

Hussein Ahmed Hussein

student ID: A2035

KBU International College

School of Engineering and Computing

**Computer Science Programing (SOFT 10101)**

**Survival of the Solar System  
Game Documentation**

By

Hussein Ahmed Hussein

Module Leader: Ms. Anurehka

Table of Contents

[I. UML Diagram: 2](#_Toc427926687)

[II. Implementation: 3](#_Toc427926688)

[a) GUI, Guidance & Result: 3](#_Toc427926689)

[b) Functions: 6](#_Toc427926690)

[1. Elements () 6](#_Toc427926691)

[2. Fire () 6](#_Toc427926692)

[3. Jet () 6](#_Toc427926693)

[4. Rocks () 6](#_Toc427926694)

[5. Star () 6](#_Toc427926695)

[6. SSS () 6](#_Toc427926696)

[7. World () 6](#_Toc427926697)

[III. Black Box Testing: 7](#_Toc427926698)

[IV. References: 9](#_Toc427926699)

# UML Diagram:

SSS

Star(double initx, double inity);

move(); void

Element

Element();  
initx: double  
inity: double  
x: double  
y: double  
dx: double  
dy: double  
health: int

Element();

Star()  
initx: double  
inity: double

Star

Fire()  
initx: double  
inity: double  
hp: int

Element();

Fire

World

World::World()  
World::runGame()

Struct Point  
struct HighScore  
playerName: string  
TotallScore: int  
shootCounter: int  
shootDelay: int  
std::vector<Rocks> stones;  
spawnCounter: int  
starSpawnCounter: int  
std::vector<Star> stars;  
spawnDelay: int  
starSpawnDelay: int  
GTimer timer;  
timer.start();  
Score = 0: int  
screen = 1: int  
gameRun: bool  
GSound sund;  
OFfile: ofstream  
temname: string

Rocks();  
initx: double  
inity: double  
r: int  
radius: int

Rocks

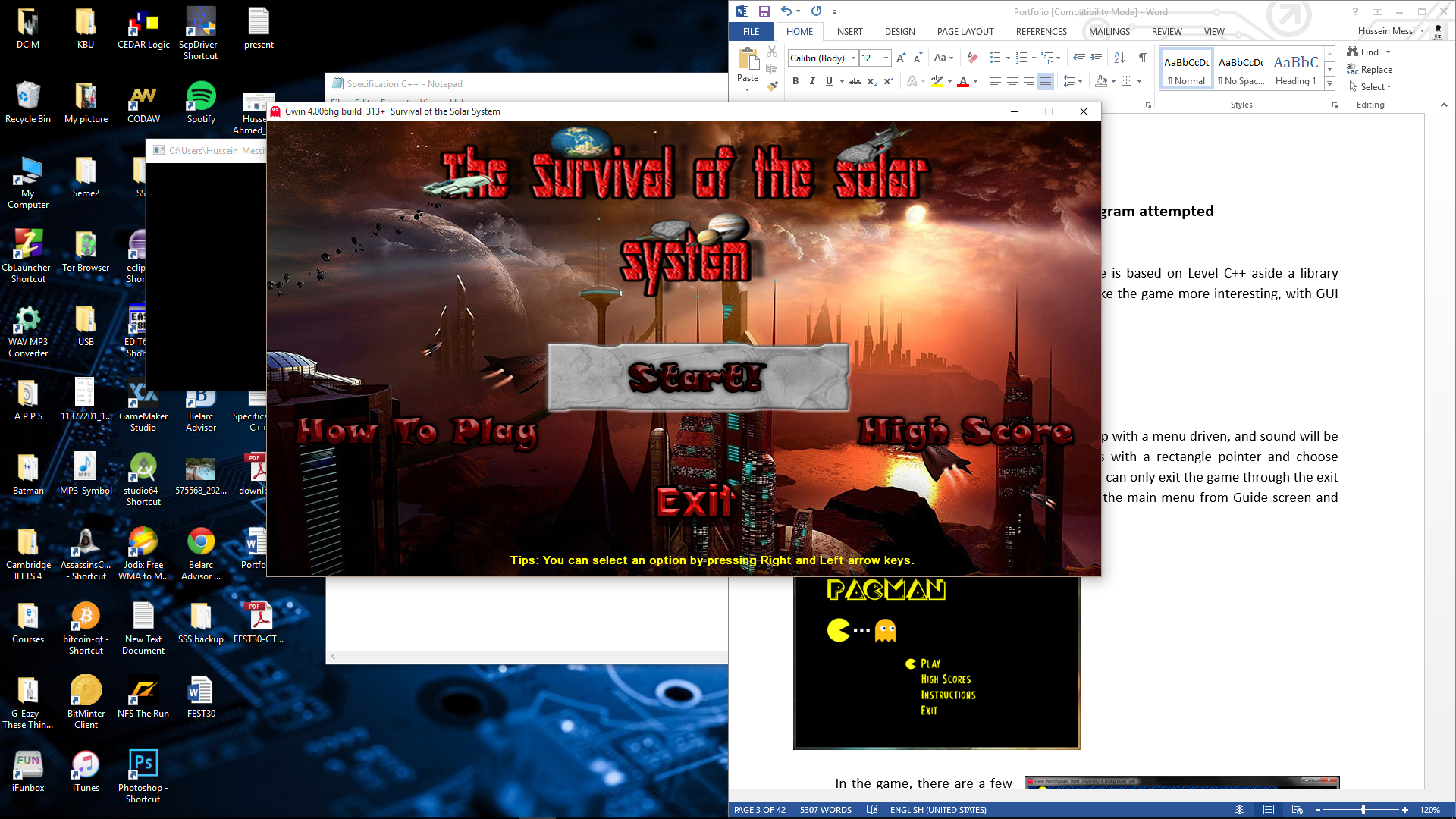
Element();

