James Cox

Programing project

Contents

[Introduction: 2](#_Toc474390634)

[Who it is for: 2](#_Toc474390635)

[Game Concept: 2](#_Toc474390636)

[Analysis: 3](#_Toc474390637)

[Success criteria: 3](#_Toc474390638)

[Scope: 4](#_Toc474390639)

[Requirements from client: 4](#_Toc474390640)

[Writing: 4](#_Toc474390641)

[Diagrams: 4](#_Toc474390642)

[Design drawings: 6](#_Toc474390643)

[Physics in the game 7](#_Toc474390644)

[Maths relating to motion of elements on screen: 7](#_Toc474390645)

[Flow chart: 8](#_Toc474390646)

[Potential Tests 8](#_Toc474390647)

[Testing: 9](#_Toc474390648)

[Technical tests 12](#_Toc474390649)

[key elements of the game 13](#_Toc474390650)

[Table of making the program: 15](#_Toc474390651)

[Making the ball bounce was the easy part of making the game because it is simple to do and I have done it before, also making the dictionary was easy, making the quick sort was hard because I have not made or done quicksort before making this program. 18](#_Toc474390652)

[Colours: 18](#_Toc474390653)

[Client review: 19](#_Toc474390654)

[Idea: 19](#_Toc474390655)

[Why do you want it? 19](#_Toc474390656)

[What do you want? 19](#_Toc474390657)

[Question and Answer 19](#_Toc474390658)

[What do you think of the programme? 19](#_Toc474390659)

[Does the programme work? 19](#_Toc474390660)

[Is the programme clear to use? 19](#_Toc474390661)

[Is the programme fun? 19](#_Toc474390662)

[Does the physics work? 19](#_Toc474390663)

[Are the colours ok 19](#_Toc474390664)

[Rate out of ten 19](#_Toc474390665)

[How can it be improved? 19](#_Toc474390666)

[How well the project went: 19](#_Toc474390667)

[Opinion: 20](#_Toc474390668)

[Even better if? 20](#_Toc474390669)

[How I would: 20](#_Toc474390670)

[Possible future extensions: 20](#_Toc474390671)

[21](#_Toc474390672)

[Code relating to prototype 21](#_Toc474390673)

[Subroutines and magic number constants: 24](#_Toc474390674)

[Uml class diagram: 26](#_Toc474390675)

[Program code: 1](#_Toc474390676)

[Ball.cs 28](#_Toc474390677)

[Question.cs 31](#_Toc474390678)

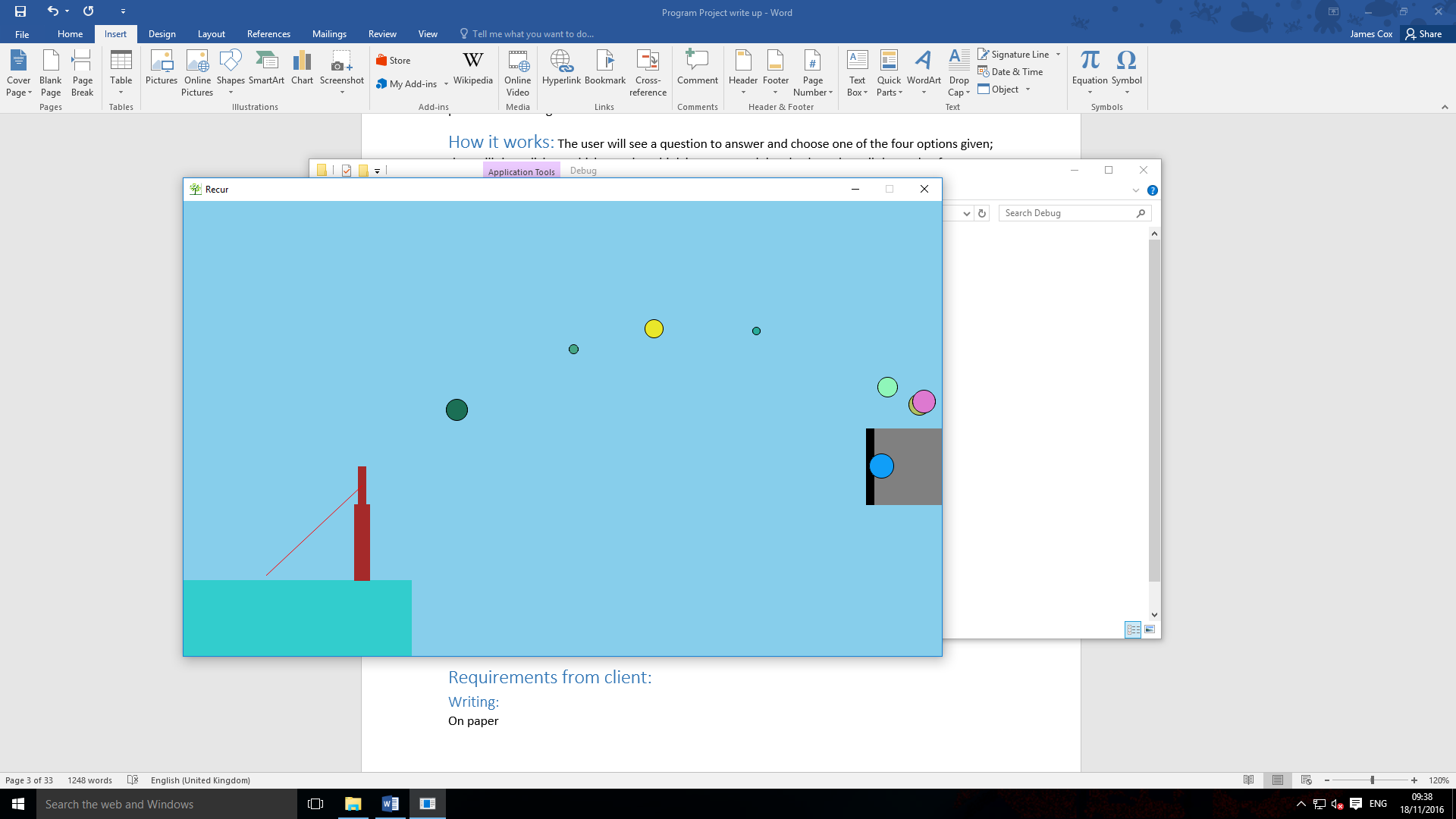
[Awncer.cs 37](#_Toc474390679)

[Awncers.cs 40](#_Toc474390680)

[Quick sort 41](#_Toc474390681)

# Introduction:

This project will be about making a program for someone that will be used to help people get better at using addition, subtraction, division and multiplying while still being fun for ages from 5 upwards. The game will be made on visual studio in c#.



Ball is bouncing off of the wall

The ball is being flung from the stick using the line

Who it is for: Dr Joe Noyes, senior science teacher.

Game Concept: The user will see a question to answer and choose one of the four options given; they will then click on which one they think is correct and drag backwards until the angle of trajectory they have made will send it into the answer box. If the answer is correct then they will get points and if not a new question will appear.

Analysis: The program is a game that helps people who are struggling with basic math skills (adding, subtraction, multiplying and dividing) and keeping them from seeing it as revision, using a game where you throw the Answer at the Answer box as fast and accurately as possible to score points and beat high scores.

Success criteria:

1

S: allow the user to answer questions about addition

M: progress will be measured by their score

A: I will program this feature from the skills I learnt

R: addition questions will be available in the game for the user to see is what I think will be achieved

T: 2-3 days

2

S: allow the user to answer questions about subtraction

M: progress will be measured by the users score

A: I am going to add this into the game using coding in c#

R: subtraction questions will be possible to answer in the game and if answer correctly will score points for the user is what I think will be achieved

T: 1-2 days

3

S: the questions get harder the more correct answers are completed

M: an increase in difficulty when answers are correct

A: harder questions for higher scores from maths books

R: user can get questions based on their skill

T: 7-9 days

4

S: works on windows

M: the program works on windows

A: the program works on windows on any device

R: the program can run on windows efficiently

T: 1-2 days

5

S: starts fast

M: the program starts quickly once opened

A: the program is not slow to start

R: the program loads fast

T: 1-2 days

6

S: use c# to program the game

M: use c# to program the coding more easily

A: the program is programmed in c# and is running smoothly

R: the program is programmed in c#

T: 1-2 months

7

S: the code has no errors

M: the code has no errors when being played

A: the code having no errors as it is running on computer

R: the code does not have errors

T: 1-2 months

Scope: a game that uses maths and physics to help people struggling with basic math skills to be more confident and help with their mental calculations. The game is a single player and is controlled by clicking and dragging the answer to the answer box.

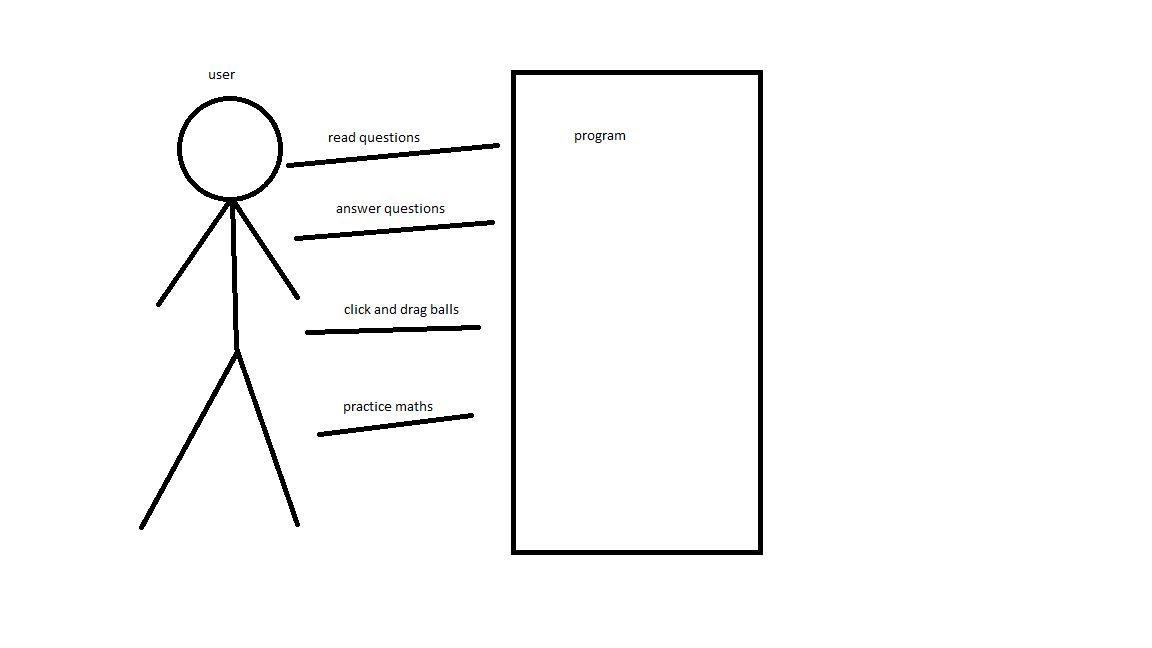
# Requirements from client:

## Writing:

On paper

## Diagrams:

On paper





Design drawings:

Point at tops of stick = 8, 8

Point at end if line = 5, 4

Y speed of ball = 4

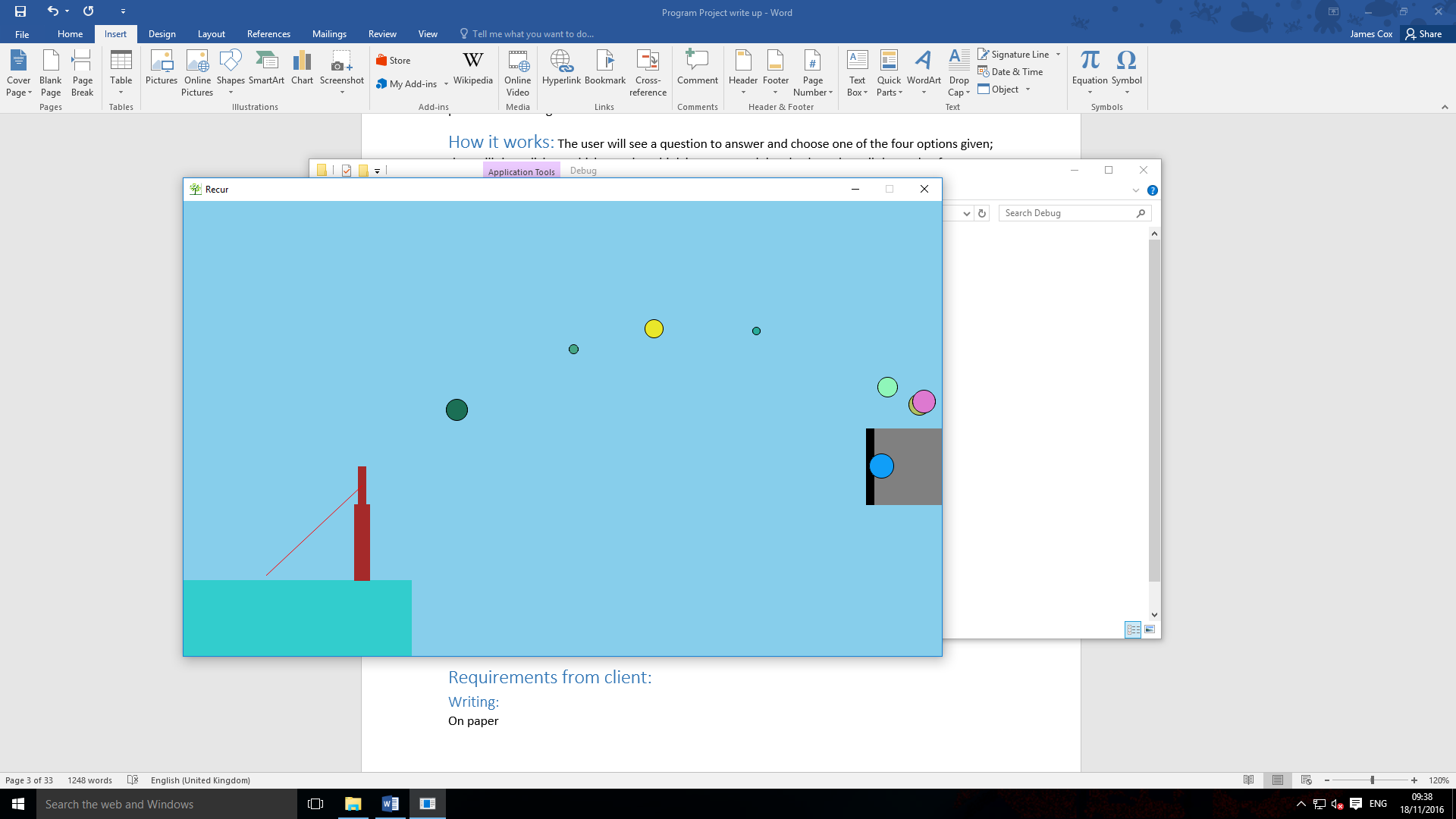
X speed of ball = 3

Calculates how the ball will move

Ball X = X + 2

Ball Y = Y – 1

Moves the ball



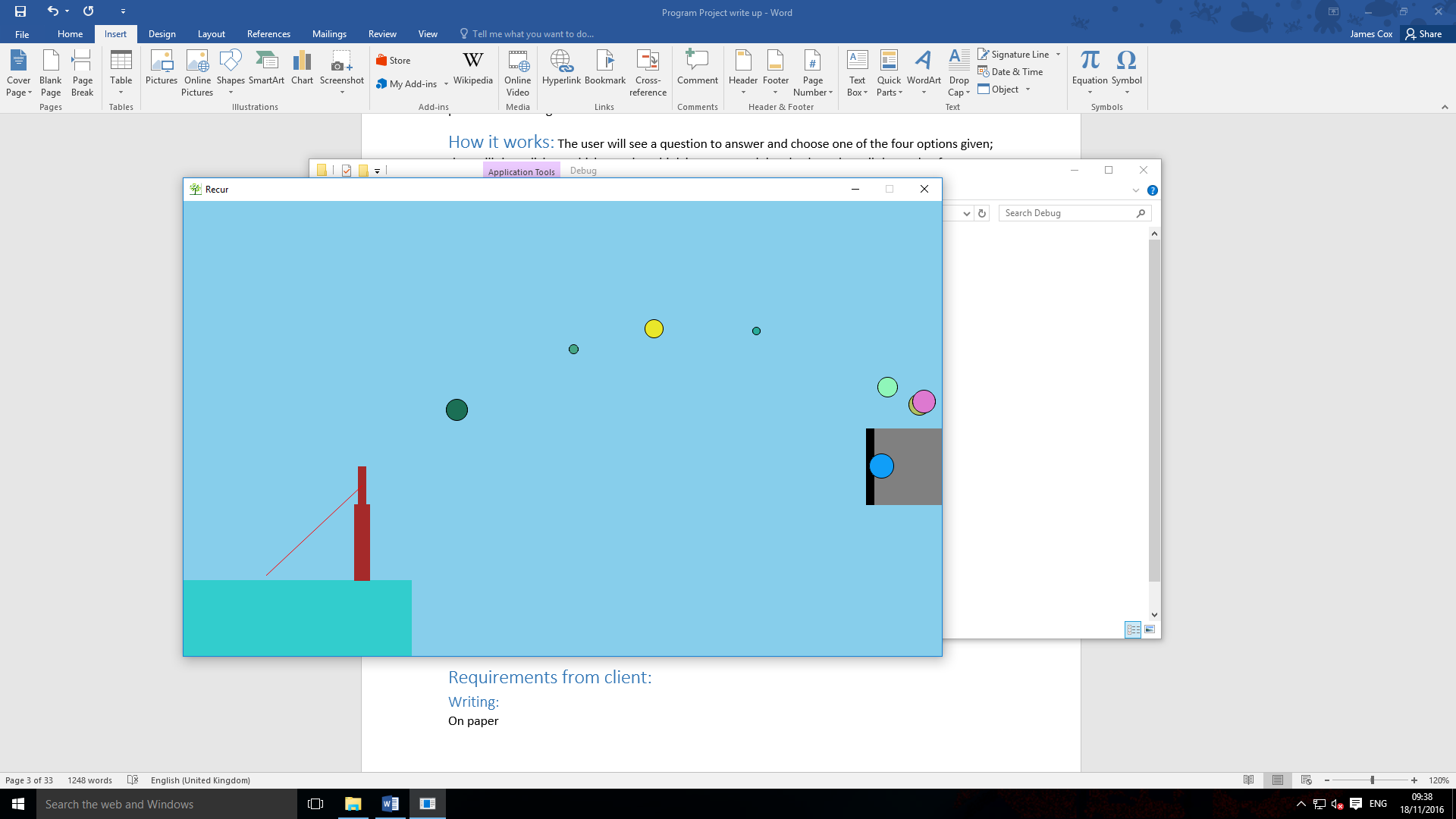
Ball X = X + 3

Ball Y = Y + 4

Moves the ball

This photo is of the design game and only uses basic shapes using in program features and the base code for the ball moving and the pinging feature to send the ball in a direction to control the ball. This is a mock version of the final game. To make the final version I will improve the goal detection by using more accurate numbers to set the collision lines on the answer and questions, graphics of the game by using more detailed sprites and how it will be played and run by increasing the users control of settings about the gameplay so they can use a playstyle that fits them better.

# Physics in the game



Potential energy is converted into kinetic energy

Kinetic energy: kg m2 / s2.the city changes the balls x and y to make the ball look like it is moving when displayed

Gravitational energy is used to make the ball fall

Gravitational energy: P.E. = mgh

The velocity changes the balls x and y to make the ball look like it is moving when displayed

Maths relating to motion of elements on screen:

If the ball touches the edge, then it reverses the relative velocities of the ball so it looks like it bounces off of the wall. Bounce equation: *v = -v. v* is the velocity variable

Depending on how far the player drags their mouse after clicking an answer then set x/y velocity to that.

If x velocity is more than 0 then -1, if x velocity is less than 0 +1. Velocity equation:

If the ball is close to the answer box and set as the correct answer, then do the code for the correct answer

If certain balls y is more than *“x”* set active to false / destroy it

If the like that shows the “angle of trajectory” changes re draw/throw arrow

Arrow = fake ball throwing but drawing a line between each point, a ball is thrown but not drawn but instead a line is drawn between where it is and where it last was to create a line showing where the ball will go

If the screen changes then spawn balls that are drawn as characters at the top of the screen in a random area, set gravity to true and set x velocity to a random number.

The balls trajectory is based on gravity and the x velocity. Gravity makes the ball go down and the x velocity is increased or decreased until is 0 so the ball is not moving left or right

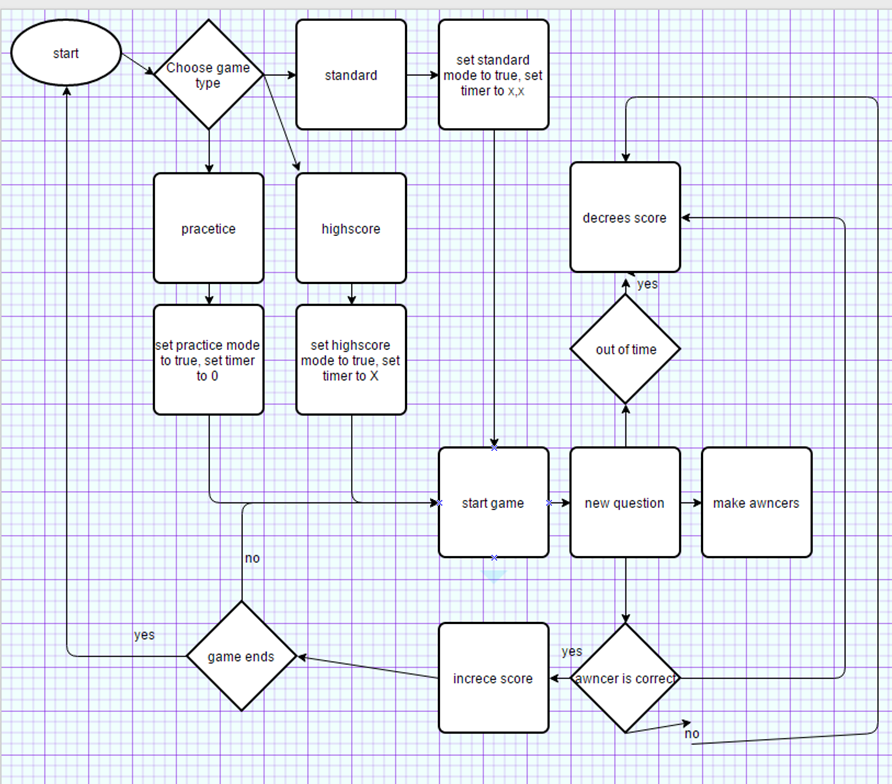
The word converter converts a sting in to letters and the letters into a sprite of the letter and places it next to the previous letter by incrementing the x.

