

Prathamesh Tarde

Snake

|  |  |
| --- | --- |
| Name | Page |
| Defining and Understanding the problem | 2 |
| problem | 2 |
| objectives | 2 |
| ipo chart | 2 |
| project management | 4 |
| gantt chart | 4 |
| log book | 5 |
| algorithms | 9 |
| psuedocode | 9 |
| systems flow chart | 13 |
| desk check | 14 |
| structured walkthrough | 17 |
| Building software solution | 18 |
| sys | 18 |
| Railroad/Ebnf diagrams | 20 |
| checking software solution | 21 |
| testing methods | 21 |
| User testing | 22 |
| Evaluating software solution | 23 |
| evaluation | 23 |
| peer testing | 23 |
| modifying software solution | 24 |
| modification 1 | 24 |
| modification 2 | 24 |
| Intrinsic Documentation | 24 |

**Table of Contents**

**Defining and Understanding**

**Problem:**

You have been employed by a local children’s day care centre to develop a program that will be implemented on their computers for students to use during free time or when they have been given permission for good behaviour. This software should be suitable for students within the age bracket of 3-5 years old. Remember suitable social and ethical issues should be employed when generating ideas. You should ensure that the program is appropriate for the capability of children of that age. The software should also contain relevant help that will be suitable for children and the adult supervisors.

**Objectives**

The objectives that the system will meet are:

* To create an error less game
* To create a game which is suitable for kids ranged between 3-5
* To create a game which will be which kids will enjoy playing within their free time
* To allow the kids to understand how to play the game
* Create a game which doesn’t take up time
* To create a user interface which is easy to use
* The program should be free of malware
* Take into consideration and oblige by social and ethical issues

**IPO Chart**

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| Key “a” | The system will recognise the key and move the snake | The snake moves to the left |
| Key “s” | The system will recognise the key and move the snake | The snake moves downwards |
| Key “d” | The system will recognise the key and move the snake | The snake moves to the right |
| Key “w” | The system will recognise the key and move the snake | The snake moves upwards |
| The snake collides with the playing field | The system analyses this as a collision/interaction and the game will finish | The game will stop and a message box will pop up displaying “You lose” and the player’s score |
| The snake collides with its body | The system analyses this is as a collision/interaction and the game will finish | The game will stop and a message box will pop up displaying “You lose” and the player’s score |
| The snake collides with a rock | The system analyses this as a collision/interaction and the game will finish | The game will stop and a message box will pop up displaying “You lose” and the players score |
| The snake intersects with an apple | The system analyses this as an interaction and the system will add a point to your score and another rock will be added onto the playing field | A point is added to your score and another rock will randomly pop up on the playing field |
| Start Button | The system will recognise the user input and open the game selection page | The game selection page will be shown |
| Controls Button | The system will recognise the user input and open the controls page | The control page will be displayed |
| Rules Button | The system will recognise the user input and open the rules button | The rules page will be displayed |
| Tutorial Button | The system will recognise the user input and open the tutorial button | The tutorial page will be displayed |
| Arrow Button | The system will recognise the user input and open the previous page | The previous page will be displayed |
| Levels Button | The system will recognise the user input and will open the level selection page | The level selection page will be displayed |
| Infinite Button | The system will recognise the user input and will open the infinite game | The infinite game mode will be opened |

**Project Management**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Gantt Chart |  | Estimated Time |  | Actual Time |  |
|  | Week 1 | Week 2 | Week 3 | Week 4 | Week 5 |
| Game |  |  |  |  |  |
|  |  |  |  |  |  |
| Algorithms |  |  |  |  |  |
| Pseudocode |  |  |  |  |  |
|  |  |  |  |  |  |
| Desk Checking |  |  |  |  |  |
|  |  |  |  |  |  |
| Structured Walkthrough |  |  |  |  |  |
|  |  |  |  |  |  |
| Defining and understanding |  |  |  |  |  |
| Objectives |  |  |  |  |  |
|  |  |  |  |  |  |
| IPO Chart |  |  |  |  |  |
|  |  |  |  |  |  |
| Project Management |  |  |  |  |  |
| Log Entry |  |  |  |  |  |
|  |  |  |  |  |  |
| Building Software Solution |  |  |  |  |  |
| Syntax Diagrams |  |  |  |  |  |
|  |  |  |  |  |  |
| EBNF Diagram |  |  |  |  |  |
|  |  |  |  |  |  |
| Railroad Diagram |  |  |  |  |  |
|  |  |  |  |  |  |
| Screen Designs |  |  |  |  |  |
|  |  |  |  |  |  |
| Checking Software Solution |  |  |  |  |  |
| Testing Process |  |  |  |  |  |
|  |  |  |  |  |  |
| User Testing |  |  |  |  |  |
|  |  |  |  |  |  |
| Evaluating Software Solution |  |  |  |  |  |
| Peer Testing |  |  |  |  |  |
|  |  |  |  |  |  |
| Modifying Software Solution |  |  |  |  |  |
| Modifications |  |  |  |  |  |
|  |  |  |  |  |  |