main(): int  
World world;  
world.runGame();

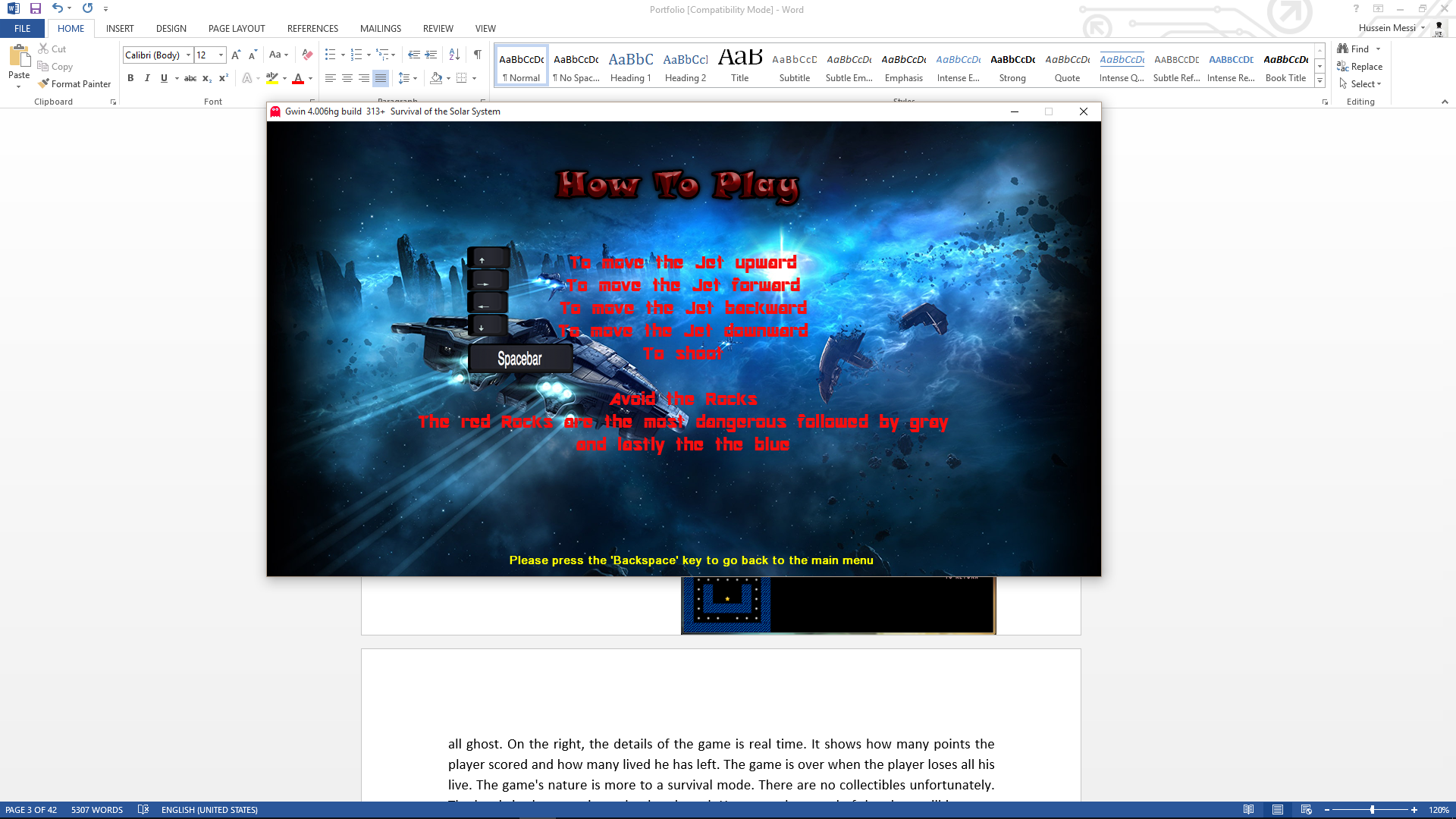
# Implementation:

## GUI, Guidance & Result:

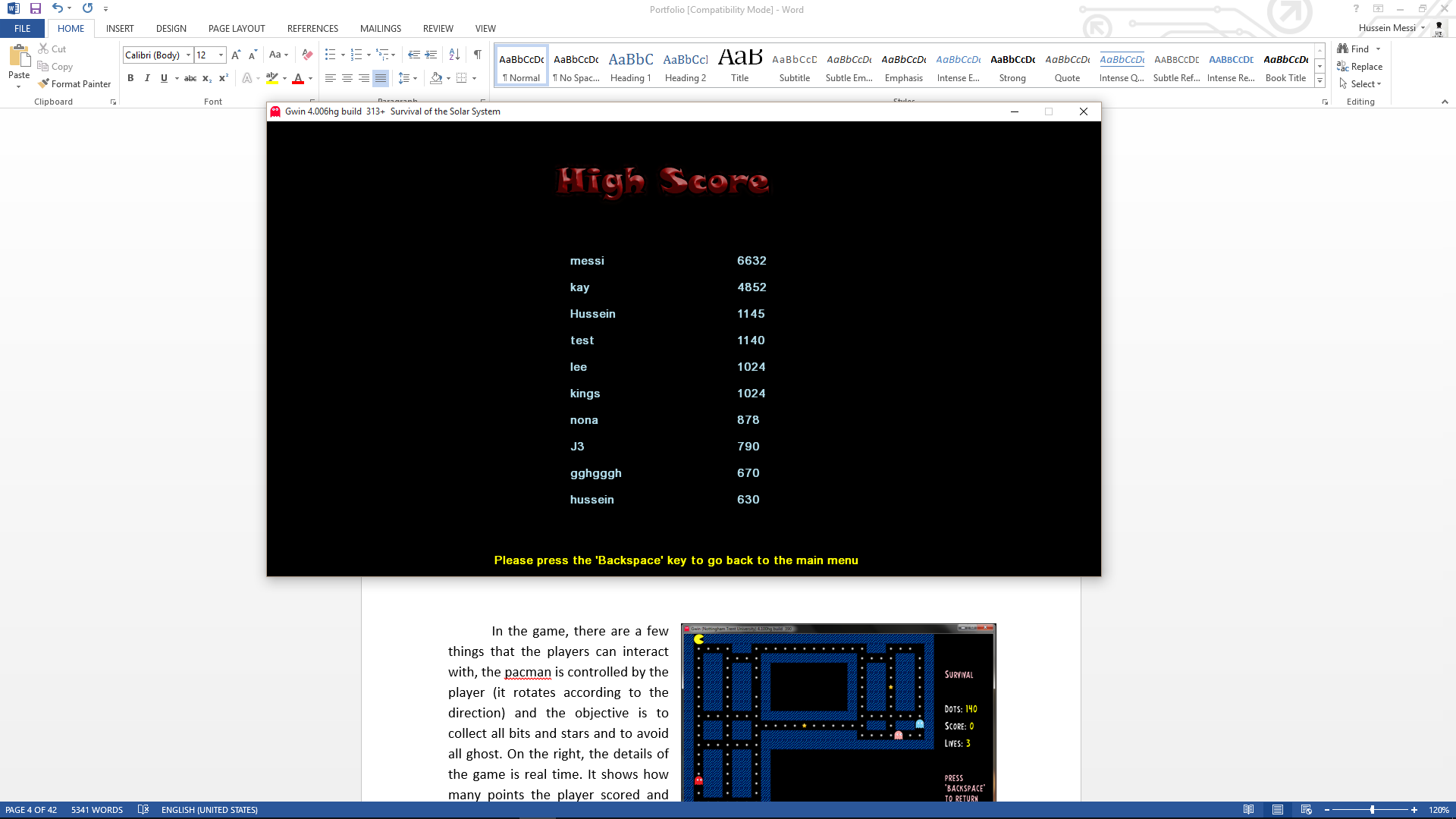
The program that is attempted for the “SSS” game is based on Level C++ aside a library called ‘GWin’ and some additional attributes to make the game more interesting, with GUI of (1100 x 600).



