = 6
    bullet_v = 6
```

This is the test harness for the removal of bullets.

By pressing the ‘a’ key this simulates a bullet being “shot”. The value of ‘bullet_v’ decrease by 1 causing a gunshot to play. The value is then decreased by one straight after causing the sprite to be removed.

This process is repeated 3 times for the three different bullet sprites.

```
If KeyHit (KEY_A) Then bullet_v = bullet_v -1
```

```
If bullet_v = 5 Then
    PlaySound(gunshot,1,0)
    bullet_v = 4
Endif

' this is removing the contents of "ammo1" given "bullet_v"=2
If bullet_v = 4 Then
For Local ammo1:=Eachin ammo1_collection
    ammo1_collection.Remove ammo1
Next
Endif
```

```
If bullet_v = 3 Then
    PlaySound(gunshot,1,0)
    bullet_v = 2
Endif

' this is removing the contents of "ammo2" given "bullet_v"=1
If bullet_v = 2 Then
For Local ammo2:=Eachin ammo2_collection
    ammo2_collection.Remove ammo2
Next
Endif
```

```
If bullet_v = 1 Then
    PlaySound(gunshot,1,0)
    bullet_v = 0
Endif

' this is removing the contents of "ammo3" given "bullet_v"=0
If bullet_v = 0 Then
For Local ammo3:=Eachin ammo3_collection
    ammo3_collection.Remove ammo3
Next
Endif
```

What’s being tested?	input	justification	Expected outcome	Actual outcome
If the gun shot sound effect will play when a bullet sprite is removed.	N/A	This will act as an indicator for the user.	The sound effect should play once when one sprite is removed. This should occur for all three sprites.	The sound effect played once when one sprite is removed. This occurred for all three sprites.

Candidate Name: [redacted] Candidate Number: [redacted]

I used the method I designed to add sound to sprites being removed on for the health screen elements as well.

```
healthloss =  LoadSound ("sfx_sounds_button8.wav")
```

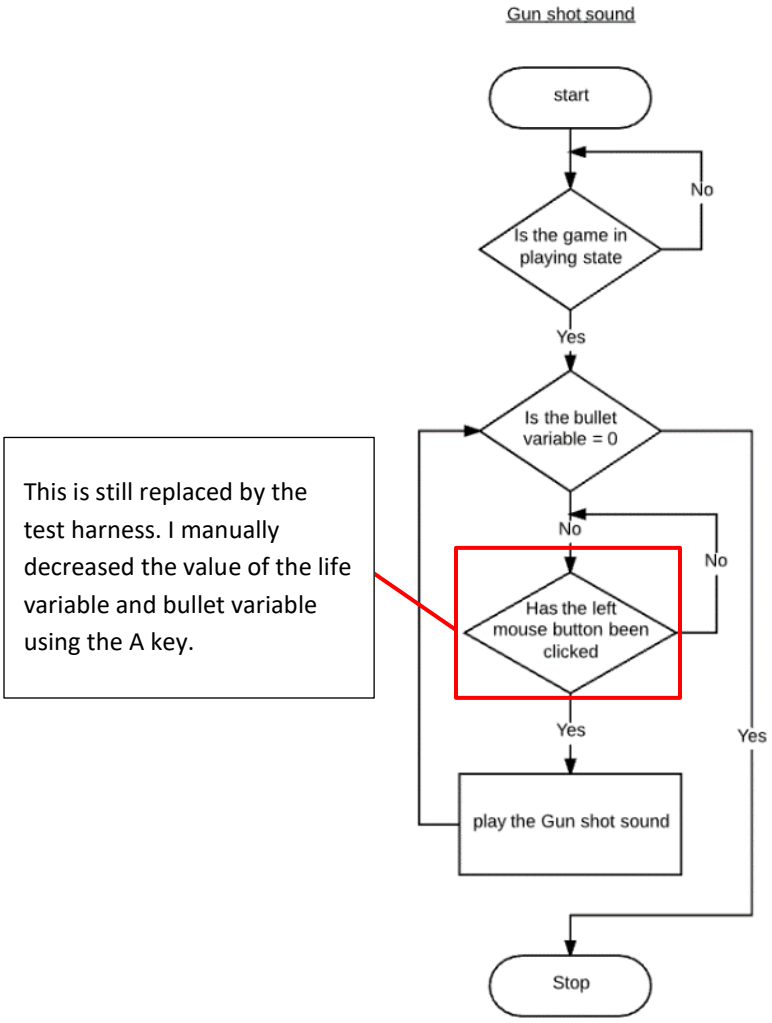
Here the sound effect for health loss was loaded into the game.

This is the code for removing sprites with a sound effect. The code for this construct is explained in the previous section for the removal of bullet sprites with sound effects.

```
If KeyHit (KEY_L) Then life_v = life_v -1

If life_v = 5 Then
PlaySound(healthloss,1,0)
life_v = 4
Endif
' this is removing the contents of "health1" given "life_v"=2
If life_v = 4 Then
For Local health1:=Eachin health1_collection
    health1_collection.Remove health1
Next
Endif

If life_v = 3 Then
PlaySound(healthloss,1,0)
life_v = 2
Endif
' this is removing the contents of "health2" given "life_v"=1
If life_v = 2 Then
For Local health2:=Eachin health2_collection
    health2_collection.Remove health2
Next
Endif

If life_v = 1 Then
PlaySound(healthloss,1,0)
life_v = 0
Endif

' this is changing playing state to the gameover state when no health is left
If life_v = 0 Then
gamestate = "GAMEOVER"
Endif
```

What's being tested?	input	justification	Expected outcome	Actual outcome
If the health removal sound effect will play when a heart sprite is removed.	N/A	This will act as an indicator for the user.	The sound effect should play once when one sprite is removed. This should occur for all three sprites.	The sound effect played once when one sprite is removed. This occurred for all three sprites.

FLOW CHART



This is still replaced by the test harness. I manually decreased the value of the life variable and bullet variable using the A key.

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 14/02/18

Now I want to add some of the other sound effects to the game. I want to add: a transitioning sound from the playing state to the game over state; pause sound effects; menu music; intro sound and menu sounds for selecting.

Transitioning sound from the playing state to the game over state

```
' this is changing playing state to the gameover state when no health is left
If life_v = 1 Then
'this is changing game states
gamestate = "GAMEOVER"
Endif
```

```
Case "GAMEOVER"
    'this is playing the game over sound effect
    PlaySound(gameoverS,1,0)
    'this pauses the game on the game over state for a while
    For Local x:Int = 0 To 90000
        x = x + 1
        For Local y:Int = 0 To 20000
            y = y + 1
        Next
    Next
    'this is changing game states
    gamestate = "MENU"
```

To get the transitioning sound effect to play properly I just let the sound effect play once at the start of the game over game state and I didn't put in a sound effect for the last heart sprite being removed as I want to indicate the game is over, not that they have lost another life.

What's being tested?	input	justification	Expected outcome	Actual outcome
If the transitioning sound plays between the playing and game over state.	N/A	This will act as an indicator for the user for when the game has ended.	The sound effect should play when the player has lost all health	The sound effect played when the player lost all health

Candidate Name: [redacted] Candidate Number: [redacted]

Pause sound effects

Here I just got the sound effect of entering the game state to play when the game state changes from playing to paused.

```
Case "PLAYING"
    If KeyHit (KEY_P) Then
        'this is changing game states
        gamestate="PAUSED"
        'this is playing the sound effect for entering the paused state
        PlaySound(pausein,1,0)
    Endif
```

I also got this to happen when the game state changes from paused to playing but with an 'exiting' sound effect.

```
Case "PAUSED"
    If KeyHit (KEY_P) Then
    'this is changing game states
    gamestate="PLAYING"
    'this is playing the sound effect for exiting the paused state
    PlaySound(pauseout,1,0)
    endif
```

What's being tested?	input	justification	Expected outcome	Actual outcome
A sound effect plays when entering the paused state	N/A	This will act as an indicator for the user to tell the user if the game has paused.	The sound effect should play when the paused game state image is displayed.	The sound effect played when the paused game state image is displayed

What's being tested?	input	justification	Expected outcome	Actual outcome
A sound effect plays when returning from the paused state to the playing state	N/A	This will act as an indicator for the user when the game has been resumed.	The sound effect should play when the game is resumed.	The sound effect played when the game resumed.

Candidate Name: [redacted] Candidate Number: [redacted]

Menu music

This sound effect is so that the menu has music playing when the game is in the menu state.

The last parameter of the 'PlaySound()' allows the sound effect to play on loop by setting the parameter to 1.

To get the sound effect to play at the right time I played the sound effect every time the code would return to the menu game state.

```
Case "START_UP"
    'this is playing the start up sound
    PlaySound(intro,1,0)
    'this is code to pause the display of the splash screen
    For Local x:Int = 0 To 90000
        x = x + 1
        For Local y:Int = 0 To 20000
            y = y + 1
        Next
    Next
    'this is changing game states
    gamestate = "MENU"
    'this will play the menu music
    PlaySound(menuM,1,1)


    If KeyHit (KEY_M) Then
        'this is changing game states
        gamestate="MENU"
        'this is playing the
        PlaySound(menuM,1,1)
     Endif
Case "GAMEOVER"
    'this is playing the game over sound effect
    PlaySound(gameoverS,1,0)
    'this pauses the game on the game over state for a while
    For Local x:Int = 0 To 90000
        x = x + 1
        For Local y:Int = 0 To 20000
            y = y + 1
        Next
    Next
    'this is changing game states
    gamestate = "MENU"
    'this is playing the menu music
    PlaySound(menuM,1,1)

    Case "CONTROLS"
        If KeyHit (KEY_ESCAPE) Then
        'this is changing game states
        gamestate="MENU"
        PlaySound(menuM,1,1)
        Endif
```

What's being tested?	input	justification	Expected outcome	Actual outcome
The menu music plays when transitioning from the 'start up' to the 'menu' game state	N/A	This separates the menu from the playing state a little more and makes it more interesting	The music will play when in the menu state	The music played when in the menu state

What's being tested?	input	justification	Expected outcome	Actual outcome
The menu music plays when transitioning from the 'playing' to the 'menu' game state as a result of the 'm' key being pressed	N/A	This separates the menu from the playing state a little more and makes it more interesting	The music will play when in the menu state	The music played when in the menu state

What's being tested?	input	justification	Expected outcome	Actual outcome
The menu music plays when transitioning from the 'game over' to the 'menu' game state	N/A	This separates the menu from the playing state a little more and makes it more interesting	The music will play when in the menu state	The music played when in the menu state

What's being tested?	input	justification	Expected outcome	Actual outcome
The menu music plays when transitioning from the 'control' to the 'menu' game state	N/A	This separates the menu from the playing state a little more and makes it more interesting	The music will play when in the menu state	The music played when in the menu state

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 17/02/18

Intro sound

I wanted the introduction of the game to have a sound effect. So, I placed the sound before the pause created by the 'for' loops so that the sound would play during the pausing of the display.