**Log Book**

|  |  |  |
| --- | --- | --- |
| Date | Completed Task | Pictures |
| 20th May | The idea for the project was created. The idea is to create a classic snake game. | NA |
| 22nd May | The game has been created however the coding became corrupt and further updates on the program were not allowed. |  |
| 27th May | After some time spent the program has been created again without any errors. The new program includes a main menu and the basic game. The start button on the main menu is linked to the game. An issue which came up was that the after a certain score the rocks did not have an effect on the snake. |  |
| 31st May | I have created a page for controls and rules and connected the two pages with the home pages. The issue from the last log entry has been fixed as now when the snake and rock collide it stops the game. |  |
| 3rd June | I created a page where users can play a tutorial and then continue to the game. The tutorial page is connected to the home page and the game page. |  |
| 7th June | Instead of having one game for the user to play, I have decided to create 5 different levels. The selection page for the game and levels has changed. |  |
| 13th June | I have created levels 1; 2 & 3 and they all connect to the level selection page. The 1st is at an extremely easy difficulty, 2nd level is at an easy difficulty and 3rd level is an average difficulty. |  |
| 16th June | I have created levels 4 & 5. Level 4 is at a hard difficulty and level 5 is at an extremely hard difficulty. I have added background music to the game aswell. |  |
| 18th June | In the infinite game mode, I added coding which changes the snake speed after the player hits certain scores |  |
| 19th June | I changed the level requirements for level 5 in order to suit the gaming skill of the intended users. |  |
| 21st June | I inserted comments into the coding to allow the easy understanding for users. |  |

**Algorithms**

**Pseudocode**

**Infinite Game Mode**

START SNAKEINFINITEGM

Score = 0

Timer = False

Snake = Picture box

Apple = Picture Box

Boulder = Picture Box

Rock = Picture Box

If User Input = A, W, D, S Then

Timer = True

Endif

If User = A Then

Snake = -10 Left\_Right

Snake = 0 Up\_Down

Elseif user = W Then

Snake = 0 Left\_Right

Snake = 10 Up\_Down

Elseif user = S Then

Snake = 0 Left\_Right

Snake = -10 Up\_Down

Elseif user = D Then

Snake = 10 Left\_Right

Snake = 0 Up\_Down

Endif

While Timer = True

If Snake intersects with Apple Then

Score += 1

Display Score AS Label

Apple = random location

Rocks = random location

Boulder = random location

Create Rock

Elseif Snake intersects with Boulder Then

Timer = false

Display Message Box “You collided with a boulder! Your score is” + Score

Score = 0

Open GameMode

Elseif Snake intersects with Rock Then

Timer = false

Display Message box “You collided with a rock! Your score is” + Score

Score = 0

Open GameMode  
Elseif Snake intersects with Wall Then

Timer = False

Display Message Box “You collided with the wall! Your score is” + Score

Score = 0

Open GameMode

Endif

Endwhile

END SNAKEINFINITEGM

**Level Game Mode**

START SNAKELVLG

Score = 0

Set score = {lv1 – 10, lv2 -15, lv3 – 10, lv4 – 15, lv5 – 7}

Timer = False

Snake = Picture box

Apple = Picture Box

Boulder = Picture Box

Rock = Picture Box

If User Input = A, W, D, S Then

Timer = True

Endif

If User = A Then

Snake = -10 Left\_Right

Snake = 0 Up\_Down

Elseif user = W Then

Snake = 0 Left\_Right

Snake = 10 Up\_Down

Elseif user = S Then

Snake = 0 Left\_Right

Snake = -10 Up\_Down

Elseif user = D Then

Snake = 10 Left\_Right

Snake = 0 Up\_Down

Endif

While Timer = True

If Snake intersects with Apple Then

Score += 1

Display score as label

Apple = random location

Rocks = random location

Boulder = random location

Create Rock

If score = set score Then

Open NextPage

Endif

If Snake intersects with Rock Then

Timer = false

Display Message box “You collided with a rock!”