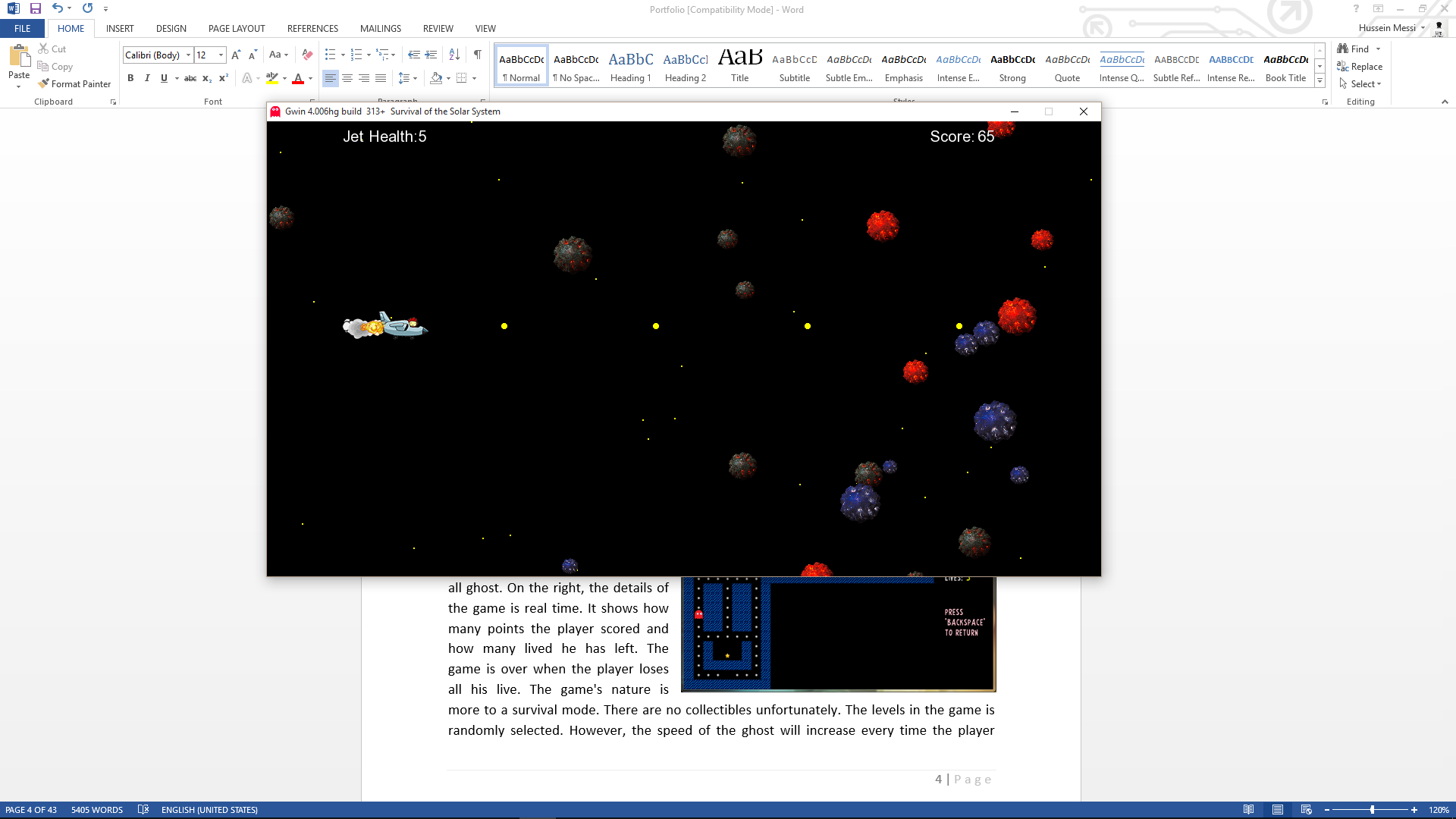
Once the Execution file is lunched a menu will pop up with a menu driven, and sound will be played. The players can navigate between options with a rectangle pointer and choose either (Start!, How To Play, High Score, or Exit). They can only exit the game through the exit option in the menu. As we as they can A return to the main menu from Guide screen and the score screen by pressing the backspace key, also while they are playing the game.