```
Case "START UP"
    'this is playing the start up sound
    PlaySound(intro,1,0)
    'this is code to pause the display of the splash screen
    For Local x:Int = 0 To 90000
        x = x + 1
        For Local y:Int = 0 To 20000
            y = y + 1
        Next
    Next
    'this is changing game states
    gamestate = "MENU"
```

What's being tested?	input	justification	Expected outcome	Actual outcome
The intro sound effect playing for the games splash screen	N/A	It goes well with the splash screen making the game more interesting to experience	The intro sound effect will play when the splash screen image appears	The intro sound effect played when the splash screen image appeared

Menu sounds for selecting

```
Case "MENU"
    If KeyHit (KEY_C) Then
        'this is changing game states
        gamestate="CONTROLS"
        'this is playing the menu select sound effect
        PlaySound(menuS,1,0)
    EndIf

    If KeyHit (KEY_1) Then
        'this is changing game states
        gamestate="INITIALISE"
        'this is playing the menu select sound effect
        PlaySound(menuS,1,0)
    EndIf

    If KeyHit (KEY_2) Then
        'this is playing the negative menu select sound effect
        PlaySound(menuSnegative,1,0)
```

A 'select' sound effect was to be included in the menu state. Any time the menu game state is going to change to another game state a select sound effect would be played.

As I included a 'player two' option in the menu display which can't be interacted with, I decided to add a sound effect that would imply no access. I did this by playing a negative sound effect when the '2' key was pressed.

What's being tested?	input	justification	Expected outcome	Actual outcome
The menu select sound effect is played when entering the 'playing' state.	N/A	This acts as indicator for when the player has selected something in the menu.	The select sound effect plays when the 'playing' game state is entered from the menu.	The select sound effect played when the 'playing' game state was entered from the menu.

What's being tested?	input	justification	Expected outcome	Actual outcome
The menu select sound effect is played when entering the 'control' state.	N/A	This acts as indicator for when the player has selected something in the menu.	The select sound effect plays when the 'control' game state is entered from the menu.	The select sound effect played when the 'control' game state was entered from the menu.

What's being tested?	input	justification	Expected outcome	Actual outcome
The negative menu select sound effect is played when trying to enter the two player mode, then the menu music continues.	N/A	This lets the user know that they cannot access the two player game mode	The negative menu select sound is plays and then the menu music continues playing	The negative menu select sound played but was followed by silence

To try to solve this problem I got the menu sound effect to continue playing after the negative select sound effect played.

```
If KeyHit (KEY_2) Then
    'this is playing the negative menu select sound effect
    PlaySound(menuSnegative,1,0)
    'this is playing the menu music
    PlaySound(menuM,1,1)
EndIf
```

The menu music being restarted.

What's being tested?	input	justification	Expected outcome	Actual outcome
The negative menu select sound effect is played when trying to enter the two player mode, then the menu music continues.	N/A	This lets the user know that they cannot access the two player game mode	The negative menu select sound is plays and then the menu music continues playing	The menu music is just restarted.

I realized the negative select sound effect wasn't playing because it didn't get enough time to play as the menu music was just being played instead.

```
If KeyHit (KEY_2) Then
    'this is playing the negative menu select sound effect
    PlaySound(menuSnegative,1,0)
    'this is giving the sound effect time to play before returing to the playing of menu music
    For Local x:Int = 0 To 10000
        x = x + 1
        For Local y:Int = 0 To 10000
            y = y + 1
        Next
    Next
    'this is playing the menu music
    PlaySound(menuM,1,1)
Endif
```

I added a 'for' loop to "pause" the game long enough for the sound effect to play and then allow the menu music to continue.

What's being tested?	input	justification	Expected outcome	Actual outcome
The negative menu select sound effect is played when trying to enter the two player mode, then the menu music continues.	N/A	This lets the user know that they cannot access the two player game mode	The negative menu select sound is plays and then the menu music continues playing	The negative menu select sound is played and then the menu music continued playing

DATE: 22/02/18

INTERVIEW:22/02/18

No.	Requirements	Justification	☑	Reference
31.	A sound effect for when health is lost	It is a clear indicator for the user	☐	A feature from the Nintendo: Duck Hunt game
32.	Game music for some of the game states	Makes the game more interesting by giving the game a bit more depth with sound	☐	A feature from the Nintendo: Duck Hunt game
6.	Ammo disappears as the player shoots at the bats until there is no ammo left so the player can no longer shoot.	This acts as an indicator for the user as they play the game, so they have an idea of how many shots they have already used	☐	A feature from the Nintendo: Duck Hunt game
33.	Re-playability of game	This will allow the user to replay a game as many times as they want without having to re start the game	☐	Interview: 04/02/18
33.	Sound effect for the shooting. This will occur when left mouse button is clicked	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.	☐	A feature from the Nintendo: Duck Hunt game

Me:
I have managed to code all the requirements successfully. The sound effects have been added to the screen elements. So, when the hearts/bullets are removed the sound effects play in the correct manner. I have implemented menu music but didn't think it was necessary to add music to any other game states as the user will not spend much time in them. However, I did add a sound effect for splash screen and a sound effect for the transition between the playing state and the game over state. Furthermore, I also made the game re-playable by including an initialize state, so whenever the user wants to play another game all the playing state features are reset.

Candidate Name: [redacted] Candidate Number: [redacted]

Ben Newton:

I like the way this prototype has come out; this prototype is starting to become more of a game. I like that it has sound now and that the game can be played as many times as the user wants. However, I would like the test harness to be removed and have some bat features.

Following this interview I have decided that I will the get the bullets to disappear with a mouse left click and the bullets to be reset after a bat has been killed or has escaped. I will also have to create a bat class and draw it to the screen.

DATE: 24/02/18

BULLET REMOVAL WITH A LEFT MOUSE CLICK

```
'declaring mouse coordinates
Field mouseX:Int
Field mouseY:Int
```

I first declared a global variable for the y coordinate of the mouse and the x coordinate of the mouse.

The 'MouseX()' and the 'MouseY()' function is updating the value of the mouse coordinates. These values are then given to the 'mouseX' and 'mouseY' variables to be used in the code.

```
'this is updating the mouse coordinates
mouseX = MouseX()
mouseY = MouseY()

'when the left mouse is clicked and the y mouse -
'- coordinate is less than 610 the bullet variable will be reduced
If MouseHit (MOUSE_LEFT) And mouseY<610 Then bullet_v = bullet_v -1
```

Here I removed the test harness which was the 'if' statement, that was triggered when the key A was pressed, that removes the bullet sprites. Now the bullet removal is triggered by pressing the left mouse button.

What's being tested?	input	justification	Expected outcome	Actual outcome
When there is a LMC once the right bullet is removed	LMC	This makes the game more challenging as the user has a finite number of tries to hit the target	The right bullet in the ammo box is removed	The right bullet in the ammo box was removed

What's being tested?	input	justification	Expected outcome	Actual outcome
When there is a LMC twice the right and middle bullets are removed	LMC	This makes the game more challenging as the user has a finite number of tries to hit the target	The right and middle bullets in the ammo box are removed	The right and middle bullets in the ammo box were removed

What's being tested?	input	justification	Expected outcome	Actual outcome
When there is a LMC three times all bullets are removed	LMC	This makes the game more challenging as the user has a finite number of tries to hit the target	There are no bullets in the ammo box	There are no bullets in the ammo box


AMMO
HEALTH
SCORE

TESTING INPUT DATA

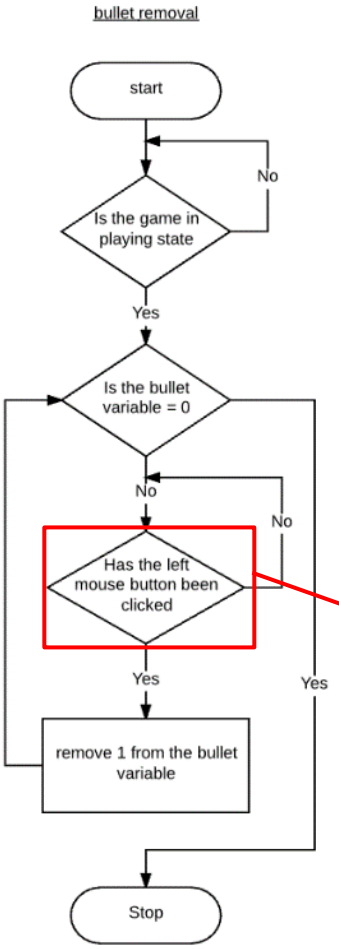
Bullet removal

Test data	Expected	Actual
“LMC”(left mouse click) within the parameter of (1250,850)	Valid	Valid
“RMC”(right mouse click) within the parameter of (1250,850)	Invalid	Invalid
“LMC”(left mouse click) on the target	Valid	Valid
“RMC”(right mouse click) on the target	Invalid	Invalid
“LMC”(left mouse click) outside the parameter of (1250,850)	Invalid	Invalid
“RMC”(right mouse click) outside the parameter of (1250,850)	Invalid	Invalid
“LMC”(left mouse click) on the parameter of (1250,850)	Valid	Valid
“RMC”(right mouse click) on the parameter of (1250,850)	Invalid	Invalid

Gun shot

Test data	Expected	Actual
“LMC” (left mouse click) within the parameter of (1250, 610)	Valid	Valid
“RMC” (right mouse click) within the parameter of (1250, 610)	Invalid	Invalid
“LMC”(left mouse click) on the target	Valid	Valid
“RMC”(right mouse click) on the target	Invalid	Invalid
“LMC” (left mouse click) outside the parameter of (1250, 610)	Invalid	Invalid
“RMC” (right mouse click) outside the parameter of (1250, 610)	Invalid	Invalid
“LMC” (left mouse click) on the parameter of (1250, 610)	Borderline	Borderline
“RMC” (right mouse click) on the parameter of (1250, 610)	Borderline	Borderline

FLOW CHART



I removed the test harness that consisted of pressing the 'A' key to remove the bullets.

Now I want to create the bat class and instantiate an object. After I get this to work I will implement the feature that restores all ammo after the bat is killed or escapes.

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 02/03/18

BAT OBJECT

First, I created a class for the bat object. I gave the class an image of a pixel art bat and parameters of the coordinates for when the class is instantiated.