Velocity equation: *v = u + at*

Bounce equation: *I = -I*

Word converter example: Letter = H, get sprite called H, draw sprite next to last sprite.

# Flow chart:



# Potential Tests

The system generates random values; to test I need to check a range of plays to see that it is working. My test plan will require the tester to log the outcome.

|  |  |
| --- | --- |
| test | explanation |
| Playing the game | To check if the game runs correctly and how I expected it to run |
| Answering a question | To check if you can answer a question correctly. This is checking the code not your skill |
| Random questions with answers | Make the game generate new questions with answers when needed |
| Getting a score | To check if your score increases/decreases with a right/wrong answer by seeing if the score changes correctly when answering a question right or wrong |
| Adding questions | Allow the user to get questions that require addition |
| Subtraction questions | Allow questions about subtraction for the user to answer |
| Yellow background | To test if the background was yellow by looking at it and seeing if it looks yellow |
| Ball bouncing off of wall | Make the ball bounce off of a wall when it hits a wall |
| letters being drawn correctly | To check if the string is correctly being converted into sprites and displayed |
| Randomly generate questions | To create new questions using random numbers for the user to awncer |

Testing:

**Questions asked:**









|  |  |  |  |
| --- | --- | --- | --- |
| test | explanation | Pass/fail | Screenshot |
| Playing the game | To check if the game runs correctly and how I expected it to run | Pass |  |
| Answering a question | To check if you can answer a question correctly. This is checking the code not your skill | Pass |  |
| Random questions with answers | Make the game generate new questions with answers when needed | Pass |  |
| Getting a score | To check if your score increases/decreases with a right/wrong answer by seeing if the score changes correctly when answering a question right or wrong | Pass |  |
| Adding questions | Allow the user to get questions that require addition | Pass |  |
| Subtraction questions | Allow questions about subtraction for the user to answer | Pass |  |
| Yellow background | To test if the background was yellow by looking at it and seeing if it looks yellow | Fail |  |
| Ball bouncing off of wall | Make the ball bounce off of a wall when it hits a wall | Fail |  |
| letters being drawn correctly | To check if the string is correctly being converted into sprites and displayed | Fail |  |
| Randomly generate questions | To create new questions using random numbers for the user to awncer | pass | Test 1    Test 2 |
| Spelling errors | To test if words are spelt wrong | fail |  |

The yellow background failed because they didn’t set it to yellow, to fix it they will set the background to yellow. To change the colour, they need to set where it sets the colour to orange to yellow so it sets the colour to yellow instead of orange. To change it I changed the line console.backgroundcolour = colour.yellow;

To console.backgroundcolour = colour.orange;

The ball didn’t bounce off of the wall because they did the maths wrong so it doubled its momentum instead of reversing it. To fix this they change the values so it reversed the momentum instead of doubling it.

The drawing of the word didn’t reset the pointer where it looked for the letter, didn’t reset the letter so it was looking a letter after a previous letter it would not find it, to fix this they made it reset the pointer after it found a letter.

They will re-test them all to make sure they work.

I know the tests were performed properly because I saw that the tests were being performed correctly and the answers are correct.

|  |  |  |  |
| --- | --- | --- | --- |
| Re-test | explanation | pass/fail | Screenshots |
| Yellow background | To test if the background is yellow by looking at it and seeing if it looks yellow | Pass |  |
| Spelling errors | To test If words are spelt wrong | pass |  |
| Ball bouncing off of a wall | Make the ball bounce so if it hits a wall then it looks like it bounces | Pass |  |
| letters not being drawn correctly | To check if the string is correctly being converted into sprites and displayed | Pass |  |

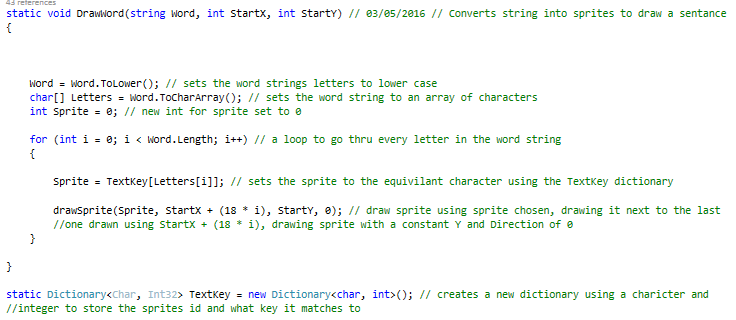
# Technical tests

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Reason | Pass / fail | What happened |
| Run on windows | To check if the program can properly run on windows | Pass | The program can run on windows |
| Run in full screen | To check if the program can run in full screen mode | pass | The program worked in full screens |
| More than 30 frames per second | To see if the program runs smoothly | Pass | The program ran at around 60 frames per second |
| Run in windowed mode | To check if the program can run in windows mode | pass | The program ran in windowed mode |

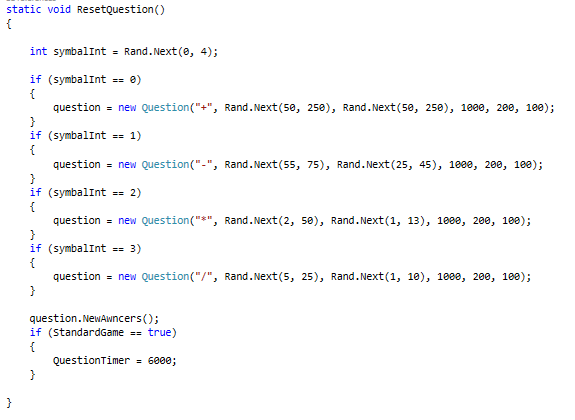
# key elements of the game



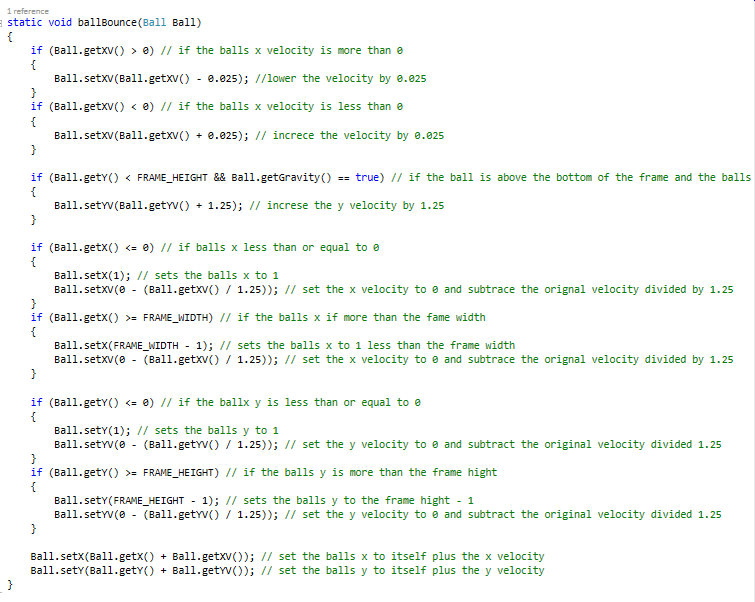
Converts the sprite sheet into separate parts so the sprites can be used in the program



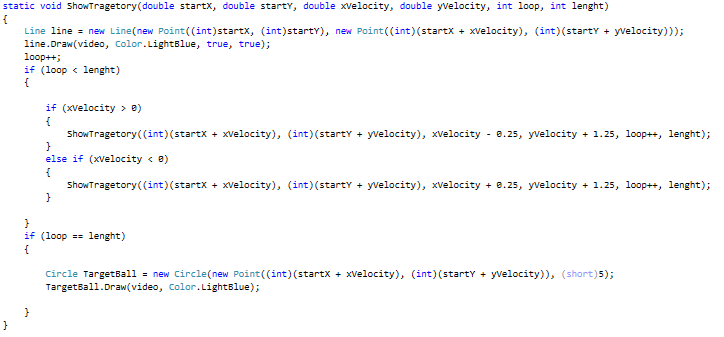
Converts a string into sprites and draws them at the location given



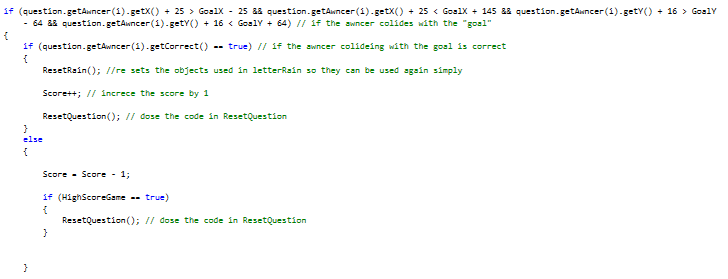
Randomly creates a new question with answers when called



Checks if the ball is touching things it should bounce of and changes its velocity’s to make it look like it bounces



Draws a line to show where the ball will go when pinged by mimicking the balls physics and drawing a line between points



Checks if the ball is colliding with the answer box and if it is correct and if it is inceses the score or decrees it depending on if it is correct

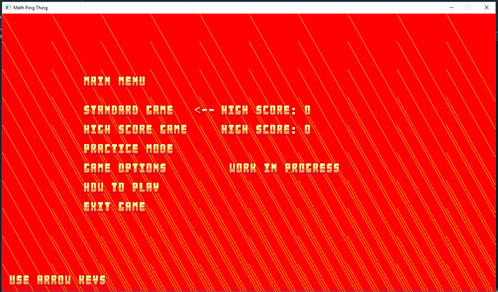
# Table of making the program:

|  |  |  |
| --- | --- | --- |
| week | What happened in week | Screenshot of program |
| 1 | Added sprites / string to sprites  (sprites being drawn to test what it would look like) |  |
| 2 | Added physics |  |
| 3 | Made objects throw able |  |
| 4 | Added questions and changed colours |  |
| 5 | Added answer |  |
| 6 | Added answer goal |  |
| 7 | Added score |  |
| 8 | Added menu and game modes |  |
| 9 | Added how to play |  |
| 10 | High score list |  |

Making the ball bounce was the easy part of making the game because it is simple to do and I have done it before, also making the dictionary was easy, making the quick sort was hard because I have not made or done quicksort before making this program. I chose quick sort because it is complicated to make compared to other sorts and it is fast which is useful in my program because it is fast.