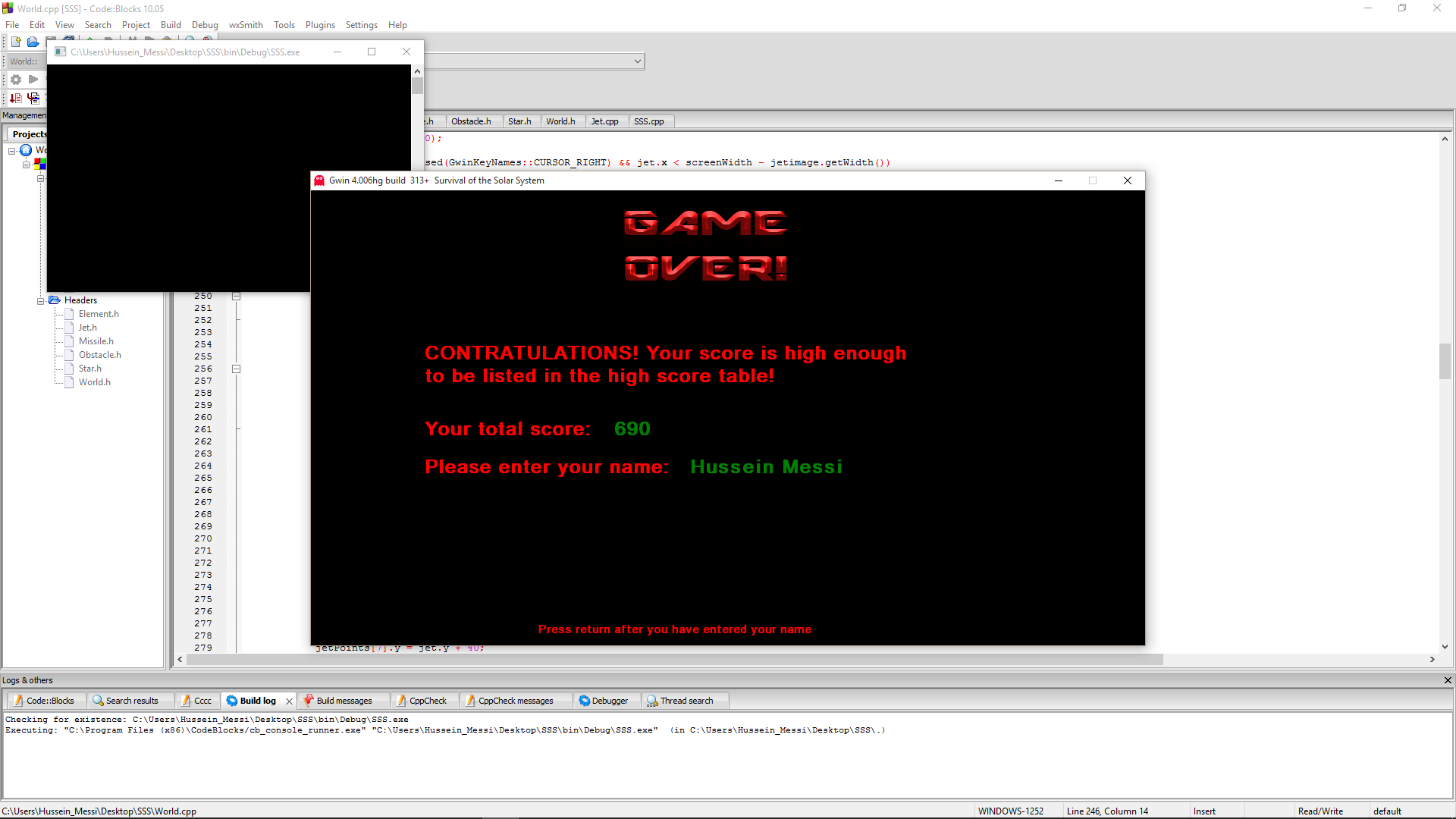
On the “How To Play” screen, the players will be able to read the instruction of how to play the game and by pressing the ‘Backspace’ key the will return to the main menu.