```
'this is the class for the bullet screen elements:
Class bat
    'sprite image
    Field sprite:Image = LoadImage ("bat.png")

    'variable for the location of the sprite
    Field x:Float
    Field y:Float

    'this is giving parmeters for the location of the sprites
    Method New(x_spawn:float,y_spawn:float)
        x = x_spawn
        y = y_spawn
    End
End
```

```
'declaring the bat list
Field batl_collectoin:List<bat>
```

I created a list in which the bat objects will be placed

Here I instantiated a bat object with the coordinates (500, 300)

```
'declaring the bat list
batl_collectoin = New List<bat>
batl_collectoin.AddLast(New bat(500, 300))
```

What's being tested?	input	justification	Expected outcome	Actual outcome
The bat object is spawned when the game enters the game state.	N/A	This is the target which the user will shoot at to gain points.	The bat is spawned roughly in the centre of the screen.	The bat was spawned roughly in the centre of the screen.

Candidate Name: [redacted] Candidate Number: [redacted]

My next step is to be able to remove the bat using collision. Is started off by going to the “Craig n Dave” resources, the same resource I visited when doing my code research. By doing this I found a function that would allow the code to detect when two objects have collided.

These are the coordinates of both objects.

These are the width and heights of the objects.

```
'collision detection
Function intersects:Bool (x1:Int, y1:Int, w1:Int, h1:Int, x2:Int, y2:Int, w2:Int, h2:Int)
If x1 >= (x2 + w2) Or (x1 + w1) <= x2 Then Return False
If y1 >= (y2 + h2) Or (y1 + h1) <= y2 Then Return False
Return True
End
```

This is checking if the objects are over lapping.

This is only check when the left mouse button is clicked.

These are the coordinates of the cursor and the bat object respectively.

These are the width and height of the cursor and the bat object respectively.

```
If KeyHit (KEY_LMB) then
    For Local bat1:=Eachin bat1_collectoin
        If intersects(mouseX,mouseY,60,60,500,300,94,51) Then
            bat1_collectoin.Remove bat1
        End
    Next
End
```

This is removing all the bat objects from the list bat1.

What’s being tested?	input	justification	Expected outcome	Actual outcome

The bat object is removed when clicked on.	Left mouse click.	This will cleat the screen for the next bat to appear.	The bat will disappear when left mouse clicked on by the user.	The bat disappeared when the bat was left mouse clicked on.

TESTING INPUT DATA

Dying animation

Test data	Expected	Actual
“LMC”(left mouse click) on the target	Valid	Valid
“RMC”(right mouse click) on the target	Invalid	Invalid
“LMC”(left mouse click) within the parameter of (1250, 850)	Invalid	Invalid
“RMC”(right mouse click) within the parameter of (1250, 850)	Invalid	Invalid

Now that the user can kill the bat by pressing on it I want all the ammo to be restored after the bat is killed or if it escaped ready for the next bat that will be spawned in.

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 08/03/18

```
If KeyHit (KEY_LMB) then
    For Local bat1:=Eachin bat1_collectoin
        If intersects(mouseX,mouseY,60,60,500,300,94,51) Then
            bat1_collectoin.Remove bat1
            ammo2_collection.AddLast(New bullet(156.5, 740))
            ammo1_collection.AddLast(New bullet(206.5, 740))
            ammo3_collection.AddLast(New bullet(106.5, 740))
            bullet_v = 6
        End
    Next
End
```

Directly after the bat is removed off the screen the three bullet sprites are drawn back on screen and the "bullet_v" is set back to the value of 6.

What's being tested?	input	justification	Expected outcome	Actual outcome
All the bullet sprites are drawn on the screen after the bat has been killed.	Left mouse click.	This will restore the ammo for the next bat.	The bat will disappear, and all the bullet sprites will be drawn on the screen.	The bat disappeared but only two of the bullet sprites were drawn on the screen.

I check that I was drawing all the sprites on the screen correctly and found no error within the code for that area. Following this I concluded it must have something to do with the "bullet_v" value so I changed it to 7 instead.

```
If KeyHit (KEY_LMB) then
    For Local batl:=Each batl_collectoin
        If intersects(mouseX,mouseY,60,60,500,300,94,51) Then
            batl_collectoin.Remove batl
            ammo2_collection.AddLast(New bullet(156.5, 740))
            ammo1_collection.AddLast(New bullet(206.5, 740))
            ammo3_collection.AddLast(New bullet(106.5, 740))
            bullet_v = 7
        End
    Next
End
```

The bullet variable was changed to 7 instead of 6.

What's being tested?	input	justification	Expected outcome	Actual outcome
All the bullet sprites are drawn on the screen after the bat has been killed.	Left mouse click.	This will restore the ammo for the next bat.	The bat will disappear, and all the bullet sprites will be drawn on the screen.	The bat disappeared, and all the bullet sprites were drawn on the screen.

It was a concern of mine that the left mouse clicking, and the removal of bullets would not work properly by making the "bullet_v" value 7.

What's being tested?	input	justification	Expected outcome	Actual outcome
Three left mouse clicks will remove all three of the bullet sprites.	Left mouse click.	This will ensure the bullet system will work properly for every bat that is spawned.	Each bullet should be removed with one left mouse click for all three of the bullets.	Each bullet was removed with one left mouse click for all three of the bullets.

Candidate Name: [redacted] Candidate Number: [redacted]

When performing this test, I noticed that the bat could be removed using the left mouse click when there was no ammo in the ammo box. This would defeat the purpose of the ammo box. To fix this I just added another condition to the 'if' statement. The condition will stop the bat from being removed when the "bullet_v" has a value of 0.

This is the added condition to stop the bat from being removed when there is no ammo in the ammo box.

```
If KeyHit (KEY_LMB) And bullet_v > 0 Then
    For Local bat1:=Eachin bat1_collectoin
        If intersects(mouseX,mouseY,60,60,500,300,94,51) Then
            bat1_collectoin.Remove bat1
            ammo2_collection.AddLast(New bullet(156.5, 740))
            ammo1_collection.AddLast(New bullet(206.5, 740))
            ammo3_collection.AddLast(New bullet(106.5, 740))
            bullet_v = 7
        End
    Next
End
```

Candidate Name: [redacted] Candidate Number: [redacted]

What's being tested?	input	justification	Expected outcome	Actual outcome
The bat can't be removed when there is no ammo in the ammo box.	Left mouse click.	This will make the game more challenging as the user only a certain number of attempts to hit their target.	The bat won't be removed with one left mouse click along with no ammo in the ammo box.	The bat wasn't removed with one left mouse click along with no ammo in the ammo box.

My next step is to remove health when the bat escapes. I'll do this by setting up a boarder and when the bat goes through it, it will class as escaping and health will be removed. I haven't coded the bats movement, so I'll just make the bat object move up. For ideas on how to do this I looked at the "Craig n Dave" resources again.

DATE: 12/03/18

```
'this is the class for the bullet screen elements:
Class bat
    'sprite image
    Field sprite:Image = LoadImage ("bat.png")

    'variable for the location of the sprite
    Field x:Float
    Field y:Float

    Method New(x_spawn:float,y_spawn:float)
        x = x_spawn
        y = y_spawn
    End

    Method Move(y_distance:float)
        y-=y_distance
    End
End
```

First, I implemented a move method into the bat class so that the bat has the capability of escaping. This will just cause the bat to move up.

Candidate Name: [redacted] Candidate Number: [redacted]

The bat now has a position that changes. Therefore, I changed the input for the bat in the collision detection function from the set values “(500,300)” to the variables “(bat1.x, bat1.y)”. Now the user can click on the bat when moving to remove it.

```
If KeyHit (KEY_LMB) And bullet_v > 0 Then
    For Local bat1:=Eachin bat1_collectoin
        If intersects(mouseX,mouseY,60,60,bat1.x, bat1.y ,94,51) Then
            bat1_collectoin.Remove bat1
            ammo2_collection.AddLast(New bullet(156.5, 740))
            ammo1_collection.AddLast(New bullet(206.5, 740))
            ammo3_collection.AddLast(New bullet(106.5, 740))
            bullet_v = 7
        End
    Next
End
```

```
If KeyHit (KEY_LMB) And bullet_v > 0 and intersects(mouseX,mouseY,60,60,bat1.x,bat1.y,94,51) Then
        bat1_collectoin.Remove bat1
        ammo2_collection.AddLast(New bullet(156.5, 740))
        ammo1_collection.AddLast(New bullet(206.5, 740))
        ammo3_collection.AddLast(New bullet(106.5, 740))
        bullet_v = 7
        bat1_collectoin.AddLast(New bat(500,300))
    End
Next
```

I wanted to include all the current bat logic into one “bat1_collection” for loop. To do this I removed the external “if” statement and placed the conditions from it in to the “if” statement for the intersects function.