Score = 0

Reset Form

Elseif Snake intersects with Wall Then

Timer = false

Display Message box “You collided with the wall”

Score = 0

Reset Form

Elseif Snake intersects with Hedge Then

Timer = false

Display Message box “You collided with the hedge”

Score = 0

Reset Form

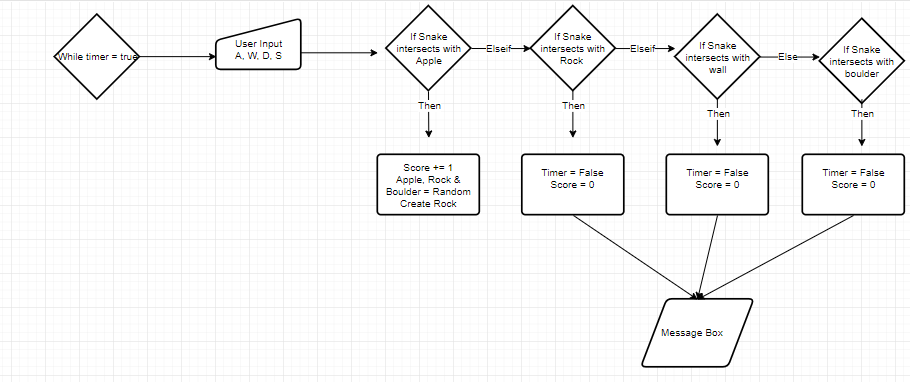
Endif

Endwhile

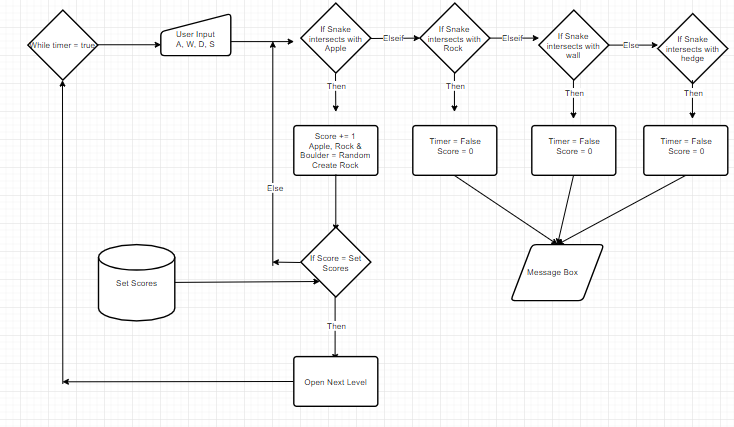
END SNAKELVLG

**pchart**

**Infinite Game Mode**



**Level Game Mode**



**Desk Checking**

**Infinite Game Mode Desk Check**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Line Number | User Input | Collision with apple | Collision with rock | Collision with wall | Collision with Boulder | Output |
| 1 | W, A, D, S | False | False | False | False | Snake keeps moving |
| 2 | W, A, D, S | True | False | False | False | A point gets added to the score. Apple will move to a random place on the playing field. The rocks will move to a random place on the playing field. The point will be displayed on the screen. |
| 3 | W, A, D, S | False | True | False | False | The snake will stop. A message box will pop up indicating that the user has collided with a rock and will state the players score. The user will then be taken to the ‘select game mode’ page. |
| 4 | W, A, D, S | False | False | True | False | The snake will stop. A message box will pop up indicating that the user has collided with the walls and will state the players score. The user will then be taken to the ‘select game mode’ page. |
| 5 | W, A, D, S | False | False | False | True | The snake will stop. A message box will pop up indicating that the user has collided with the boulder and will state the players score. The user will then be taken to the ‘select game mode’ page. |

**Levels Game Mode Desk Check**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Line Number | User Input | Collision with apple | Collision with rock | Collision with wall | Collision with Hedge | Output |
| 1 | W, A, D, S | False | False | False | False | Snake keeps moving |
| 2 | W, A, D, S | True | False | False | False | A point gets added to the score, the progress bar moves up. Apple will move to a random place within the playing field, a rock will be added to the game in a random place within the playing field. If the score reaches the required limit to pass the level a message box will pop indicating the user has passed the level and then will be directed to the next level or if the last level is passed the user will be directed to the level selection page. |
| 3 | W, A, D, S | False | True | False | False | The snake will stop. A message box will pop up indicating that the user has collided with a rock and the level will start again. |
| 4 | W, A, D, S | False | False | True | False | The snake will stop. A message box will pop up indicating that the user has collided with the wall and the level will start again. |
| 5 | W, A, D, S | False | False | False | True | The snake will stop. A message box will pop up indicating that the user has collided with a hedge and the level will start again. |