# Colours:

I chose the final colours because they worked well. I think it may be viewed differently because I am colour blind but I don’t know because I cannot see non colour blind colours to compare. People say the orange and different orange colour scheme work well and better than blue. I think the orange looks good compared to the blue which I don’t think looked better because the blue made it look strange. I changed the colour to red to see if this would work better than the orange below.



The red background looks brighter than when the background was orange. I do not think the game works in red because It is brighter and more distracting compared to the orange colour. I gathered feedback to see if others agreed with my opinion.

|  |  |
| --- | --- |
| Interview with Matt | Me: how does the colour look in orange?  Matt: better than before. Better to read, less distracting/distorting on the text  Me: ok.  Me: can you read the text on the orange?  Matt: yes |
| Interview with Richard | Me: do you prefer orange or blue background?  Richard: it looks better in orange than the blue cause orange matches better.  Me: ok |
| Interview with Charlie | Me: Pick one. blue or orange?  Charlie: orange.  Me: Why?  Charlie: Blue is less appealing and harder to read because of the colour contrast |

# Client review:

Idea: A method to be used to help pupils practise their math skills. This would be based around multiple choices questions and have some interactive and fun method for selecting the answers.

## Why do you want it?

I want it so that pupils being taught maths will essentially be tricked into learning their math skills. Maths lessons are difficult for a large number of pupils, so they tend to avoid doing anything related to them. Hiding maths skills behind a fun game will help them learn the basic skills while bypassing their reluctance to engage with mathematics.

## What do you want?

A nice fun game that has randomly generated multiple choice maths questions based on multiplication, addition, subtraction and division.

## Question and Answer

### What do you think of the programme?

Pretty. I really like it. I think it is quite impressive. It’s a nice way of letting students to practise their abilities. I like the style and think it is very nice the controls are nice and simple. I really like the practise mode it very easily shows what they should be doing. I really like this. I think there are people that can already benefit from it as it currently stands. I don’t know how portable it is to different levels, but if it portable it could be easily sold.

### Does the programme work?

Yes, it does. I didn’t notice any bugs, but the only qwerk was after you got something wrong, it wasn’t as quick to allow you to take the next one.

### Is the programme clear to use?

The practise mode is fantastic.

### Is the programme fun?

Yes. I think if it could be shared with friends and had a scoring system then people could quite easily get involved in it quickly. I think it is something they can enjoy a lot and

### Does the physics work?

Yes, it does

### Are the colours ok

I like the red and yellow. It’s quite a nice contrast for what it is. I thought it wasn’t going to be as good as it is because of the colours, but now I think they work really well.

### Rate out of ten

If it is on a scale of this type of game, I would give it an 8 or 9 out of ten. As it is well put together, looks good and is a nice development. Some others have more images though. Very nearly an outstanding game

### How can it be improved?

You could do a theme and resale it with different themes e.g. my little pony, baseball version, football version. And then you and have global leader boards to give players something to aim for.

How well the project went:

I believe the project went well because I completed the program and the client was pleased with it. I believe with a bit more time and no write up to do I could extend the project more. During the programming, I came across a few bugs like letters not being drawn correctly and the ball getting stuck in walls but I managed to solve them relatively fast so I could continue with the programing and complete the project.

In the first criteria I believe I completed all the points because the player can answer addition questions and their score will increase it if is correct. It also took me less than 2 days to complete. In the second criteria I believe I completed it all because the game has working subtraction questions that perform correctly when answered. It took me less than 2 days to complete. Also in the third criteria I think I did not complete it because the questions won’t go up in difficulty when they are answered. I am considering adding it later because it could be a good addition. It may also take less than 1 week. I did not add it because the client did not specify the feature but I thought it was a good idea to challenge the players.

I think I completed the project because the game matches the specification given to me at the start, the game is playable in its current state and still meets the specification I have been given.

I think the smart objectives have been met because e.g. in the test table there is a test for the ball bouncing that failed originally but I corrected it to work properly. Also creating new random questions witch passed originally.

## Opinion:

This is one of the best examples of a learning game I have seen. I recently worked with a University student on their project producing educational games and theirs were not as advanced as this.

## Even better if?

Having variable difficulties or timers will be useful to have. This would allow pupils to start at a simple level and then increase the difficulty for bigger rewards.

Having themes would be great as it could be sold to a wider audience based on their interest in the Simpsons, my little pony or Pokémon

## How I would:

Variable difficulties: I think this is a good idea because it would add more to do in the game, I could do this by adding some sort of multiplier to increase the numbers used in the equations and a number that is used as a difficulty multiplier.

Variable timers: I would change the timer based on the difficulty so you have longer or easier questions and shorter on harder difficulties witch may make the game more fun.

Themes: I would have special sprite packs that replace the default sprites to make the game look like the selected theme without effecting the gameplay.

# Possible future extensions:

In the future, I would like to add different levels of difficulty, a system to save and compare scores with different players, a Day le challenge mode, rewards / medals for completing goals and possibly more game modes to play. I would add a better way for the player to know when they have gotten an answer correct or not correct. If they miss it may make the screen flash a different colour and explode if an answer is correct.

Ball.getxv() is the balls x velocity

Ball.getyv() is the balls y velocity

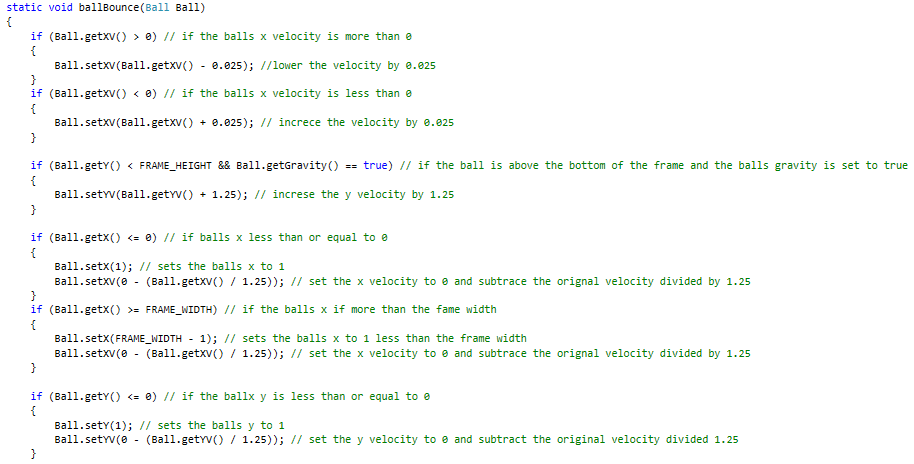
Frame\_width is the width of the window

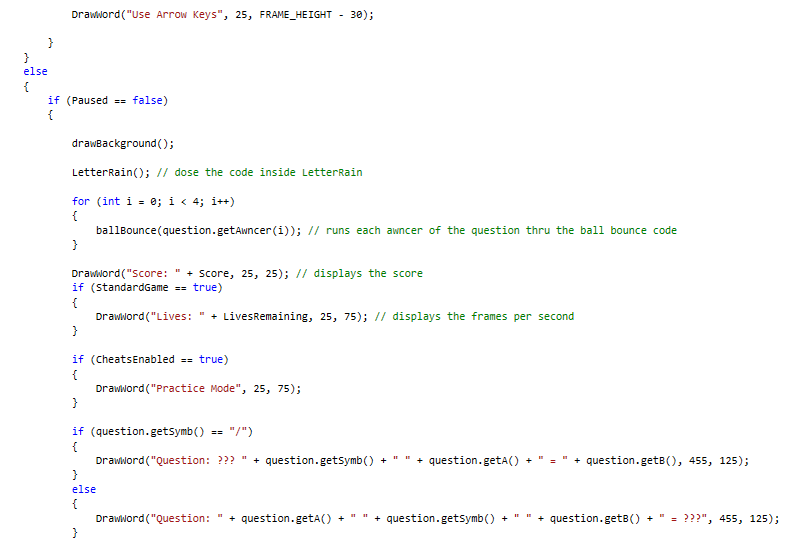
Frame\_height is the height of the window

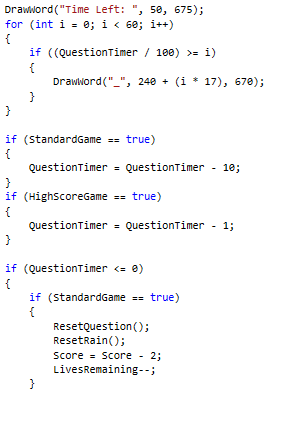
Ball.getx() is the balls x

Ball.gety() is the balls y

# Code relating to prototype



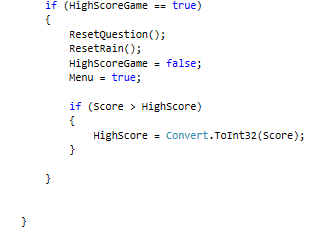


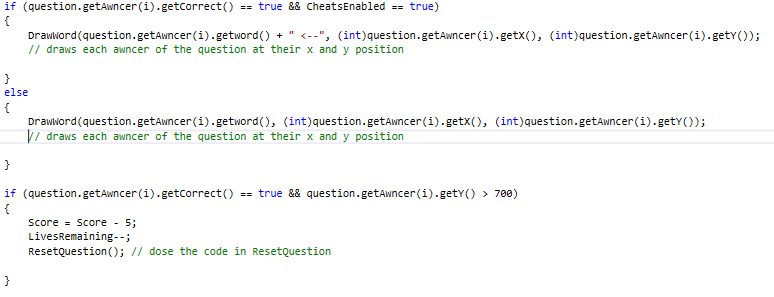


Question timer is how long the question has left to be answered

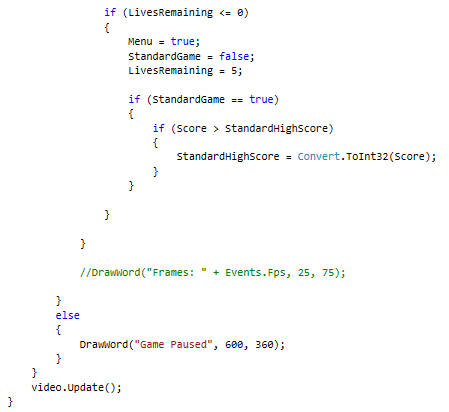
Standard game / highscoregame is the type of game being played