```
For Local bat1:=Eachin bat1_collectoin
    bat1.Move(1)
    If bat1.y<10 Then
        bat1_collectoin.Remove bat1
        life_v = life_v -1
        bat1_collectoin.AddLast(New bat(500,300))
    end
    If KeyHit (KEY_LMB) And bullet_v > 0 and intersects(mouseX,mouseY,60,60,bat1.x,bat1.y,94,51) Then
        bat1_collectoin.Remove bat1
        ammo2_collection.AddLast(New bullet(156.5, 740))
        ammo1_collection.AddLast(New bullet(206.5, 740))
        ammo3_collection.AddLast(New bullet(106.5, 740))
        bullet_v = 7
        bat1_collectoin.AddLast(New bat(500,300))
    End
Next
```

Then I placed the “Move” method from the bat class into the loop with the speed of “1”.

I also include an “if” statement that would remove the bat from the screen along with one life if it reached the boarder of the screen and then spawn a new bat.

I was now able to remove the test harness to remove the user’s health.

```
'pressing the key L is a test harness for the players health removal
If KeyHit (KEY_L) Then life_v = life_v -1
```

What's being tested?	input	justification	Expected outcome	Actual outcome
Does the bat move up after being spawned?	N/A	This will make the game more challenging as the bat is harder to shoot	The bat will move up until it is removed at the top boarder of the display	The bat moved up until it was removed at the top boarder of the display

TESTING INPUT DATA

Flying animation

Test data	Expected	Actual
Bat spawned within in the parameters of (1250,850)	Valid	Valid
Bat spawned on the perimeter of (1250,850)	Borderline	Borderline
Bat spawned outside the parameters of (1250,850)	Invalid	Invalid
Bat can "fly" within in the parameters of (1250,850)	Valid	Valid
Bat can "fly" on the perimeter of (1250,850)	Borderline	Borderline
Bat can "fly" outside the parameters of (1250,850)	Invalid	Invalid

What's being tested?	input	justification	Expected outcome	Actual outcome
The first heart is removed when the bat escapes and after a new bat will spawn.	N/A	This will allow the user to lose life making the game challenging.	The first heart will be removed from the health box and a new bat will appear	The fist heart disappeared from the health box and a new bat was spawned.

What's being tested?	input	justification	Expected outcome	Actual outcome
The second heart is removed when the bat escapes and after a new bat will spawn.	N/A	This will allow the user to lose life making the game challenging.	The second heart will be removed from the health box and a new bat will appear	The second heart disappeared from the health box and a new bat was spawned.

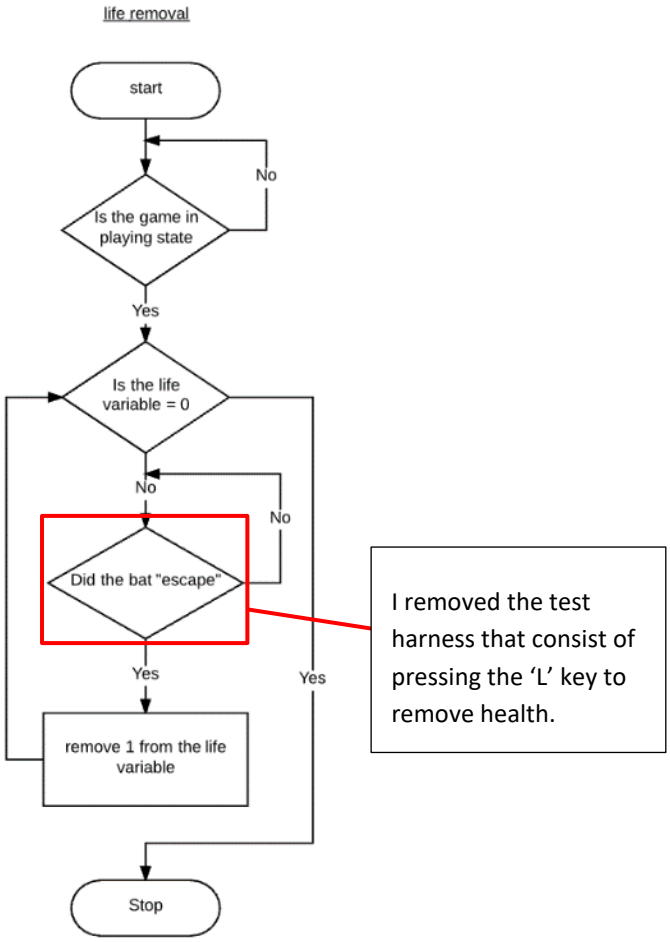
What's being tested?	input	justification	Expected outcome	Actual outcome
After the last bit of health was lost the game state changes to the game over game state.	N/A	This will end the users game when they have lost all their health.	The game will end and the game over screen will be displayed.	The game ended and the game over screen was displayed.

Testing input data

Life removal

Test data	Expected	Actual
The target disappears from screen	Valid	Valid
The target is still on the screen	Invalid	Invalid
“LMC”(left mouse click) on the target	Invalid	Invalid

FLOW CHART



I wanted the ammo box to be rest every time a bat escapes as well. As this is in preparation for the next bat that will be spawned.

DATE: 15/03/18

```
'all the logic for the bat is inside this loop
    For Local bat1:=Eachin bat1_collectoin
        'this is the move method from the bat class causing the bat to move up the screen
        bat1.Move(1)
        If bat1.y<10 Then
            'this is removing the bat from the list "bat1"
            bat1_collectoin.Remove bat1
            'this is decreasing the health variable by 1
            life_v = life_v -1
            'this is spwaning a new bat onto the screen
            bat1_collectoin.AddLast(New bat(500,500))
            'this is drawing all the bullets back onto the screen
            ammo2_collection.AddLast(New bullet(156.5, 740))
            ammo1_collection.AddLast(New bullet(206.5, 740))
            ammo3_collection.AddLast(New bullet(106.5, 740))
            'this is reseting the bullet variable
            bullet_v = 6
        End
```

I added to the "if" statement. When a bat escapes the ammo will also be restored ready for the next bat to spawn.

What's being tested?	input	justification	Expected outcome	Actual outcome
All ammo is restored after a bat escapes.	The bat escapes	This fills up the ammo box ready for the next bat that is spawned.	All the bullet sprites are drawn on the screen again in the ammo box after a bat escapes.	All the bullet sprites were drawn on to the screen again in the ammo box.

I wanted the bat to seem like it was appearing from below like in the “Nintendo: Duck hunt” game. Therefore, I changed the value of the y-coordinate to 500.

The new coordinates for the bat spawn point.