On the “High Score” screen, the player will be able to see they high score displayed with the name. The list will held only 10 names beside their scores. Once a new higher score than the existed scores is achieved, the lowest score will be remove and the new higher score will be registered to its position, depends on how high the score is.



In the game, there are a few things that the players can interact with, the Jet which will be controlled by the player. It move only in four directions (Up, Down, Left and Right) by pressing the arrow keys, and by pressing the ‘Space’ key the jet start shooting. Also, the player will interact with the rocks where they are moving automatically and randomly form the right side of the screen. On the top side of the screen will be displayed the “Jet Health” and the “Score”, where they are real time. As many rocks the player hits the score will increase and lose health when collide with rocks. Therefore the objective of the game is to destroy as many rocks as possible and also avoid the rocks to not to lose health. The game is over when the player loses all the health.



“Game Over” screen will be displayed when the players lose all the jet’s health and they have a score that is high enough to be listed on the score list. Therefore the score that they achieved will be shown, and also they will be asked to type their names and then press “Enter”.

## Functions:

The “SSS” game application contains basic functions just like any other games do. For instance reading inputs from the users as well as writing output to a file, and also other many algorithm functions which are used within the game system as shown below:

### Elements ()

The elements function is where it is created with its variables  
health as integer where it is used as a health for the jet as well as for the Rocks   
x and y are the points declared as double  
dx and dy as the direction and positions in the screen.

### Fire ()

In fire function I declared the variables dx as integer which equals to 20, and that is to set the fire which are coming out from the jet to move in a specific speed. And dy as integer equals to 0, also to make the fire move in one straight line which horizontally. And lastly is the ‘health’ which is inherited from element class, I set health equal to ‘hp’ so when the fire hit other objects like the rocks, it makes them lose their health.

### Jet ()

Jet function is where to assign the health to jet as 5, again the health is inherited from the element class. Another function is created to add the bullets with their direction and also the health for the jet.