Score is the players score

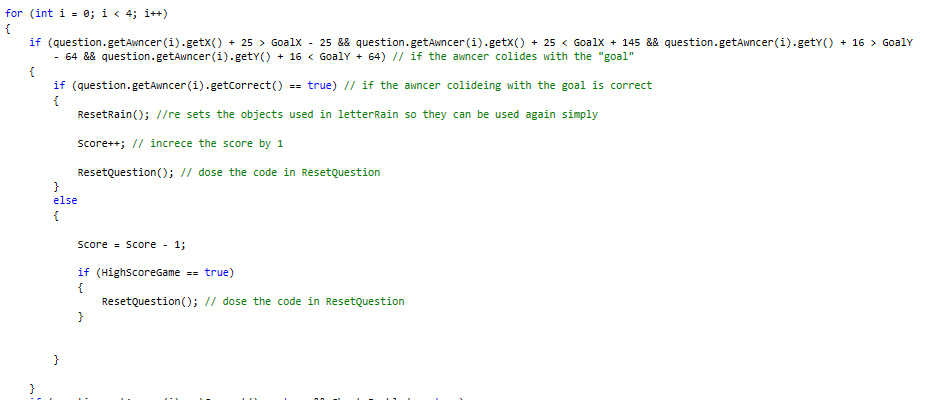


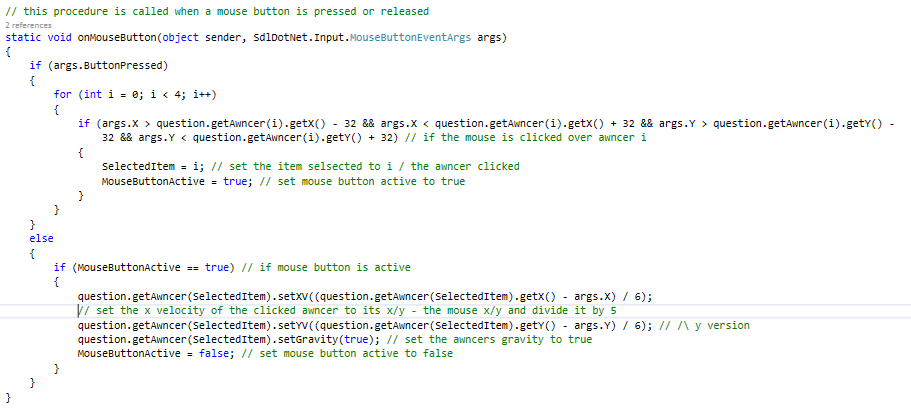


Livesremainig is how many lives the player has left



Menu is whether or not the player is in the menu



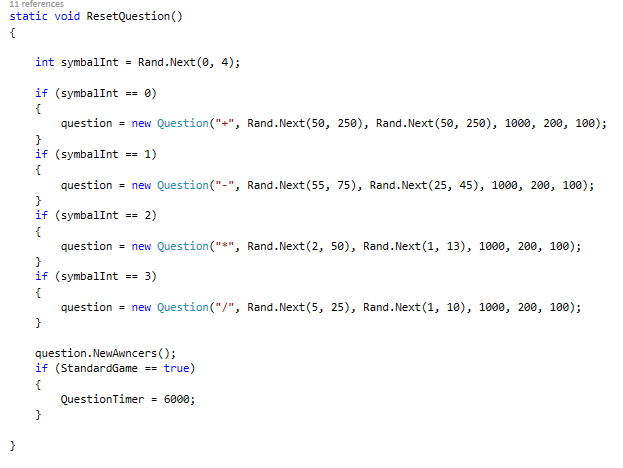


Mousebuttonactive is if the mouse button is being held down

Args.buttonpressed is whether or not the mouse button is being pressed

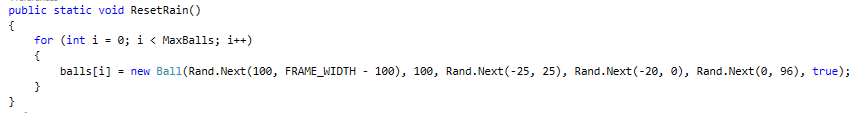
# Subroutines and magic number constants:

Subroutine:

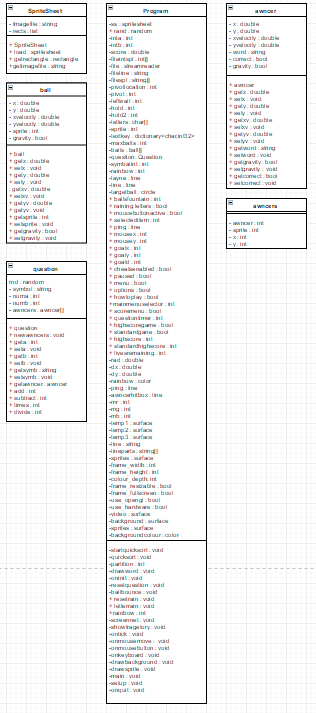


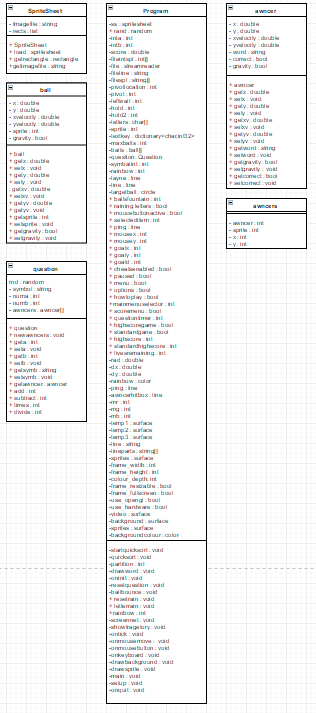
Magic number constant:



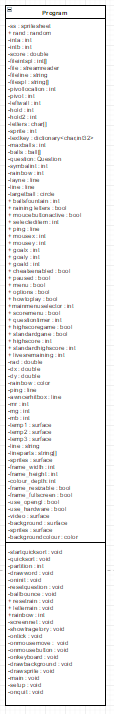
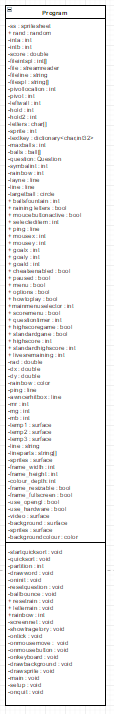


# Uml class diagram:

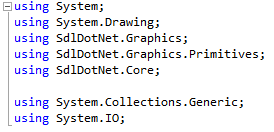


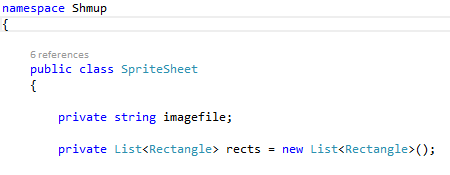


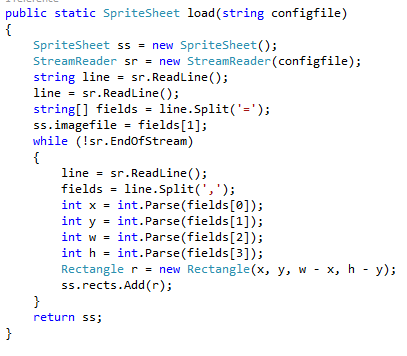
# Program code:



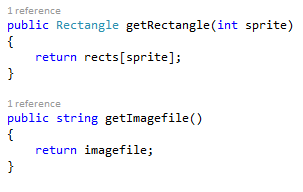
Program.cs

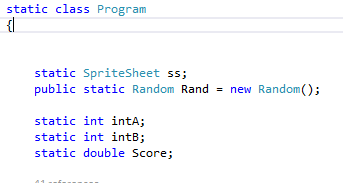


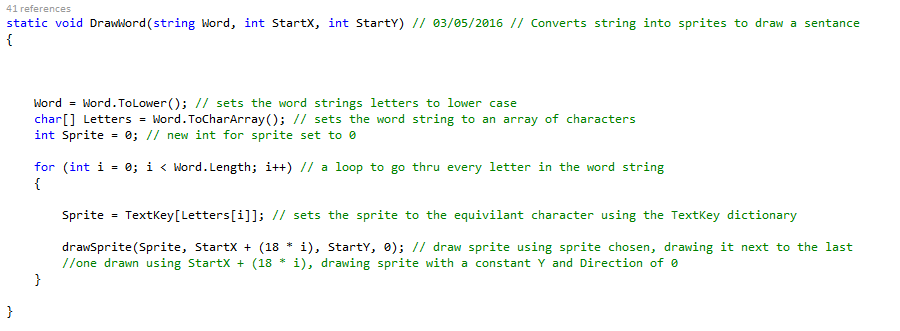




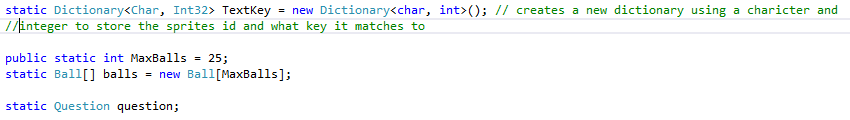
This code loads the sprite sheet into the game and splits it into sprites

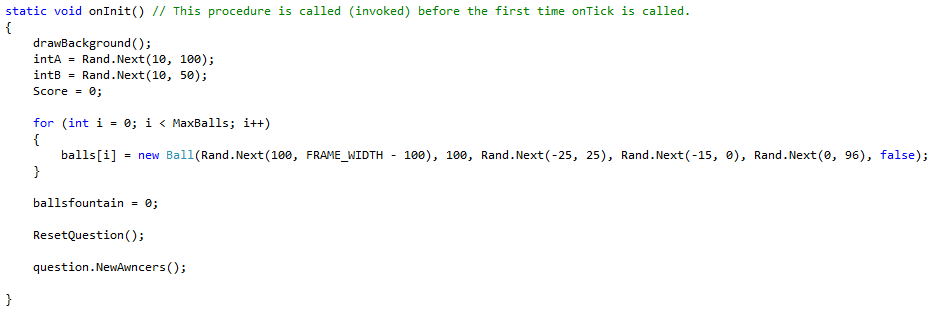




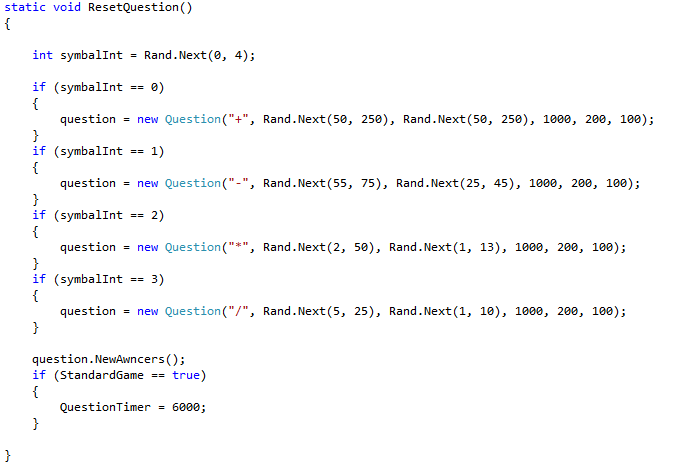


This code convert strings into sprites

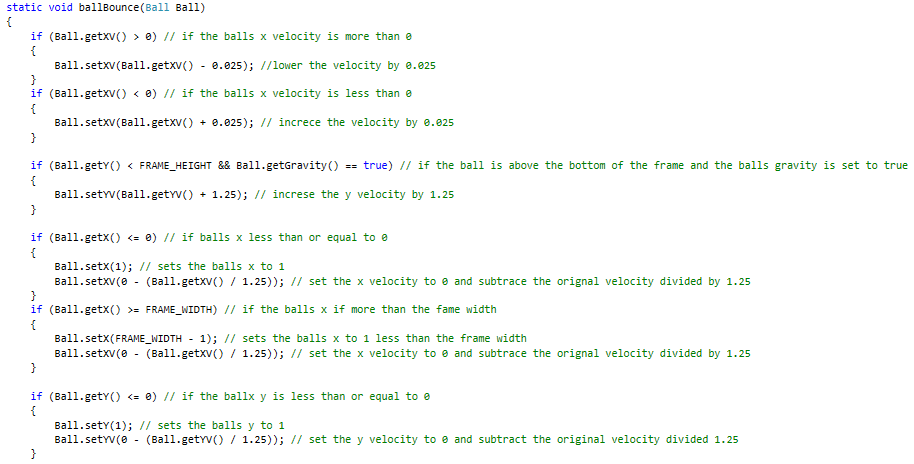




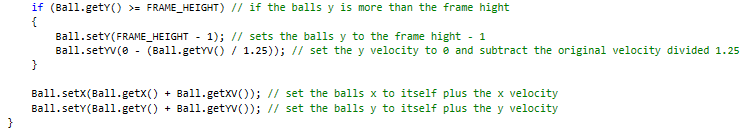
This code initialises the programs things that need to be initialised for the program to work in this stage