```
batl_collectoin.AddLast(New bat(500,500))
```

Candidate Name: [redacted] Candidate Number: [redacted]

DATE: 18/03/18

INTERVIEW: 18/03/18

No.	Requirements	Justification	☑	Reference
5.	Three shots per mini round	This was done to make the game more challenging, so the game wouldn't become too easy. Easy games can become easily boring	☐	A feature from the Nintendo: Duck Hunt game
9.	If a bat "escapes" the user loses a life	This acts as an indicator for the user as they play the game, so they have an idea of how many lives they have left	☐	Interview: 24/09/17
27.	Use the mouse to click on the moving targets in the game with left click.	This was the decided control for the game as it felt the most appropriate for aiming and shooting the targets	☐	Interview: 24/09/17

Me:
I have coded everything we talked about in are last interview. So, every time a new bat is spawned all the ammo is restored which ticks the requirement for having three shots per mini round. I also managed to remove all the test harnesses I previously put in place. I feel that I have created the basic elements of this game.
Ben Newton:
I too believe you have successfully coded everything that I asked for and you have created the basic elements of this game. However, I think you should stop with the coding and move on with the project dude to the time restraint. You may not have been able to code everything that was planned but good progress was made.

EVALUATION

TESTING FOR THE EVALUATION

START UP

This section is testing the splash screen.

No.	Tests	Input	Expected outcome	Actual outcome	Video reference
30.	A splash screen is displayed for a while then changes to the menu	**N/A**	A splash screen is displayed for a while then changes to the menu	As expected	

SCORING

This section will test the score system of the game. I was unable to implement the score system features into the game, other than placing the score box in the bottom right corner of the screen in the play state, due to time restrictions.

No.	Tests	Input	Expected outcome	Actual outcome	Video reference
1.	The users final score is added to the score board if it is greater than the score boards lowest score	**Valid:** a final score of 1000	**Valid:** the score is placed on the score board next to the user's name	**This feature was not implemented**	**N/A**
		Invalid: the string "quickshot2"	**Invalid:** the name "quickshot2" is displayed twice, once as the name and once as the score	**This feature was not implemented**	**N/A**
2.	The users name can be used for the score board if the name is greater than 2 characters and less than 21 characters	**Valid:** the string "duck321"	**Valid:** the name "duck321" is placed next to the users final score on the score board	**This feature was not implemented**	**N/A**
		Invalid: the string "ya"	**Invalid:** the name "ya" is placed next to the users final score on the score board	**This feature was not implemented**	**N/A**

3.	The player gains 10 points when a bat is shot	**Valid:** A left mouse click on the target when the bullet variable isn't 0	**Valid:** 10 points are added to the users score variable	**This feature was not implemented**	**N/A**
		Invalid: A left mouse click within the parameter of (1250,850) when the bullet variable isn't 0	**Invalid:** 10 points are taken away from the users score variable	**This feature was not implemented**	**N/A**
4.	Score box in the bottom right of the screen.	**N/A**	The score box is in the bottom right of the screen	As expected	
5.	Score board is shown at the end of each game	**N/A**	Score board is shown at the end of each game	**This feature was not implemented**	**N/A**
		Invalid: the game has switched to the menu state	**Invalid:** the game state switches from the menu state to the score board state	**This feature was not implemented**	**N/A**
29.	Back ground will change every time the score box's value increase by 100 points. These back grounds will be of different locations	**Valid:** Score becomes a multiple of 100	**Valid:** the background changes according to the multiple of 100	**This feature was not implemented**	**N/A**
		Invalid: Score is a integer that is not a multiple of 100	**Invalid:** the background doesn't change	**This feature was not implemented**	**N/A**

Candidate Name: [redacted] Candidate Number: [redacted]

SCREEN ELEMENTS

This section will test the screen elements in the game.

No.	Tests	Input	Expected outcome	Actual outcome	Video reference
6.	Three shots per mini round	**Valid:** the bullet variable will be set to 3	**Valid:** when the game is initialized the bullet, variable is set to 3	As expected	
		Invalid: the bullet variable will be set to a string	**Invalid:** when the game is initialized an error, message will appear	As expected	
7.	Ammo disappears as the player shoots at the bats until there is no ammo left so the player can no longer shoot.	**Valid:** left mouse click within the parameter of (1250,850) when the bullet variable isn't 0	**Valid:** a bullet is removed from the ammo box	As expected	
		Invalid: left mouse clicks outside the parameter of (1250,850) when the bullet variable isn't 0	**Invalid:** no bullets are removed from the ammo box	As expected	
8.	Shot box, with three bits of ammo, in the bottom left of the screen.	**N/A**	The ammo box Is in the bottom left corner	As expected	
9.	Three lives in total	**Valid:** the life variable will be set to 3	**Valid:** when the game is initialized the life, variable is set to 3	As expected	
		Invalid: the life variable will be set to a number other than 3	**Invalid:** when the game is initialized the life, variable is set to a number other that 3	As expected	
10.	If a bat "escapes" the user loses a life	**Valid:** a bat has "escaped"	**Valid:** 1 is removed from the life variable	As expected	

11.	Life box, has three lives inside, in the middle of the bottom of the screen.	**N/A**	The ammo box is in the bottom left corner	As expected	
17.	Sound effect for the shooting. This will occur when left mouse button is clicked	**Valid:** A left mouse click within the parameter of (1250, 1090) when the bullet variable isn't 0	**Valid:** a gunshot sound is played	As expected	
		Invalid: A left mouse click within the parameter of (1250, 1090) when the bullet variable is 0	**Invalid:** no sound is played	As expected	
		Borderline: A left mouse click on the perimeter of (1250, 850) when the bullet variable isn't 0	**Borderline:** a gunshot sound is played	As expected	

NAVIGATION

This section will test the navigation through the game. Due to time restraints and my unfamiliarity of how to implement navigation with a mouse in the menu state I did not implement this function.

No.	Tests	Input	Expected outcome	Actual outcome	Video reference
14.	Menu with the title and three options, being the one and two player modes and the game controls option.	**N/A**	The menu will have three options, being the one and two player modes and the game controls option.	As expected	
24.	Press the 'm' key to access the menu	**Valid:** when in the playing state "m" is pressed	**Valid:** the game state will change from playing to the menu state	As expected	
		Invalid: when in the playing state "r" is pressed	**Invalid:** the game will continue in the playing state	As expected	
25.	Press the 'p' key to enter the paused game state	**Valid:** when in the playing state "p" is pressed	**Valid:** the game state will change from playing to the paused state	As expected	
		Invalid: when in the playing state "y" is pressed	**Invalid:** the game will continue in the playing state	As expected	
26.	Press the 'p' key to re-enter the playing game state	**Valid:** when in the paused state "p" is pressed	**Valid:** the game state will change from paused to the playing state	As expected	
		Invalid: when in the playing state "m" is pressed	**Invalid:** the game state will change from paused to the menu state	As expected	
28.	Use the mouse to navigate on the menu screen	**Valid:** the user is able to left mouse click on the different game options to select	**Valid:** the user can select different game options using the mouse	**This feature was not implemented**	**N/A**
		Invalid: enter is pressed to select a game option	**Invalid:** nothing will happen	**This feature was not implemented**	**N/A**

BAT

This section will test the bats features in the game. I was unable to implement all the bat features into the game, other than allowing the user to click on the bat to remove it, due to time restrictions.

No.	Tests	Input	Expected outcome	Actual outcome	Video reference
12.	A counter is started when the bat starts to fly around. It decreases until 0. This will act as a timer	**Valid:** Counter resets after new target spawns	**Valid:** the counter decrease until 0	**This feature was not implemented**	**N/A**
		Invalid: Counters value is a non-integer/negative value	**Invalid:** the counter will become negative	**This feature was not implemented**	**N/A**
13.	The bat flies around until score counter becomes 0 then the sprite disappears/ “escapes”	**Valid:** the counter has decrease to 0	**Valid:** the bat “escapes”	**This feature was not implemented**	**N/A**
		Invalid: the counter becomes negative	**Invalid:** the bat will just keep flying around with no end	**This feature was not implemented**	**N/A**
15.	As game goes on the bats become faster. This happens every multiple of 100	**Valid:** Score becomes a multiple of 100	**Valid:** the time between each line of execution is decreased	**This feature was not implemented**	**N/A**
		Invalid: Score is an integer that is non-integer/negative value	**Invalid:** the speed at which the bat moves doesn’t change	**This feature was not implemented**	**N/A**
19.	Sound effect for the bat flying around	**Valid:** the bat has been spawned within the game screen (1250,850)	**Valid:** a flapping sound is played when the bat is “flying” around the screen	**This feature was not implemented**	**N/A**
		Invalid: the bat has been left mouse clicked on when the shot variable isn’t 0	**Invalid:** hurling sounds are played on loop	**This feature was not implemented**	**N/A**
20.	Sound effect for the bat falling	**Valid:** bat hurling sound has been played	**Valid:** a whistling sound is played	**This feature was not implemented**	**N/A**

		Invalid: "m" has been pressed	**Invalid:** the game state changes from playing to the menu state	**This feature was not implemented**	**N/A**
21.	The animation for the bat being "killed". A hurling sound is played	**Valid:** the bat has been left mouse clicked on when the shot variable isn't 0	**Valid:** A "dying" bat animation is played	**This feature was not implemented**	**N/A**
		Invalid: the bat has been clicked on when the shot variable is 0	**Invalid:** No animation is played	**This feature was not implemented**	**N/A**
22.	The animation for the bat flying around	**Valid:** the bat has been spawned within the game screen (1250,850) or the bat can "fly" within in the parameters of (1250,850)	**Valid:** a flapping animation is played when the bat is "flying" around the screen	**This feature was not implemented**	**N/A**
		Invalid: Bat spawned outside the parameters of (1250,850) or Bat can "fly" outside the parameters of (1250,850)	**Invalid:** the bat doesn't appear on screen	**This feature was not implemented**	**N/A**
		Borderline: Bat spawned on the perimeter of (1250,850) or Bat can "fly" on the perimeter of (1250,850)	**Borderline:** a flapping animation is played when the bat is "flying" around the screen	**This feature was not implemented**	**N/A**
23.	The animation for the bat falling	**Valid:** bat "dying" animation has been played	**Valid:** bat "falling" animation is played	**This feature was not implemented**	**N/A**
		Invalid: "3" has been pressed	**Invalid:** no animation is played	**This feature was not implemented**	**N/A**
27.	Use the mouse to click on the moving targets in the game with left click.	**Valid:** LMC on the bat when there is still ammo in the ammo box	**Valid:** when the user clicks on the bat when they have ammo it is removed of screen	As expected	

		Invalid: enter is pressed to "shoot"	**Invalid:** nothing happens	As expected	

WHOLE GAME

This is testing whether the game has the correct look for the target audience.

No.	Tests	Input	Expected outcome	Actual outcome	Video reference
16.	All sprites and backgrounds will be in a pixel art style	**N/A**	All sprites and backgrounds will be in a pixel art style	As expected	

USABILITY FEATURES

Here I will annotate an image of a game highlighting all its useable features. This will be followed by a brief explanation from me and then a statement from the stakeholder, Ben Newton.

NAVIGATION

Me:

As we discussed previously in a former interview the user should have been able to navigate the menu using the mouse. However, due to my unfamiliarity of how to implement this and time restraints I decide to use keys instead. Hence, it was not made clear to the user what keys to press to navigate the menu. Given more time this problem could easily be fixed by either changing the sprites used or implementing code to use a mouse to navigate the menu.

Ben Newton:

I remember us discussing this and I agree with everything that you said but this would not be acceptable as a final game as it is not very clear what buttons to press for navigation. However, using the keys is simple and could replace the use of the mouse to navigate.

To re-enter the one player modes, press the key “p” again.

Me:

The purpose of the pause state is just to stop the game and continue playing afterwards. Therefore, all you can do in this paused state is press the “p” key to return to the playing state. I included a brief instruction of how to resume the game making what to do clear to the user. Furthermore, I also chose the key “p” as this is used to enter the paused state simplifying the transition between the playing and paused state.

Ben Newton:

This is perfect as it is not complicated to perform a simple change in states and you even make it clear how to return to the playing state. I have no issues with this.

To re-enter the menu the escape key is pressed.

Me:

So that the user knows how to play the actual game, instructions where included on how to enter the paused state or the menu state and how to play the game. The user would now know how to enter these other game states in game or even know that there is the possibility of entering these other game states when in the playing state. The controls game state is only linked to the menu so all you can do in this state is press the escape key to return to the menu state.

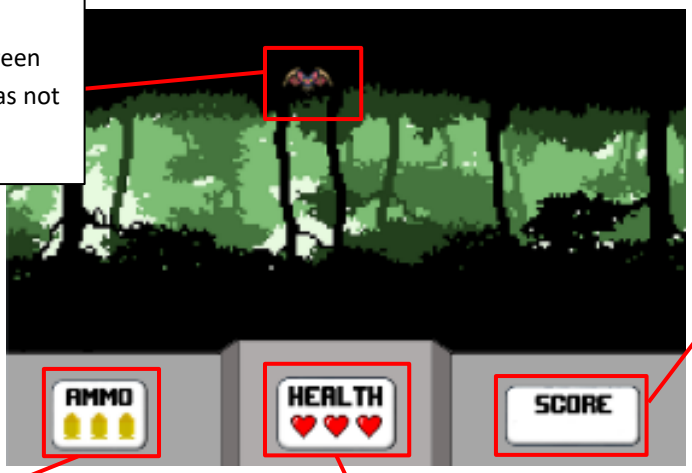
Ben Newton:

Once again, it is good that you are making the games features clear to the user because otherwise the user would have not known about these features without trying every key on the key board in the playing state. However, it would have been better if you placed a brief instruction telling the user what key to press to get back to the menu.

Candidate Name: [redacted] Candidate Number: [redacted]

SCREEN ELEMENTS

The bat only moves straight up off the screen as an AI for the bat has not been implemented.



This would have displayed the users live score during the game.

The bullets disappear, right to left, from the ammo box when the user LMC on the area in which the bat can move. A gun shot sound is also played when there is a LMC and there are bullets in the ammo box.

Every time the bat goes off the screen a heart is removed, right to left, from the health box. When the last heart is removed the game over display is shown for a short period of time then the game state changes back to the menu state. A sound effect is played when a heart is removed from the health box.

Me:

The Ammo, Health and score boxes all act as indicators for the user as they play the game. This makes sure that the player knows exactly what is going on in the game making the playing experience easier. However, I have not fully implement all the score box's features due to time restrictions. The AI for the bat hasn't been implemented due to time restrictions so it just moves straight up.

Ben Newton:

The boxes are good indicators for the player as they are placed conveniently and aren't complicated to read meaning the users attention would be drawn completely away from trying to shoot the targets. I would have liked the bat to have AI movement, but I understand with more time you would have been able to implement this. However, so far, the usability of the actual game is great.

CONTROLS

Me:

The controls for playing were kept very simple as the user just needs to use the mouse and a LMC to shoot at the target. The game controls are also briefly explained in the controls game state making the in-game controls clear. The mouse control did not need much explanation as the user will already been familiar with how to use the mouse due to the GUI on the computer.

Ben Newton:

The in-game controls are very simple, so I agree it doesn't need much explanation. The mouse is also easy to use to click on the moving target, so I am happy with the usability of this feature.

HOW WELL DOES THE SOLUTION MATCH THE SUCCESS CRITERIA?

The following will be looking all the requirements created in the research stage. I will be looking at how well the solution matches the requirements, talking about any changes that were made to the design of the game during the development process and how I would have I would have approached the completion of any unmet requirements.

START UP

No.	Requirements	Justification
1.	Start-up screen.	This is done to show the user what game they are about to play

This requirement was met because when the game starts a splash screen appears for a short period of time, displaying the title of the game, then the menu appears. Testing is found in the development stage pages 98-101.

SCORE SYSTEM

No.	Requirements	Justification
2.	The player gains 10 points when a bat is shot	This gives the user points that will be added on to their score