### Rocks ()

once more inheriting the health and directions from elements class to assign them to the rocks plus but this time the health of the rocks is 4. And the direction of the rocks is assign randomly by this function   
dx = (rand()%3)-3;  
dy = (rand()%3)-1;

### Star ()

Star function is to make the stars move in specific speed and certain direction.

### SSS ()

SSS function is actually the main class where it is inheriting World() and world.runGame()

### World ()

The main function in “World” are the screen width and height as well the rungame menu, other than are all inherited functions and variables from other classes.

# Black Box Testing:

|  |  |  |
| --- | --- | --- |
| **Sample Input** | **Expected Outcome** | **Actual Outcome** |
| *- Main menu -* | | |
| Press :  CURSOR LEFT | While the sound is playing Rectangle pointer should move left from “Start!” Option to “How To Play” option then to “High Score” option and ends at “Exit” option However; if the pointer is at the “Exit” option, it should not move to the left anymore. | The rectangle pointer moves left and point to “How to play” option followed by “High Score” option and then “Exit” option. It does not move anymore to the left when it is at the “Exit” option”. |
| Press :  CURSOR RIGHT | When the pointer is at “Start!” option, it should not move to The Right. Unless it is at the “Exit” option, then it moves to the “High Score” option followed by “How to play option and finally “Start!” option. | The rectangle pointer does not move to right when it is at the “Start!” option. But it moves to the right when it is at the “Exit” option followed by “High Score” Option then the “How To Play” option and finally back to “Start!” option. |
| Press :  BACKSPACE | The BACKSPACE should only work when the player is at any of these screen (How to play, high score or while playing the game).  By pressing the BACKSPACE the player should be led to the main menu. | The BACKSPACE only works on the (How To Play screen, High Score scree and while the game is played) and when I press it the main menu appear. |
| *- while playing the Game -* | | |
| Press :  CURSOR UP | The Jet should moves upwards. | The Jet moves upwards. |
| Press :  CURSOR DOWN | The Jet should moves downwards. | The Jet moves downwards. |
| Press :  CURSOR LEFT | The Jet should moves backwards. | The Jet moves backwards. |
| Press :  CURSOR RIGHT | The Jet should moves forwards. | The Jet moves forwards. |
| Press :  BACKSPACE | It should return the player to the main menu. | Player is sent back to the main menu. |
| Press: SPACE KEY | The Jet should release fire | The fire is released when I press SPACE KEY |
| Release:  SPACE KEY | The Jet should stop firing | The fire stopes when releasing the SPACE KEY |
| When the fire touches the rocks | Red rocks should turn to gray.  Gray rocks should turn to blue.  Blue rocks should vanish.  The score should be increasing. | When I shot at the rocks. The red rocks will become gray, and the gray rocks turn blue and the blue rocks disappear.  As well as the score is increasing every time a bullet touches a rock |
| When there is collision between the Jet and the rocks | Colliding with red rocks: the health of the jet should decrease by 3. And the red rock turn to gray.  Colliding with gray rocks: the health of the jet should decrease by 2. And the gray rock turn to blue.  Colliding with blue rocks: the health of the jet should decrease by 1. And the blue rock should vanish. | When the jet touches the red rock the health of the jet decreases by 3. And the red rock turn to gray.  When the jet touches the gray rock, the health of the jet decreases by 2. And the gray rock turn to blue.  When the jet touches the blue rock, the health of the jet decreases by 1. And the blue rock vanish. |

# References:

[CPP CODE TIPS](https://cppcodetips.wordpress.com/) (23/Dec/2013), UML Class Diagram Explained With C++ samples (https://cppcodetips.wordpress.com/2013/12/23/uml-class-diagram-explained-with-c-samples/). [Accessed 17/8//2015].

TutorialsPoint, Black box Testing (<http://www.tutorialspoint.com/software_testing_dictionary/black_box_testing.htm>). [Accessed 18/8/2015].

Wikipedia (June 2012), Black-box testing (<https://en.wikipedia.org/wiki/Black-box_testing>). [Accessed 18/8/2015].