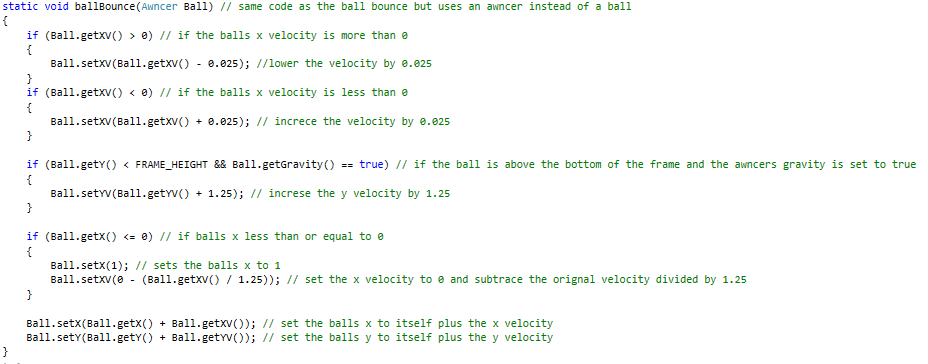


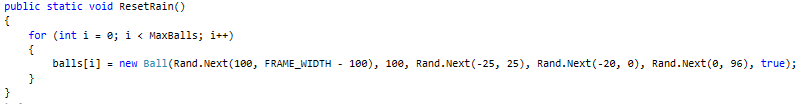
This code resets the question and answers

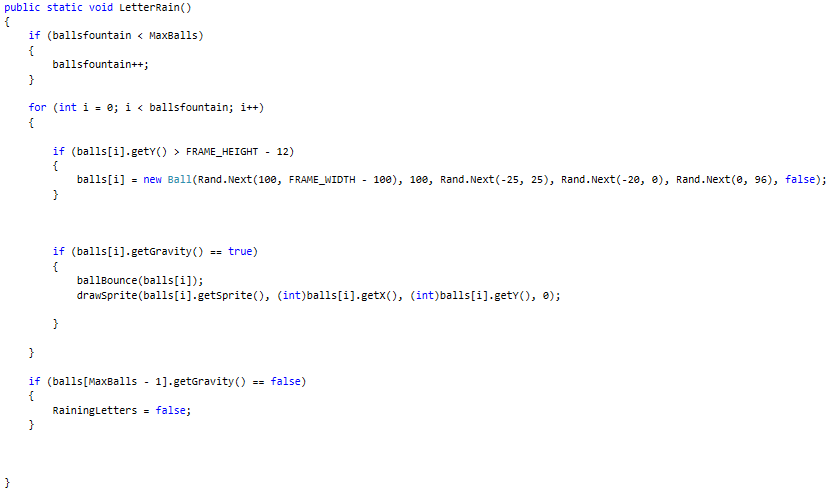


This code makes the ball bounce by changing its velocity’s

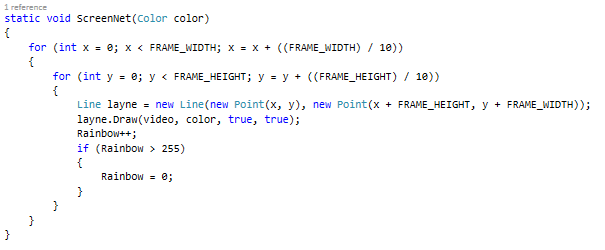




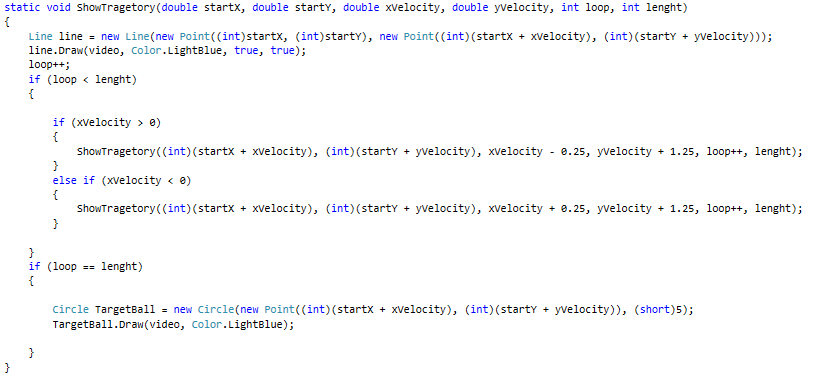




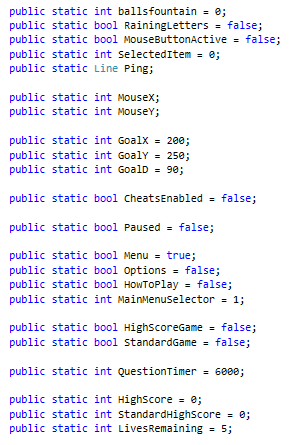
This code makes letters drop from a position



This code draws the lines on the background

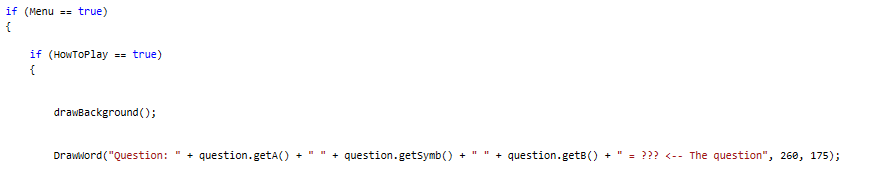


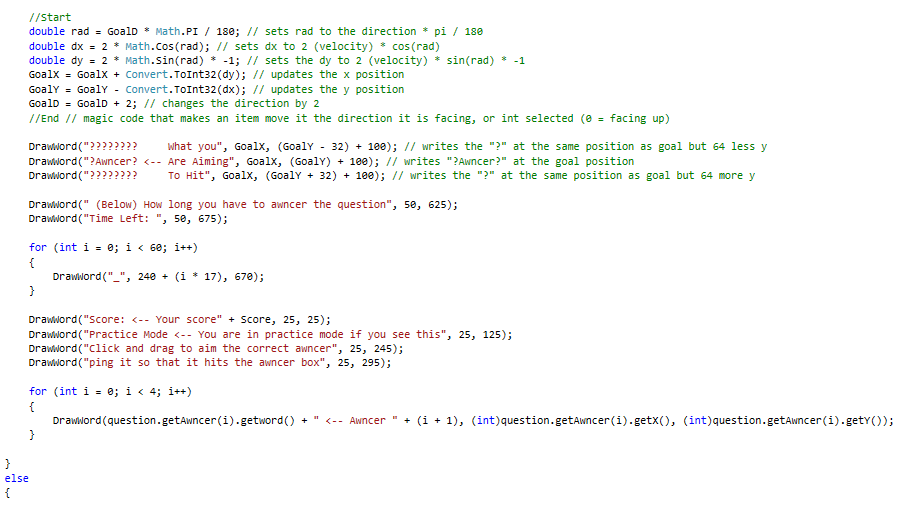
This code draws the line for where the ball will go

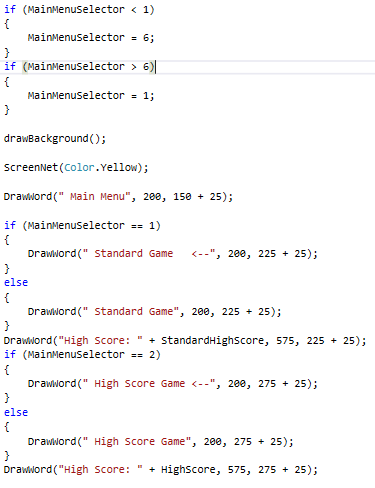


These are variables

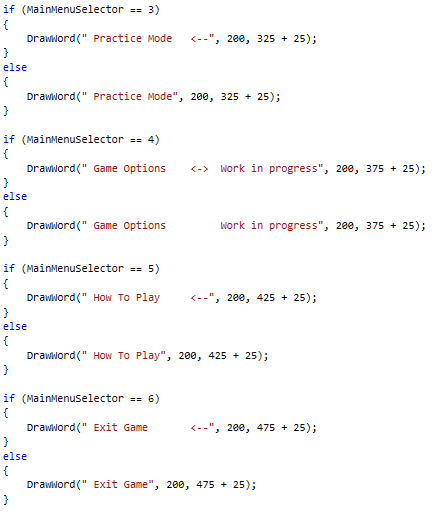




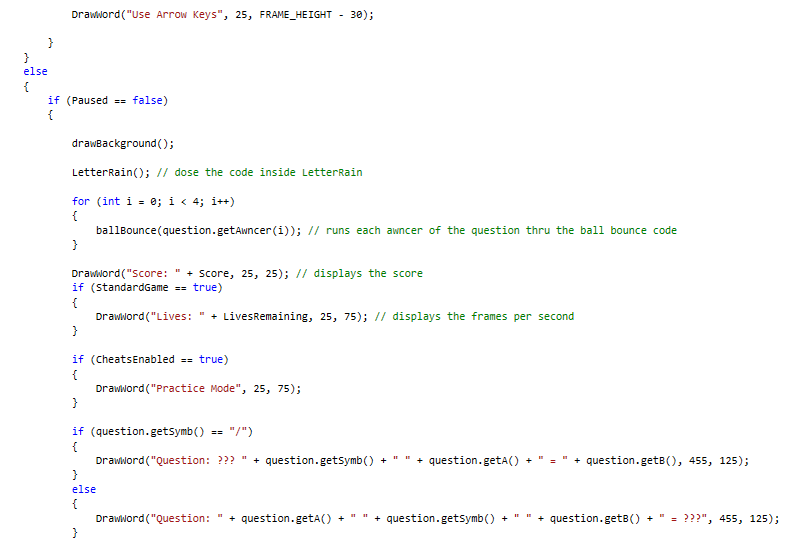




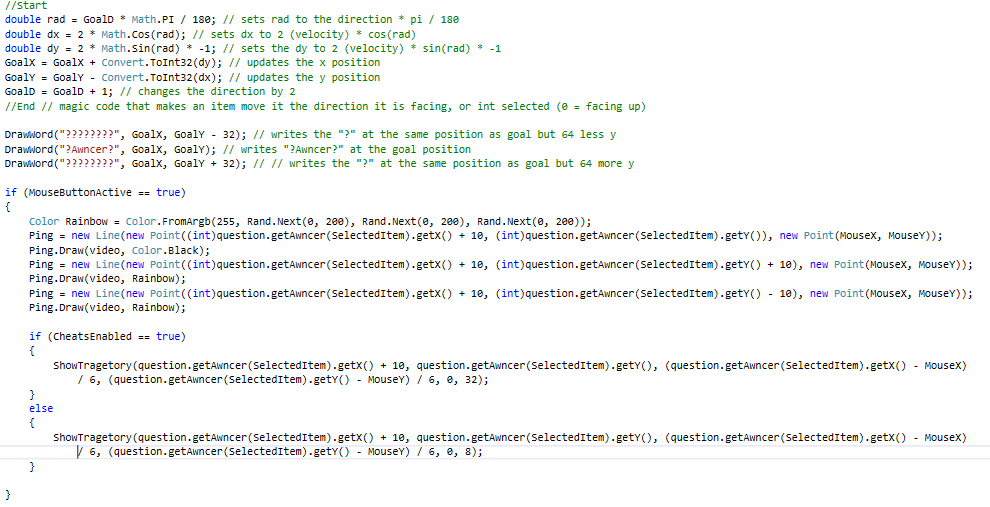
This code makes it so you can not go over the main menu button limit