3.	Score box in the bottom right of the screen.	This would place the indicator in a clear place for the user to see
4.	Score board is shown at the end of each game.	This gives the user a goal to work towards making the game more interesting
29.	Back ground will change every time the score box's value increase by 100 points. These back grounds will be of different locations	This will give the game a hint of "story telling" making it more unique and fun to play

The only requirement that was partly met for score system was the placement of the score box, requirement 3. This was not fully met because only the actual box was placed in the bottom right screen when creating the sprites, but there is no value inside the box that increases when the user gains points. The testing can be seen in the development stage pages 78-81.

If given more time I would have been able to implement the other requirements.

To implement requirement 2, I would have had a score variable. Every time the user removed a bat of the screen using a LMC 10 would have been added to the score variable.

To implement requirement 4, I would have added another game state in which a text file was read and displayed on the screen when the game state is entered. Any alterations to the high score table would be saved in the text file so that the high scores aren't lost.

To implement requirement 29, an 'if' statement would be implemented so that every time the score variable is divisible by 100 the image loaded in for the in-game background would change. The images used would have either been randomly selected or in a sequence.

SCREEN ELEMENTS

No.	Requirements	Justification
5.	Three shots per mini round	This was done to make the game more challenging, so the game wouldn't become too easy. Easy games can become easily boring
6.	Ammo disappears as the player shoots at the bats until there is no ammo left so the player can no longer shoot.	This acts as an indicator for the user as they play the game, so they have an idea of how many shots they have already used
7.	Shot box, with three bits of ammo, in the bottom left of the screen.	This place the indicator in a clear place for the user to see
8.	Three lives in total	This was done to make the game challenging as it gives the user a way to lose the game
10.	Life box, has three lives inside, in the middle of the bottom of the screen.	This place the indicator in a clear place for the user to see
30.	When the player loses all their health a game over screen appears	It is a clear indicator for the user
31.	A sound effect for when health is lost	It is a clear indicator for the user
16.	Sound effect for the shooting. This will occur when bullets are used.	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.

I was able to implement all the requirements for the screen elements.

Requirements 7 and 10 were met as both boxes were placed in the correct place when the games sprites where being created. Furthermore, both boxes have sprites that can be removed sequentially given the conditions that cause this to happen. Testing for this is in the development stage pages 78-81.

Requirement 8 was successfully completed as when the user lets the bat escape three times when in game the game finishes meaning they only have three lives. Testing for this can be seen pages 114-115 in the development stage.

Requirement 5 was fully completed because every time a bat is killed or escapes this counts as a mini round. When either of these two things happen all the ammo is restored, the user is given three bullets to use. The testing can be seen in the development stage pages 145-146 and 153.

Requirement 6 was done perfectly, as the user LMCs on the screen the bullets in the ammo box disappear from right to left until all the bullet sprites have been removed from the box. This indicator works very well because at the same time the user can no longer kill the bat. Testing for this can be seen in the development stage pages 111-119 and 139-141.

Requirement 30, this was achieved as when the player's life variable becomes 0 the game state changes to the game over state. The game over state displays for a short while and then it would have changed to the high score table but because this was not implemented the game state changes to the menu state. The testing for this can be seen pages 120-122 in the development stage.

Requirements 31 and 16 were completed, when these sprites were removed from the score boxes due to certain conditions, each time the sound effect for health loss was played for the heart removal and the sound effect for ammo loss was played for the bullet removal. Furthermore, the bullet removal sound effect was no longer played when there was no ammo left in the ammo box. The testing for this can be found in the development state pages 126-130.

BAT

No.	Requirements	Justification
9.	If a bat "escapes" the user loses a life	This acts as an indicator for the user as they play the game, so they have an idea of how many lives they have left
11.	A counter is started when the bat starts to fly around. It decreases until 0. This will act as a timer	This gives the user time to react and shoot the bat and try to shoot the bat again if they miss there shot
12.	The bat flies around until score counter becomes 0 then the sprite disappears/ "escapes"	This will make the game challenging as you must try to shoot the bat before it disappears
14.	As the game goes on the bats become faster. This happens every multiple of 100	This will make the game increasingly difficult hence making the game more interesting
17.	Sound effect for the bat being "killed". A hurling sound is played	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.
18.	Sound effect for the bat flying around	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.
19.	Sound effect for the bat falling	This will make the game more unique and help as an indicator for the user as sometimes visuals can go unnoticed. It also makes the game more exciting.
20.	The animation for the bat being "killed".	This will make the game more unique and help as an indicator for the user when playing.

21.	The animation for the bat flying around	This will make the game more unique and help as an indicator for the user when playing.
22.	The animation for the bat falling	This will make the game more unique and help as an indicator for the user when playing.
27.	Use the mouse to click on the moving targets in the game with left click.	This was the decided control for the game as it felt the most appropriate for aiming and shooting the targets

Once again due to time restriction I was not able to implement all the features for the bat other than the features that allow the basic form of the game to work.

The requirements that were met where requirements 9 and 27.

Requirement 9 was completed well as whenever a bat is spawned into the game it moves straight up meaning that if the player does not shoot it, the bat will be removed of the screen at the border of the screen. This removes a life from the score box. This occurs for every heart except for the last heart as this causes the game state to change to game over game state.

Requirement 27 was easily met as the GUI already allows the user to move the mouse and click on objects on the screen. This requirement was met as when the user LMCs on the bat there is an interaction causing the bat to be removed of the screen.

If I had more time I would have implemented the following requirements in the given ways.

Requirements 11 and 12 are both concerned with how long the bat flies around until it escapes. For requirement 11 I would have placed the counter loop inside the method for the game logic. So, when a bat is spawned a counter which decreases will begin giving the user a time frame in which to shoot the bat. Following requirement 11, requirement 12 would be set into to motion when the counter becomes 0. The bat will then immediately stop flying round and move straight up so that the bat will be removed of the screen which is the bat escaping.

To complete requirement 14, I would have implemented a function that would keep dividing the score variable by 100. Whenever this function returned an integer this would cause the speed at which the bat changes coordinates to increase which in turn increases the speed at which the bat moves.

Requirements 17, 18 and 19 are all about sound effects. For all three of these requirements I would have loaded a hurling, a flying and a falling sound into game in the ‘Oncreate’ method.

The sound for requirement 17 would play straight after the gun shot sound because only one sound can play at a time making this the logical way to place these two sound effects around.

The sound effect for requirement 18 would play on loop until the bat is either killed or escapes off the screen.

The sound for requirement 19 would play after the bat has been left mouse clicked because after this the bat will move straight down. Hence the sound is played as this gives the illusion the bat is falling.

Requirements 20, 21 and 22 are all animations for the bat. To complete the requirement for 20, 21 and 22 I would first have to load the images that would be used for the animations in the “Oncreate” method to later be used in my solution.

To implement the requirement 20, I would replace the current image is being used for the bat with a dead bat image taking the coordinates of the bat before and spawning the dead bat on to these coordinates.

To implement requirement 21, I would have to create a loop which would switch the image being used for the bat constantly with three images. There would be two variables, one for the x and the other for the y, for the coordinates of the bat that would be used to place the all three of the sprites in the correct position giving the illusion that the bat is flying round the screen.

To implement requirement 22, I would replace the sprite image from the dead bat image to the falling bat image. The bat would then move down until the bat is removed of the screen using a function causing the y coordinate to increase making the sprite move down.

NAVIGATION

No.	Requirements	Justification
13.	Menu with the title and three options, being the one player mode, two player mode and controls description.	This will create a link between the one player and two player modes
23.	Press the 'm' key to access the menu from the playing state.	This will allow the user to easily switch between the one and two player game modes
24.	Press the 'p' key to enter the paused game state form the game state.	This makes the game more convenient as the players have the freedom to stop playing whenever and continue straight away with no difficulty
25.	Press the 'p' key to re-enter the playing game state form the paused state.	Allow the user to resume playing the game
26.	Press the escape key to re-enter the menu game state from the playing state.	This will allow the user to return to the menu state from the control state
28.	Use the mouse to navigate on the menu screen.	This was the decided as a control for navigating around the menu as it is already used else were in the game. This is done for continuity purposes.
32.	Game music for some of the game states.	Makes the game more interesting by giving the game a bit more depth with sound.

I was able to implement all these requirements except for requirement 28. When I first started out with this project I was very un-familiar with how to code in monkey-X, so I was unable to get the user to navigate the menu using the mouse. Therefore, instead of using the mouse to navigate keys were used. Here are what keys were used and what for: key "1" was used to access the one player mode; key "2" was used to play the denied access sound; key "c" was used to access the controls description. The testing for this can be found pages 90-93, in the development stage.

However, given the fact that I have being coding in this language for a while now I would be able to implement this feature now. To do this I would have loaded each of the menus options as sprites. This would be done so that when the user's mouse over laps the sprites and there is a LMC this will cause the game states to change or in the special case for the two-player option a denied access sound will be played.

Requirements 23, 24, 25 and 26 are all very similar. All these requirements were met exactly how they were described in the requirements. The game states have specific links, so the user can only use certain keys to access other game states depending on which game state they are in. The testing for this can be found pages 90-93, in the development stage.

Requirement 13 was easily met because when the game is in the menu state the user is given three options to choose from. Two options allow the user to access the one player mode and the other allow the user to look at the game controls. The two-player option just plays a sound indicating that there is no access to this feature.

Requirement 32 was met reasonably well because when the user enters the menu state there is a low frequency sound that plays on a loop to mimic background music. I was unable to find any game music so was unable to implement the music correctly into the game. Furthermore, I would have like to implement some type of music into the high score table, but I did not implement the high score table, so I didn't. Given more time I would have implemented sound when the high score table is displayed. The testing for this in the development stage pages 133-134.

WHOLE GAME

No.	Requirements	Justification
15.	All sprites and backgrounds will be in a pixel art style	This was the decided theme as it allows the game to keep some realism but remove any major gore. This keeps the range of target audience large.

This requirement was easily achieved as when creating the sprites for the game I just ensure that all the sprites used where images that where drawn in a pixel art style. Furthermore, all the titles and game options were also in a pixel art style.

MAINTENANCE

CURRENT AND FUTURE MAINTENANCE

I have not implemented any maintenance features in to the game however I understand that if this game where to be professional distributed these features would be required to fix bugs that are found or update the game by adding more features.

My solution is not complete, but upon completion I would add a feature into the game that would allow the user to check for any updates, this feature could be placed into the menu. I would also need to set up some sort of form that would allow people to report problems they have found in the game which I, or a team of people, would fix and then release as a patch to remove the found issues. Furthermore, depending on the success of the game I could code more features into it, this would make the game playable for longer as the target audience would be entertained for longer by the game.

LIMITATIONS AND HOW THEY WOULD BE APPROACHED

The following consists of a series of limitations followed by how they would be approached.

Requirement	Justification
Give the mouse cursor an aiming icon.	This would also make the game more unique due to its design and it will making the aiming easier for the user.

To complete this additional requirement, I would create a sprite that was shaped like a cross within a circle. The coordinates of where the mouse is will be constantly used to update the position of the aiming sprite. This will cause the sprite to follow the cursor.

Requirement	Justification
Vampire guy pops up and laughs when you lose a life.	This would make the game visual more interesting

To complete this requirement, I would first have to create the sprites for the vampire guy that would make him look like he is laughing for the animation in the pixel art style. The vampire animation and sound effect would be triggered every time a player loses a life as described in the requirement. The animation would be created by spawning the sprite of screen, then moving it up into the screen and replacing the vampire sprite with other sprites, in a given sequence, to give the illusion he is laughing. The vampire would then move down off the screen again and be removed.

Requirement	Justification
Two player modes.	This would create a local two player competitive mode which will give the user more content

To complete this requirement, I would develop two game modes. One would be two users trying to shoot the same bats, each user has their own mouse, score, ammo and to win the game one of the users must reach a certain score. The other game mode would consist of one user playing as the bat and the other user trying to shoot the bat, the bat would be controlled using the keyboard. For the user playing as the bat to win they need three escapes and for the user shooting at the bat to win they need to achieve a certain score.

Requirement	Justification
Two mice can be used per computer	This will allow a two-player mode that has two people using their own mice to shoot at the bats

To implement this requirement, I would have to find additional software or create additional software that would allow two mice to use on screen at one time. This will allow one of the two player modes to work.

Requirement	Justification
The game will have many objects interacting on the screen on screen	This will make the game more interesting to play as more is happing at any one time.

Currently the game is very simple, but if I wanted to include more things such as multiple bats on screen, doves that would give the player health when clicked on, moving backgrounds, zombie bats that remove life when clicked on I would need to have a computer will a better GPU to do this as the one I am currently using isn't very good.

Requirement	Justification
Impressive visuals	This will make the game more appealing to a wider audience.

To implement this requirement, I would need additional software such as Pyro. This would allow me to add lighting effects and shadow into the game making it look more astatically pleasing.

PROJECT APPENDIXES

CODE LISTINGS