**User Interface Desk Check**

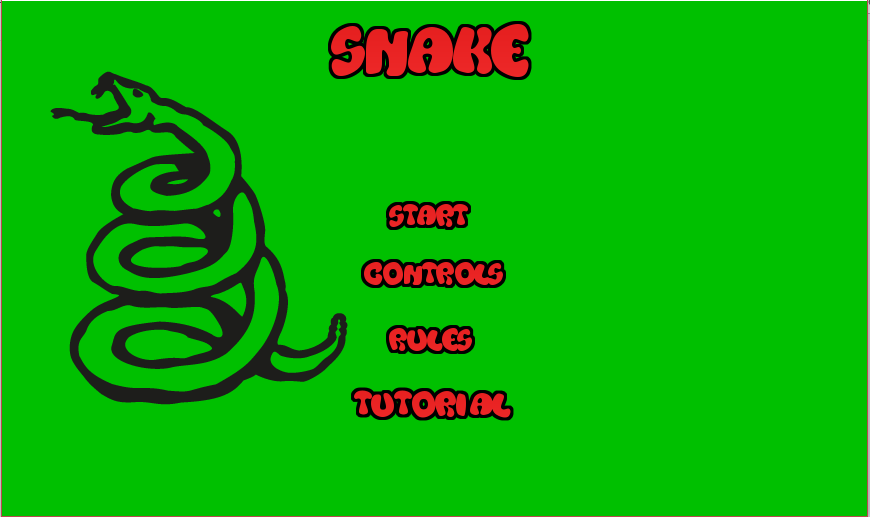
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Line Number | User Input | Requirements Met | Output | Photo |
| 1 | Start Button | True | The user is taken to the ‘game mode select’ page |  |
| 2 | Controls Button | True | The user is taken to the ‘controls’ page |  |
| 3 | Rules Button | True | The user is taken to the ‘rules’ page |  |
| 4 | Tutorial Button | True | The user is take to the ‘tutorial’ page |  |
| 5 | Levels Button | True | The user is take to the ‘level select’ page |  |
| 6 | Infinite Button | True | The user is take to the infinite game mode |  |
| 7 | 1, 2, 3, 4, 5 Button | True | The user is take to the respective level |  |
| 8 | Arrow Button | True | The user is take to the previous page |  |

**Structured Walkthrough**

The process of Quality Assurance during software development is a pivotal one. It is one of the most significant phases of the Software Development Life Cycle that requires a great deal of planning and testing. With the assistance of quality assurance, one can not only validate the quality of the product, but also ensure that the end product has exceptional effectiveness, performance and functionality. Moreover, the tests conducted by the quality assurance team guarantees that the product is developed as per the demands and requirements of the client and it provides expected and accurate results. After a carefully viewing the algorithms it is safe to say that quality of the of the algorithms is at high standard after a series of tests with peers.

**Building Software Solutions**

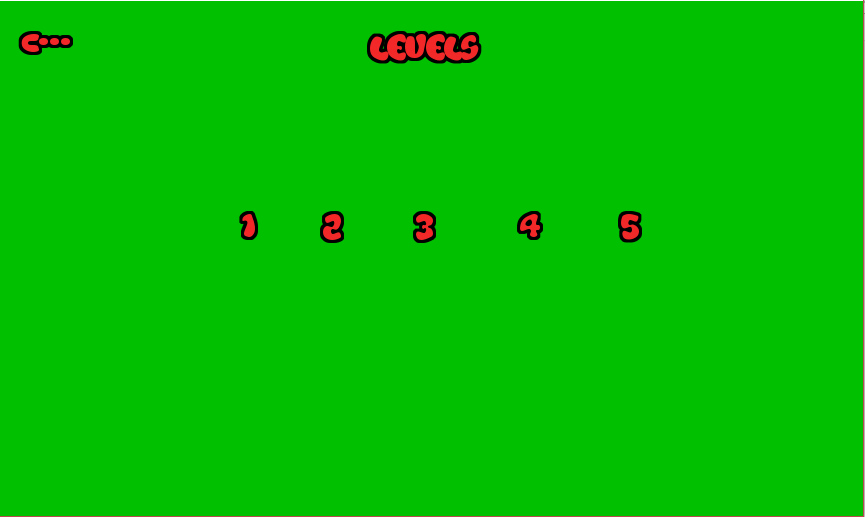
**Story Board**



The 4 options are picture boxes which act as buttons. The contrast between the green and red has been stated to show good contrast in the peer test.