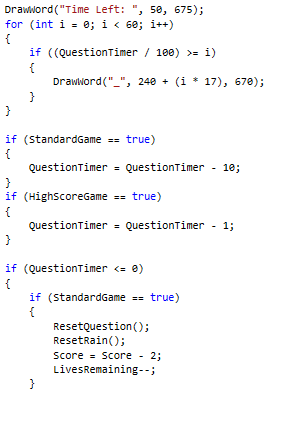


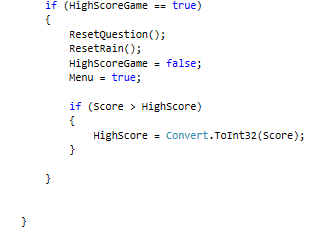
This code draws the buttons

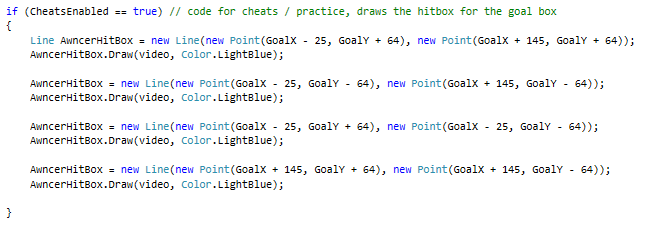


This code makes the program run if the program is not paused

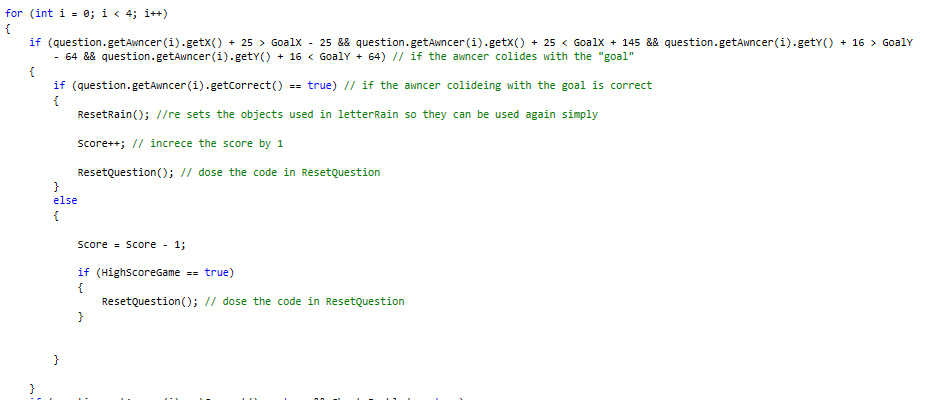


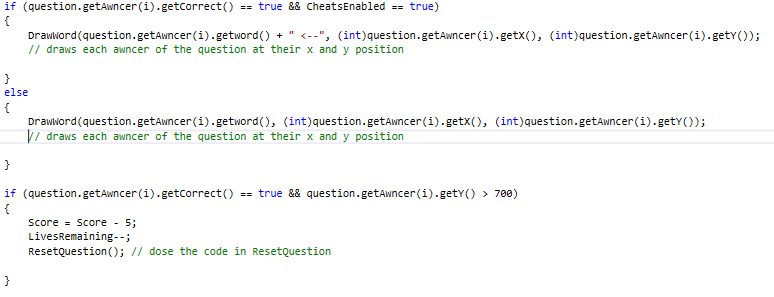


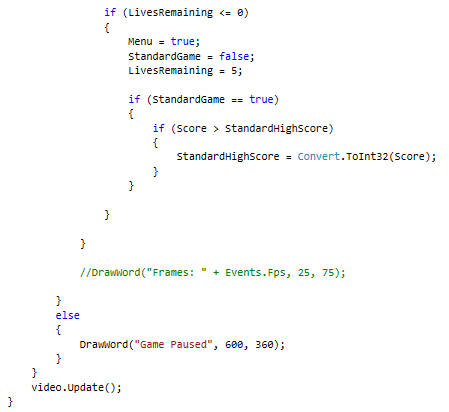


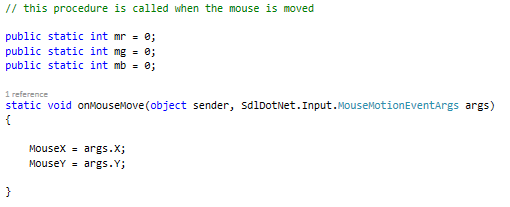


This code draws where the ball will go and the correct Awncer if cheats are enabled

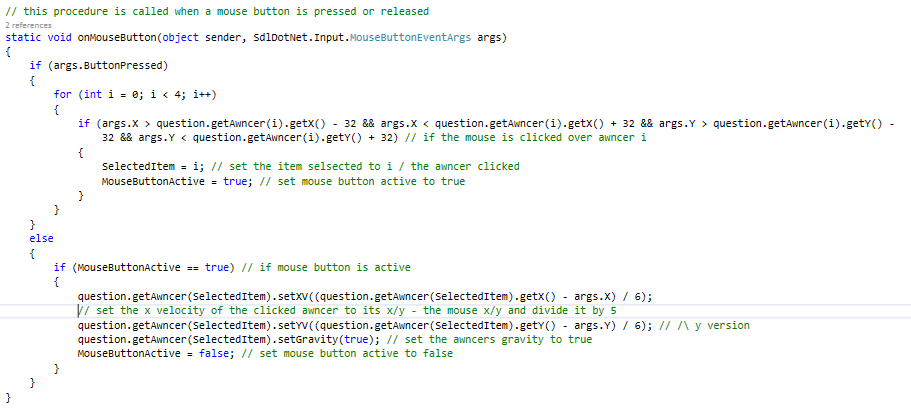




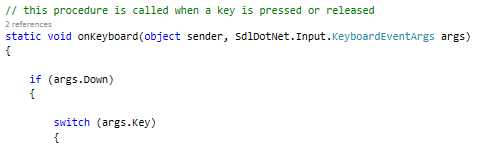




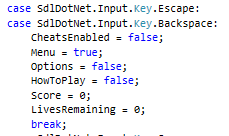
This code allows the mouse movement to be used

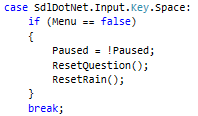


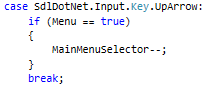
This code allows the mouse to click

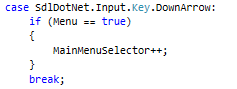


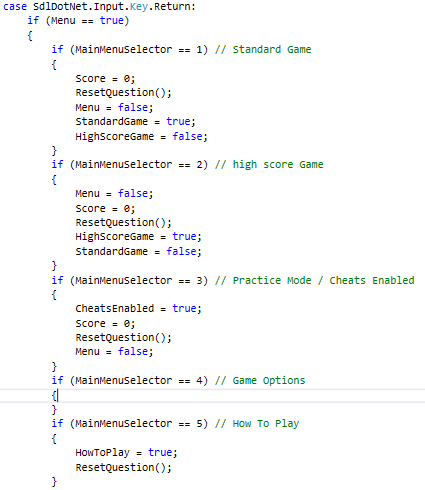
This code enables keyboard input

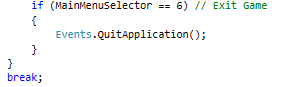


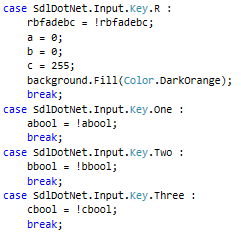


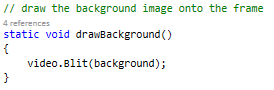




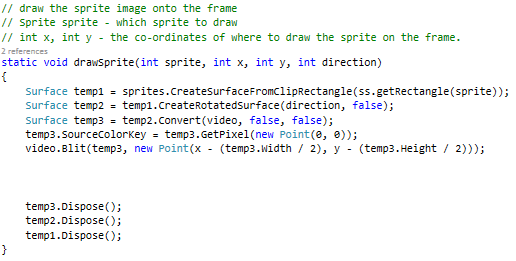




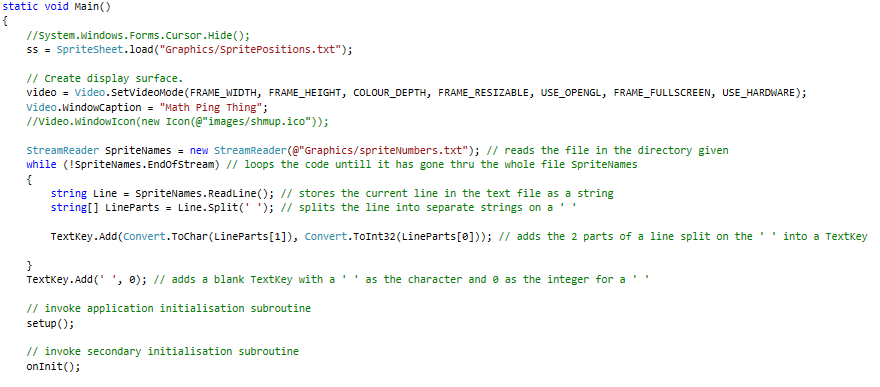




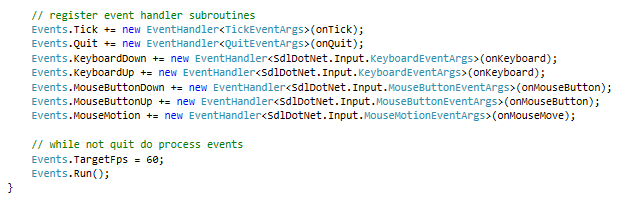
This code draws the background image

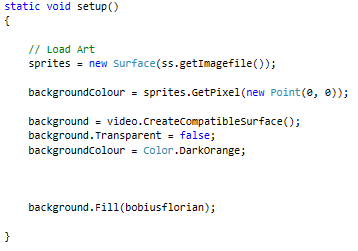


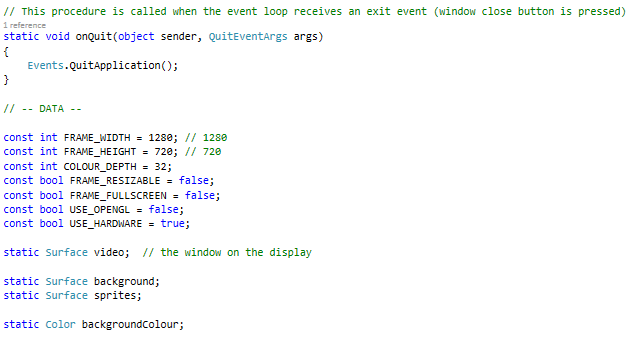
This code draws sprites



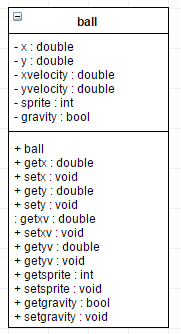
This code sets up the base of the program