```
'version 5.0
'Libraries and globals
Import mojo
Global Game:Game_app
'****************************************************************************
'Main program starts here:
Function Main ()
    Game = New Game_app
End
'****************************************************************************
'All game code goes here:
Class Game_app Extends App

    'declaring mouse coordinates
    Field mouseX:Int
    Field mouseY:Int

    'declaring the health and bullet variables
    Field life_v:Int
    Field bullet_v:Int

    'declaring images for game states
    Field menu:Image
    Field controls:Image
    Field paused:Image
    Field gameover:Image
    Field start_up:Image

    'declaring sound for the game
    Field gunshot:Sound
    Field healthloss:Sound
    Field gameoverS:Sound
    Field pausein:Sound
    Field pauseout:Sound
    Field menuS:Sound
    Field menuSnegative:Sound
    Field menuM:Sound
    Field intro:Sound

    'declaring images for playing state backgrounds
    Field backf:Image
    Field backc:Image
    Field backh:Image
    Field backm:Image
```

```
'declaring playing state sprites

    'declaring the bat list
    Field batl_collectoin:List<bat>

    'declaring the three lists for health
    Field health1_collection:List<heart>
    Field health2_collection:List<heart>
    Field health3_collection:List<heart>

    'declaring the three lists for ammo
    Field ammo1_collection:List<bullet>
    Field ammo2_collection:List<bullet>
    Field ammo3_collection:List<bullet>

    'declaring the initial game state
    Global gamestate:String = "START_UP"
'*****************************************************************************
    'All the initialisation for the whole game goes here:
    Method OnCreate ()

        'frame rate at which the game runs
        SetUpdateRate 60

        'loading sounds
        gunshot = LoadSound ("sfx_weapon_singleshot21.wav")
        healthloss =  LoadSound ("sfx_sounds_button8.wav")
        gameoverS =  LoadSound ("sfx_sound_mechanicalnoise3.wav")
        pausein = LoadSound ("sfx_sounds_pause1_in.wav")
        pauseout = LoadSound ("sfx_sounds_pause1_out.wav")
        menuS = LoadSound ("sfx_menu_move4.wav")
        menuSnegative = LoadSound ("sfx_sound_neutral8.wav")
        menuM = LoadSound ("sfx_sounds_interaction8.wav")
        intro = LoadSound ("sfx_wpn_cannon2.wav")

        'game state images
        start_up = LoadImage ("start_up.png")
        paused = LoadImage ("paused.png")
        controls = LoadImage ("controls.png")
        menu = LoadImage ("menu.png")
        gameover = LoadImage("gameover.png")

        'diffrent backgrounds for playing state
        backf = LoadImage ("forest background.png")
        backc = LoadImage ("castle background.png")
        backh = LoadImage ("hill background.png")
```

```
    End
'*****************************************************************************************************************
    'All the game logic goes here:
    Method OnUpdate ()
        'This is the code to allow the user to change between gamestates
        Select gamestate
'*****************************************************************************************************************
            'this is the start_up state:
            Case "START_UP"
                'this is playing the start up sound
                PlaySound(intro,1,0)
                'this is code to pause the display of the splash screen
                For Local x:Int = 0 To 90000
                    x = x + 1
                    For Local y:Int = 0 To 20000
                        y = y + 1
                    Next
                Next
                'this is changing game states
                gamestate = "MENU"
                'this will play the menu music
                PlaySound(menuM,1,1)
'*****************************************************************************************************************
            'this is the menu state:
            Case "MENU"
                If KeyHit (KEY_C) Then
                    'this is changing game states
                    gamestate="CONTROLS"
                    'this is playing the menu select sound effect
                    PlaySound(menuS,1,0)
                Endif

                If KeyHit (KEY_1) Then
                    'this is changing game states
                    gamestate="INITIALISE"
                    'this is playing the menu select sound effect
                    PlaySound(menuS,1,0)
                Endif

                If KeyHit (KEY_2) Then
                    'this is playing the negative menu select sound effect
                    PlaySound(menuSnegative,1,0)
                    'this is giving the sound effect time to play before returing to the playing of menu music
                    For Local x:Int = 0 To 10000
                        x = x + 1
                        For Local y:Int = 0 To 10000
```

```
                    x = x + 1
                    For Local y:Int = 0 To 10000
                        y = y + 1
                    Next
                Next
                'this is playing the menu music
                PlaySound(menuM,1,1)
            Endif
'*************************************************************************************************************
            'this is the initialise state for playing game:
            Case "INITIALISE"
                'the life and bullet variables are being giving the value 6
                life_v = 6
                bullet_v = 6