The ‘levels’, ‘infinite’ and the arrow are all also picture boxes which act as buttons

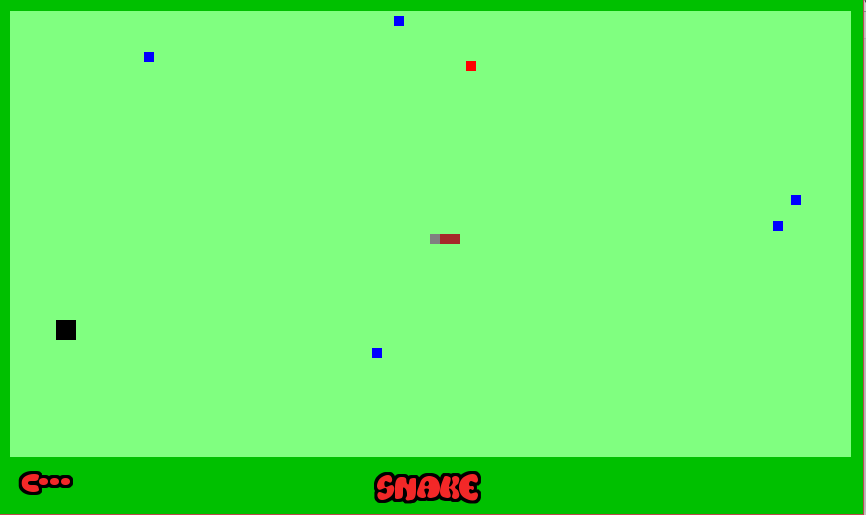


All the numbers and the arrow are picture boxes which act as buttons



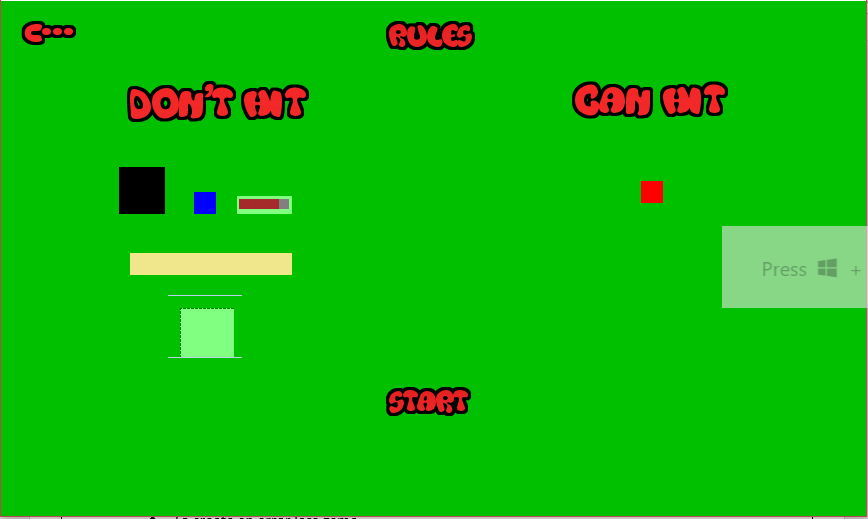
All the levels have a similar look. The snake, apple, rock, hedges are picture boxes. There is a progress bar in each level to show how close the user is to finishing the level. The arrow is a picture box which acts as a button.

The infinite game mode has snakes, boulders, rocks and apples which are picture boxes. The arrow button once again acts as a button but is a picture box. In the bottom right hand corner there is a label which shows the player score progress through the game. After the game finishes a message box indicates the player score and how the game stopped.





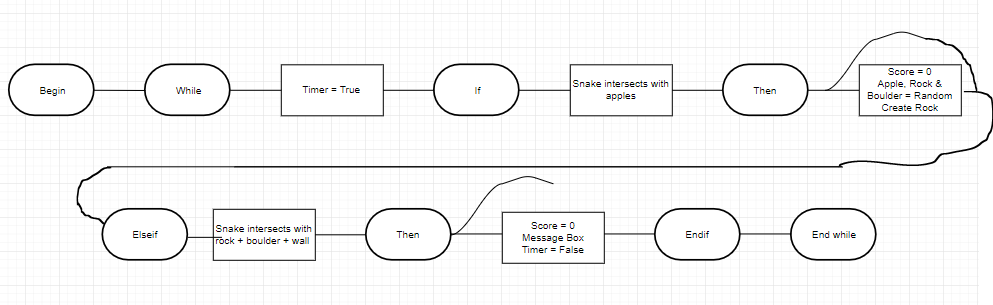
In these two pages they both have the start and arrow button which are just picture boxes acting as buttons. Everything else on the screen are also picture boxes





In the tutorial page the arrow once again acts as a button when in reality is a picture box. After the snake collects the apple a message box pops up indicating that the tutorial is finished and after takes you to the ‘select level’ page