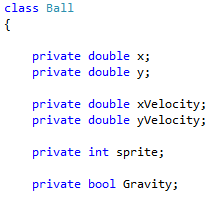




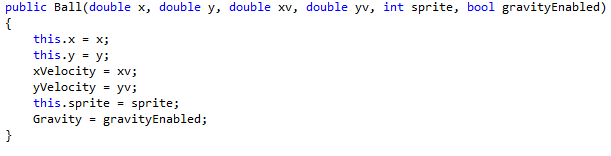
This code closes the program

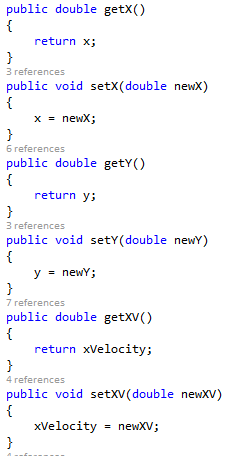


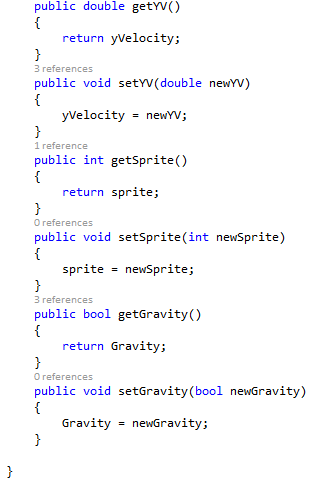
## Ball.cs

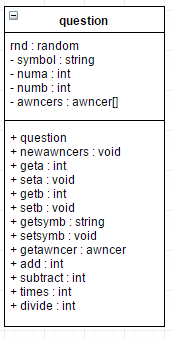


This class is the ball. It stores what is needed for the ball to work

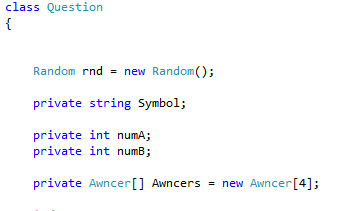




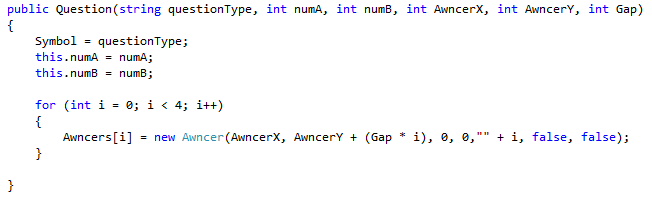


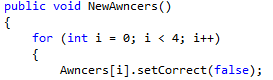


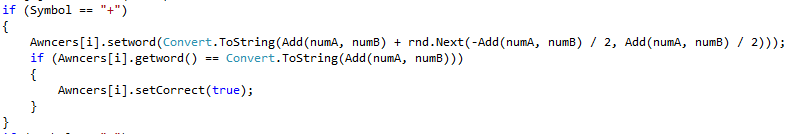
## Question.cs

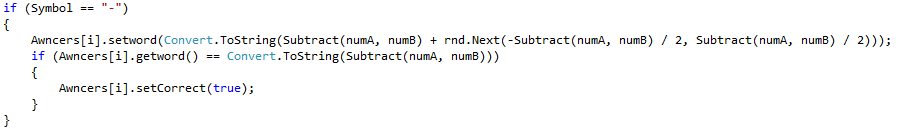


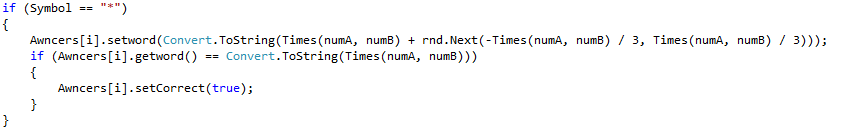
This class is for the question

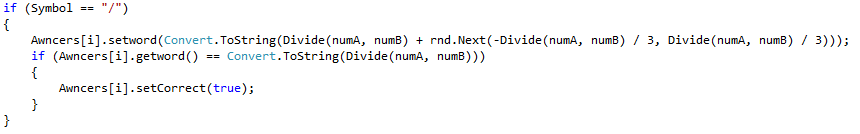


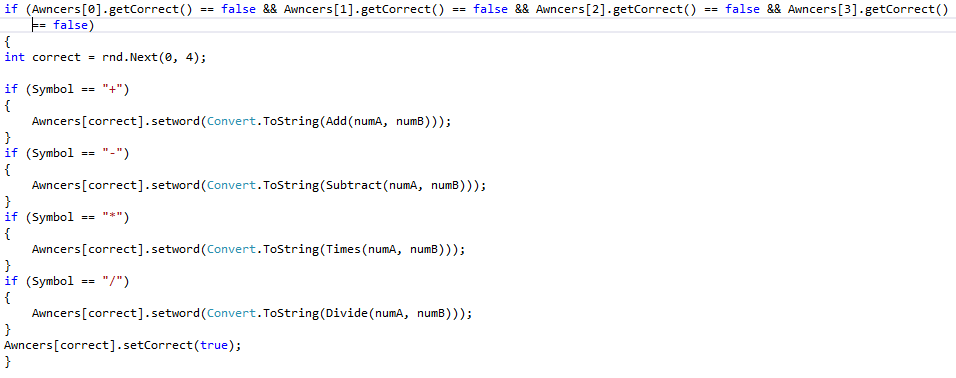


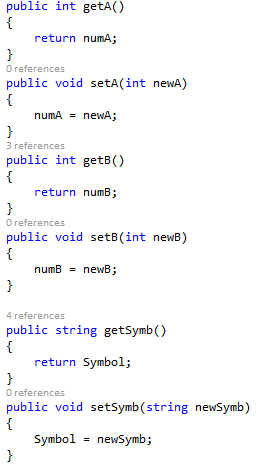


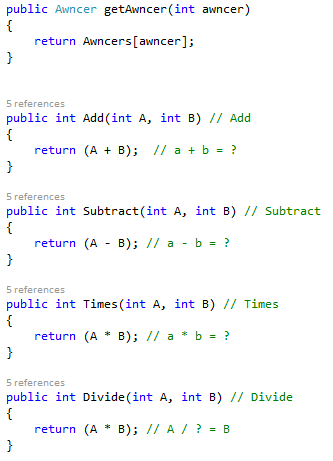


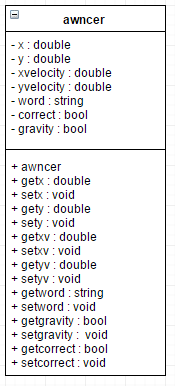




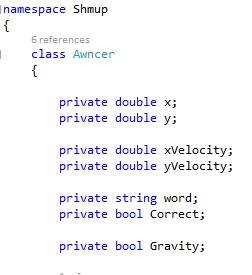




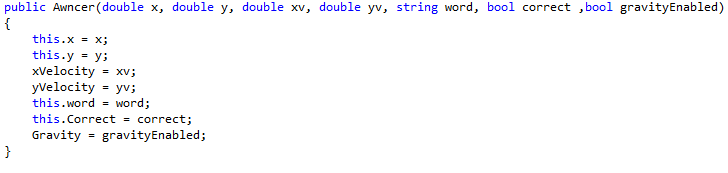


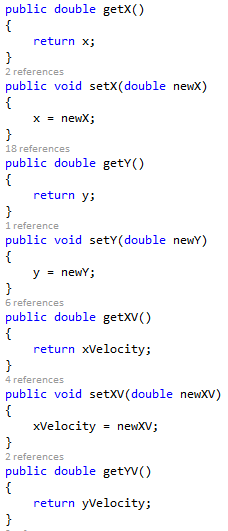


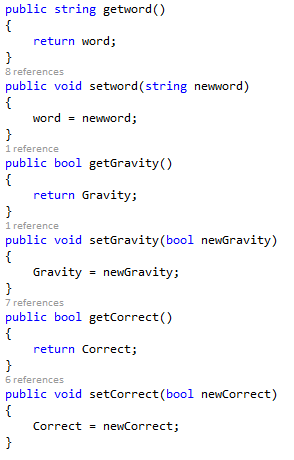
## Awncer.cs

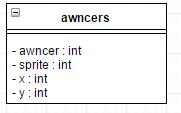


This class is for the Awncer class

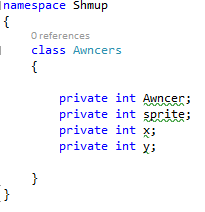






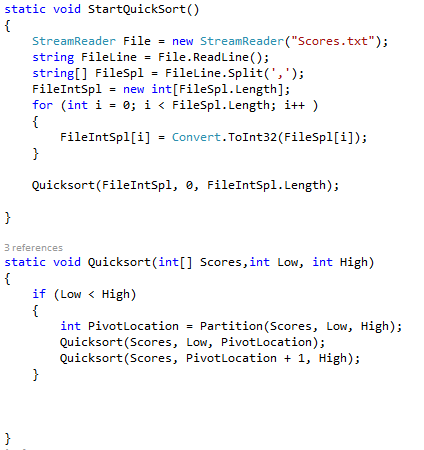


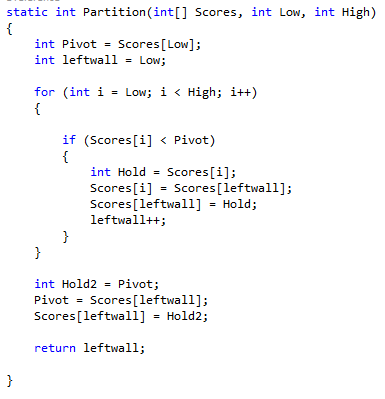
## Awncers.cs



This class is for the Awncer sprite to use as a base

# Quick sort





The list is sorted from lowest to highest so it is drawn backwards so it shows as highest to lowest.