            'objects being initialised
                'declaring the bat list
                bat1_collectoin = New List<bat>
                bat1_collectoin.AddLast(New bat(500,500))

                'a heart object is being added to each of the lists
                health1_collection = New List<heart>
                health1_collection.AddLast(New heart(640, 745))
                health2_collection = New List<heart>
                health2_collection.AddLast(New heart(580, 745))
                health3_collection = New List<heart>
                health3_collection.AddLast(New heart(520, 745))

                'a bullet object is being added to each of the lists
                ammo1_collection = New List<bullet>
                ammo1_collection.AddLast(New bullet(206.5, 740))
                ammo2_collection = New List<bullet>
                ammo2_collection.AddLast(New bullet(156.5, 740))
                ammo3_collection = New List<bullet>
                ammo3_collection.AddLast(New bullet(106.5, 740))

                'this is changing game states
                gamestate = "PLAYING"
'*************************************************************************************************************
            'this is the playing state:
            Case "PLAYING"
                'all the logic for the bat is inside this loop
                    For Local bat1:=Eachin bat1_collectoin
                        'this is the move method from the bat class causing the bat to move up the screen
                        bat1.Move(1)
                        If bat1.y<10 Then
```

```
                'this is removing the bat from the list "bat1"
                bat1_collectoin.Remove bat1
                'this is decreasing the health variable by 1
                life_v = life_v -1
                'this is spwaning a new bat onto the screen
                bat1_collectoin.AddLast(New bat(500,500))
                'this is drawing all the bullets back onto the screen
                ammo2_collection.AddLast(New bullet(156.5, 740))
                ammo1_collection.AddLast(New bullet(206.5, 740))
                ammo3_collection.AddLast(New bullet(106.5, 740))
                'this is reseting the bullet variable
                bullet_v = 6
            End

            'this is the logic for when the bat is clicked on with ammo in the ammo box
            If KeyHit (KEY_LMB) And bullet_v > 0 and intersects(mouseX,mouseY,60,60,bat1.x,bat1.y,94,51) Then
                'this is removing the bat from the list "bat1"
                bat1_collectoin.Remove bat1
                'this is spwaning a new bat onto the screen
                bat1_collectoin.AddLast(New bat(500,500))
                'this is drawing all the bullets back onto the screen
                ammo2_collection.AddLast(New bullet(156.5, 740))
                ammo1_collection.AddLast(New bullet(206.5, 740))
                ammo3_collection.AddLast(New bullet(106.5, 740))
                'this is reseting the bullet variable
                bullet_v = 7
            End
        Next

    If KeyHit (KEY_P) Then
        'this is changing game states
        gamestate="PAUSED"
        'this is playing the sound effect for entering the paused state
        PlaySound(pausein,1,0)
    Endif

    If KeyHit (KEY_M) Then
        'this is changing game states
         gamestate="MENU"
         'this is playing the
         PlaySound(menuM,1,1)
     Endif

    'this is playing the health loss sound
    If life_v = 5 Then
        PlaySound(healthloss,1,0)
```

```
            life_v = 4
        Endif
        ' this is removing the contents of "health1" given "life_v"=2
        If life_v = 4 Then
        For Local health1:=Eachin health1_collection
            health1_collection.Remove health1
        Next
        Endif

        'this is playing the health loss sound
        If life_v = 3 Then
            PlaySound(healthloss,1,0)
            life_v = 2
        Endif
        ' this is removing the contents of "health2" given "life_v"=1
        If life_v = 2 Then
        For Local health2:=Eachin health2_collection
            health2_collection.Remove health2
        Next
        Endif

        ' this is changing yhr playing state to the gameover state when no health is left
        If life_v = 1 Then
        'this is changing game states
        gamestate = "GAMEOVER"
        Endif
'#########################################################################################################

        'this is updating the mouse coordinates
        mouseX = MouseX()
        mouseY = MouseY()

        'when the left mouse is clicked and the y mouse -
        '- coordinate is less than 610 the bullet variable will be reduced
        If MouseHit (MOUSE_LEFT) And mouseY<610 Then bullet_v = bullet_v -1

        'this is playing the bullet loss sound
        If bullet_v = 5 Then
            PlaySound(gunshot,1,0)
            bullet_v = 4
        Endif
        ' this is removing the contents of "ammo1" given "bullet_v"=2
        If bullet_v = 4 Then
        For Local ammo1:=Eachin ammo1_collection
            ammo1_collection.Remove ammo1
        Next
```

```
                Endif

                'this is playing the health loss sound
                If bullet_v = 3 Then
                    PlaySound(gunshot,1,0)
                    bullet_v = 2
                Endif
                ' this is removing the contents of "ammo2" given "bullet_v"=1
                If bullet_v = 2 Then
                For Local ammo2:=Eachin ammo2_collection
                    ammo2_collection.Remove ammo2
                Next
                Endif

                'this is playing the health loss sound
                If bullet_v = 1 Then
                    PlaySound(gunshot,1,0)
                    bullet_v = 0
                Endif
                ' this is removing the contents of "ammo3" given "bullet_v"=0
                If bullet_v = 0 Then
                For Local ammo3:=Eachin ammo3_collection
                    ammo3_collection.Remove ammo3
                Next
                Endif

'*************************************************************************************************************************
            'this is the gameover state:
            Case "GAMEOVER"
                'this is playing the game over sound effect
                PlaySound(gameoverS,1,0)
                'this pauses the game on the game over state for a while
                For Local x:Int = 0 To 90000
                    x = x + 1
                    For Local y:Int = 0 To 20000
                        y = y + 1
                    Next
                Next
                'this is changing game states
                gamestate = "MENU"
                'this is playing the menu music
                PlaySound(menuM,1,1)
'*************************************************************************************************************************
            'this is the controls state:
            Case "CONTROLS"
                If KeyHit (KEY_ESCAPE) Then
```

```
                'this is changing game states
                gamestate="MENU"
                PlaySound(menuM,1,1)
                Endif
'*************************************************************************************************************
            'this is the paused state:
            Case "PAUSED"
                If KeyHit (KEY_P) Then
                'this is changing game states
                gamestate="PLAYING"
                'this is playing the sound effect for exiting the paused state
                PlaySound(pauseout,1,0)
                endif
        END
    End
'*************************************************************************************************************
    'All the graphics drawing goes here:
    Method OnRender()
        Select gamestate
            'this is the start_up state:
            Case "START_UP"
                DrawImage start_up, 0,0

            'this is the menu state:
            Case "MENU"
                DrawImage menu, 0,0

            'this is the playing state:
            Case "PLAYING"
                DrawImage backf, 0,0

            'this is drawing all the bats on the screen
                'this is drawing all the objects in the list "bat 1"
                For Local bat1:=Eachin bat1_collectoin
                    DrawImage bat1.sprite, bat1.x, bat1.y
                Next

            'this is drawing all the hearts to the screen
                'this is drawing all the objects in the list "health1"
                For Local health1:=Eachin health1_collection
                    DrawImage health1.sprite, health1.x, health1.y
                Next
                'this is drawing all the objects in the list "health2"
                For Local health2:=Eachin health2_collection
                    DrawImage health2.sprite, health2.x, health2.y
                Next
```

```
            'this is drawing all the objects in the list "health3"
            For Local health3:=Eachin health3_collection
                DrawImage health3.sprite, health3.x, health3.y
            Next

        'this is drawing all the hearts to the screen
            'this is drawing all the objects in the list "ammo1"
            For Local ammo1:=Eachin ammo1_collection
                DrawImage ammo1.sprite, ammo1.x, ammo1.y
            Next
            'this is drawing all the objects in the list "ammo2"
            For Local ammo2:=Eachin ammo2_collection
                DrawImage ammo2.sprite, ammo2.x, ammo2.y
            Next
            'this is drawing all the objects in the list "ammo3
            For Local ammo3:=Eachin ammo3_collection
                DrawImage ammo3.sprite, ammo3.x, ammo3.y
            Next

        'this is the gameover state:
        Case "GAMEOVER"
            'this is drawing the game over display to the screen
            DrawImage gameover, 0,0

        'this is the controls state:
        Case "CONTROLS"

        'this is drawing the controls display to the screen
            DrawImage controls, 0,0

        'this is the paused state:
        Case "PAUSED"
            'this is drawing the pause display to the screen
            DrawImage paused, 0,0
    End

End
End
'*************************************************************************************************************
'this is the class for the heart screen elements:
Class heart
    'sprite image
    Field sprite:Image = LoadImage ("heart.png")

    'variable for the location of the sprite
    Field x:Int
```

```
    Field y:Int

    'this is giving parmeters for the location of the sprites
    Method New(x_spawn:Int,y_spawn:Int)
        x = x_spawn
        y = y_spawn
    End
End
'*****************************************************************************************************************
'this is the class for the bullet screen elements:
Class bullet
    'sprite image
    Field sprite:Image = LoadImage ("bullet.png")

    'variable for the location of the sprite
    Field x:Float
    Field y:Float

    'this is giving parmeters for the location of the sprites
    Method New(x_spawn:float,y_spawn:float)
        x = x_spawn
        y = y_spawn
    End
End
'*****************************************************************************************************************
'this is the class for the bullet screen elements:
Class bat
    'sprite image
    Field sprite:Image = LoadImage ("bat.png")

    'variable for the location of the sprite
    Field x:Float
    Field y:Float

    'this is giving parmeters for the location of the sprites
    Method New(x_spawn:float,y_spawn:float)
        x = x_spawn
        y = y_spawn
    End

    'this will cause the bat to move up
    Method Move(y_distance:float)
        y-=y_distance
    End
End
'*****************************************************************************************************************

'function for the collision detection
Function intersects:Bool (x1:Int, y1:Int, w1:Int, h1:Int, x2:Int, y2:Int, w2:Int, h2:Int)
If x1 >= (x2 + w2) Or (x1 + w1) <= x2 Then Return False
If y1 >= (y2 + h2) Or (y1 + h1) <= y2 Then Return False
Return True
End
```