**Railroad Diagrams**





**EBNF Diagrams**

BEGIN (While) <Timer = True> (If) <Snake interacts with apples> (Then) = <score += 1, Apple, Rock & Boulder = Random, Create Rock> {< score = 0, Apple, Rock & Boulder = Random, Create Rock >} (Else if) <Snake intersects with Rock + Boulder + Wall> (Then) = <score = 0, message box, timer = false> {<score = 0, message box, timer = false>} (End if) (End while)

**Checking Software Solutions**

There are different testing processes for software solutions. They are:

**Alpha Testing:**

Alpha Testing refers to the testing of the final solution by personnel within the software development company prior to the products release

**Beta Testing**

Beta testing refers to the testing of the final solution by a limited number of users outside the software development company using real world data and conditions.

**Black Box Testing**

Also known as functional testing. The inputs and expected outputs are known; the processes occurring are unknown.

**White Box Testing**

Also known as structural, or open box testing. A software testing technique whereby explicit knowledge of the internal workings of the item being tested is used.

**System Level Testing**

System-level testing aims to ensure that the hardware, software, data, personnel and procedures that form the components of the final system are able to work together efficiently, correctly and in the manner intended with the new software product.

**Volume Data (Load Testing)**

Large amounts of data should be entered into the new software system to test the application under extreme load conditions. Muti-user products should be tested with large numbers of users entering and processing data simultaneously.

**Benchmarking**

Benchmarking is the process of evaluating a product in which from a point of reference from which quality or excellence is measured.

**Quality Assurance**

Quality assurance is about evaluating how well the software product meets or exceeds the users’ expectations based on correctness, reliability, efficiency, integrity, useability, maintainability, flexibility, testability, portability, re-usability and interoperability.

**Acceptance Testing**

Acceptance Testing is when formal tests are conducted to verify whether or not a system meets its requirements. Acceptance testing enables the client to determine whether or not to accept the new system.

The testing process which I used was quality assurance, beta testing and alpha testing. These testing processes were used due to their efficiency and reliability.

**User Testing**

|  |  |  |
| --- | --- | --- |
| Name | Positives | Improvements |
| Eric Kim | * Simple and clear instruction and format, to be easily accessed by anyone. | * More dynamic levels |
| Parasdeep Bindra | * Simple and fun game to play with different game modes (levels and infinite mode) * Easy to navigate | * Improvement on graphics |

**Evaluating the Software Solution**

The objectives that were identified at the start of the task were:

* To create an error less game
* To create a game which is suitable for kids ranged between 3-5
* To create a game which will be which kids will enjoy playing within their free time
* To allow the kids to understand how to play the game
* Create a game which doesn’t take up time
* To create a user interface which is easy to use
* The program should be free of malware
* Take into consideration and oblige by social and ethical issues

In reference to the objectives that were specified at the beginning, the project has met all of its expectations and has further exceeded them. Not only does the program tick all of the objectives it also ticks the implied conditions of the program itself.

**Peer Testing**

|  |  |  |
| --- | --- | --- |
| Name | Positives | Improvements |
| Samuel Wibawa | * Meets the set objectives. * Good for older kids as well. * Good user interface as it is easy to use * Game is colourful * Good for kids * Has nice contrast in colour * Not too difficult * Challenging but doable | * More levels * Change the music * Make the game more visual |
| Myles Pritchett | * Meets the objectives you set out * User interface is consistent * Easy to use | * More levels * Try adding different soundtracks |

**Modifying the Software Solution**

**Modification 1**

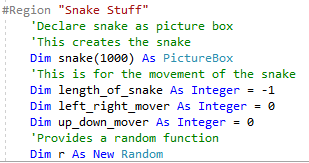
The first modification that may need to be made to the program is the improvement on the graphics of the game. The game was built in order to resemble the retro game snake however through the use of peer and user testing it was evident that making the game ‘cartoony’ would appeal more to the intended audience. Solutions for this is to import a photo file which shows the cartoony aspects in the game which can then be placed within the different components. The background, snake, apple, rock and hedge are all currently picture boxes which don’t hold any images instead have a back colour. In place of the back colour we would put the new images.

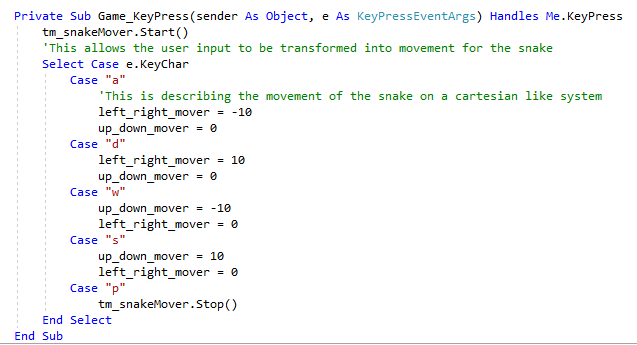
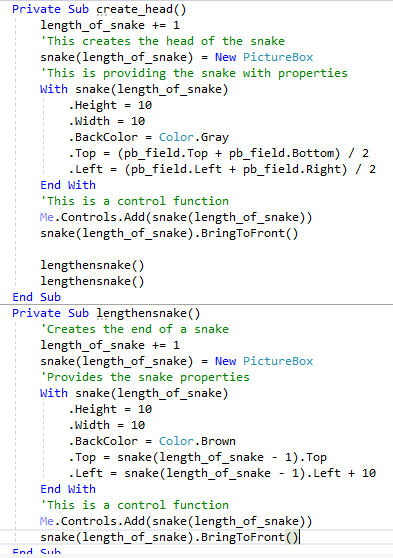
**Modification 2**

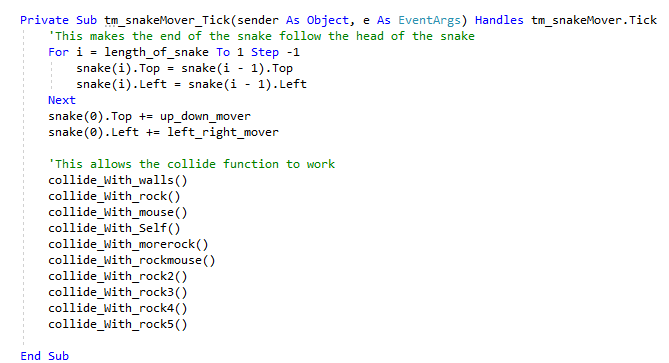
The second modification that may need to be made to the program is the creation of more levels. The game’s structure is that the user is able to select 5 different levels or an infinite mode in which they can try going for as long as they can. Through the user and peer testing it was apparent that there was a scarcity of levels. In further updates of the program, further levels would be added in at different skill requirements. The solution to this is to create more forms on visual basic and generate different level designs.

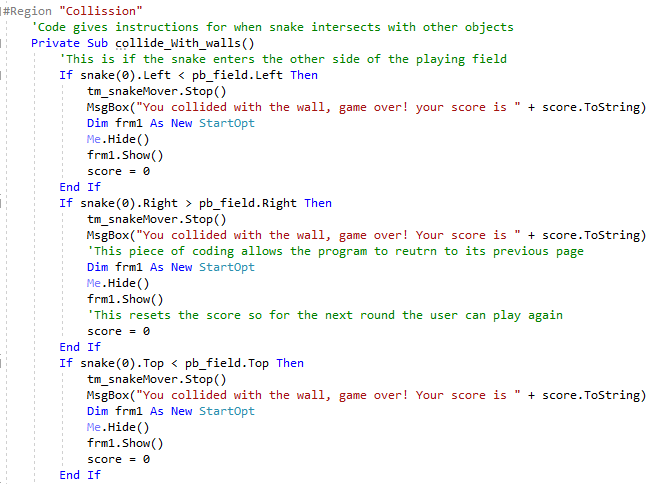
**Intrinsic Documentation**

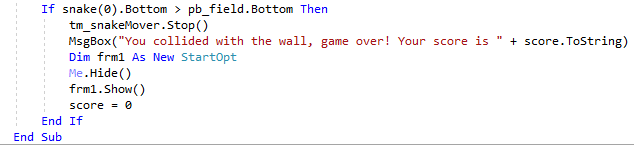
The following intrinsic documentation is taken from the infinite game mode and provides helpful information about the different functions within the code. The code in the level game mode also has the documentation however there is excessive repetition within the coding. The following documentation demonstrates all the supporting information that is required for the user to fully understand how the program works.

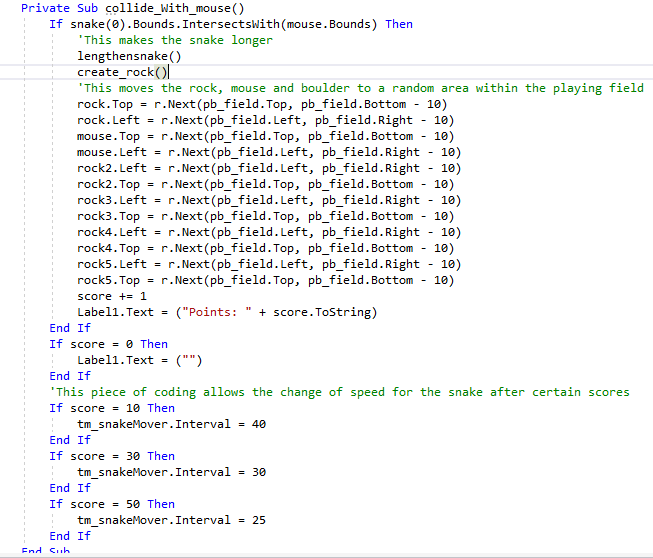


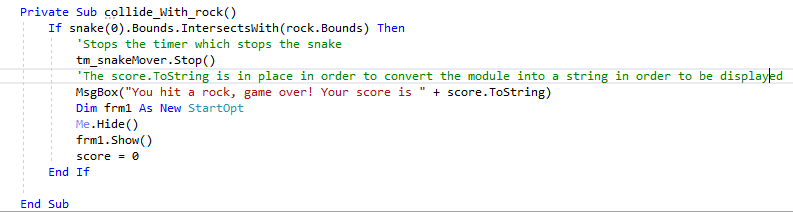


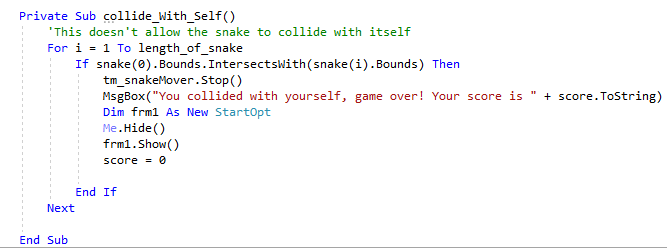


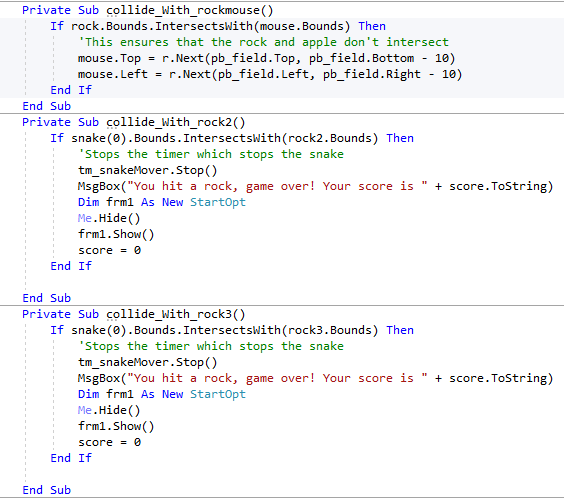


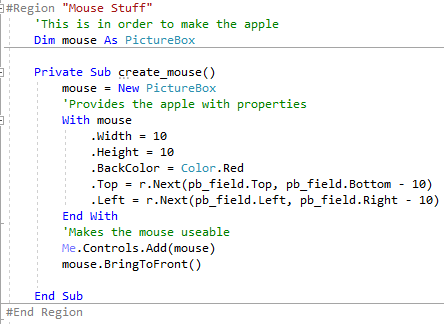


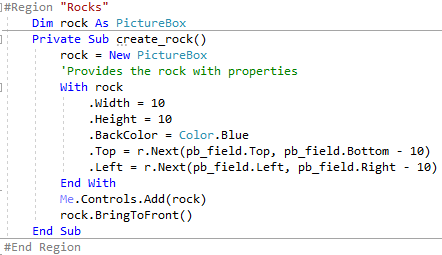


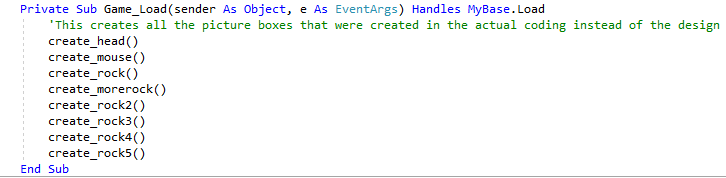


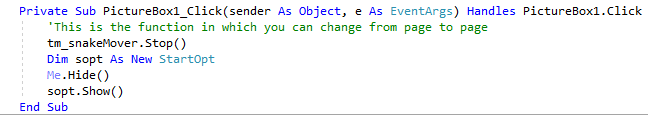


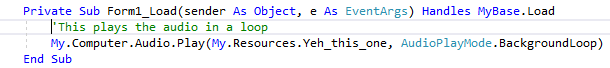












Prathamesh Tarde