Visvesvaraya Technological University

Belagavi-590 014, Karnataka

A Mini Project Report on

"Snake Game"

Submitted in partial fulfilment of the requirements for the award of

Bachelor of Engineering

in

Computer Science and Engineering

Submitted By

Manik M Vernekar **USN 2JI18CS026**

Sanket Sambhaji Patil **USN 2JI18CS031**

Under the Guidance of

Prof. PRATIK K SAYANAK

Department of Computer Science and Engineering

Sri Bhagawan Mahaveer Jain Educational & Cultural Trust's

Jain College of Engineering

Belagavi-590 014

Academic Year 2020-21

Sri Bhagawan Mahaveer Jain Educational & Cultural Trust's

Jain College of Engineering

Belagavi-590 014

Department of Computer Science and Engineering

Certificate

This is to certify that the mini-project entitled **"SNAKE GAME"** is carried out by **Mr. SANKET SAMBHAJI PATIL,** bearing **USN-2JI18CS031,** a bonafide student of **Jain College of Engineering, Belagavi**, in partial fulfilment for the award of **Bachelor of Engineering** in **Computer Science and Engineering** from **Visvesvaraya Technological University, Belagavi**, during the academic year **202021.** It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The mini project report has been approved as it satisfies the academic requirements in respect of **Computer Graphics Laboratory with Mini Project** prescribed for the said degree.

Prof. Pratik K Sayanak
Guide

Prof. Pavan Ughade
HOD, CSE

Name of Examiner

_____

_____

1.

2.

Signature of Examiner

1. _____

2. _____

Sri Bhagawan Mahaveer Jain Educational & Cultural Trust's

Jain College of Engineering
Belagavi-590 014

Department of Computer Science and Engineering

Certificate

This is to certify that the mini-project entitled **"SNAKE GAME"** is carried out by **Mr. Manik M Vernekar (2JI18CS026)** and **Mr. Sanket Sambhaji Patil (2JI18CS031),** bonafide students of **Jain College of Engineering, Belagavi**, in partial fulfillment for the award of **Bachelor of Engineering** in **Computer Science and Engineering** from **Visvesvaraya Technological University, Belagavi**, during the academic year **2020-21.** It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The mini project report has been approved as it satisfies the academic requirements in respect of **Computer Graphics Laboratory with Mini Project** prescribed for the said degree.

Prof. Pratik K Sayanak
Guide

Prof. Pavan Ughade
HOD, CSE

Name of Examiner

1. _____

2. _____

Signature of Examiner

1. _____

2. _____

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the progress and completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement ground my efforts with success.

I consider it is a privilege to express my sincere gratitude and respect to all those who guided and inspired me.

I express my sincere thanks and gratitude to our guide **Prof. Pratik K Sayanak,** Department of Computer Science & Engineering, JCE, Belagavi, for his constant guidance and suggestions. His incessant encouragement and invaluable support has been of immense help.

It's a great privilege to express my respect to **Prof. Pavan Ughade,** HOD, Department of Computer Science & Engineering, JCE, Belagavi, who had been a great source of inspiration towards taking up this mini-project and its successful completion.

I am thankful to **Dr. K. G. Vishwanath**, Principal, JCE, Belagavi for providing us with the necessary facilities for carrying out this project work successfully.

ABSTRACT

This mini project demonstrates a snake game using openGL primitive functions. This project includes colourful graphics display representation using openGL as programming language. In this project, we are creating a virtual graphical representation. The snake game works with the help of built-in graphics library functions. The display screen consists of black screen and a snake which searches for the food pops up randomly on the screen which can be eaten by the snake. The snake has a head which eats the food pops. It has a tail which increases when it eats the food pops. It can be implemented using openGL functions that creates a colourful attractive snake game. The player can move the snake from one position to another position by using keyboard.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction to Computer Graphics

COMPUTER GRAPHICS is concerned with all aspects of producing pictures or images using a computer. The field began humbly almost 50 years ago, with the display of a few lines on a cathode-ray tube (CRT); now, we can create images by computer that are indistinguishable from photographs of real objects. We routinely train pilots with simulated airplanes, generating graphical displays of a virtual environment in real time. Feature-length movies made entirely by computer have been successful, both critically and financially. Massive multiplayer games can involve tens of thousands of concurrent participants.

VISUALIZATION is any technique for creating images, diagrams or animations to communicate a message.

1.2 Image Types

○ 2D computer graphics:

2D computer graphics are the computer-based generation of digital images mostly from twodimensional models, such as 2D geometric models, text, and digital images, and by techniques specific to them.2D computer graphics are mainly used in applications that were originally developed upon traditional printing and drawing technologies, such as typography, cartography, technical drawing, and advertising. Two-dimensional models are preferred, because they give more direct control of the image than 3D computer graphics, whose approach is more akin to photography than to typography.

There are two approaches to 2D graphics: vector and raster graphics.

- Pixel art: Pixel art is a form of digital art, created through the use of raster graphics software, where images are edited on the pixel level.
- Vector graphics: Vector graphics formats are complementary to raster graphics, which is the representation of images as an array of pixels, as it is typically used for the representation of photographic images.

○ 3D computer graphics:

With the birth of the workstation computers(like LISP machines, paint box computers and Silicon Graphics workstations)came the 3D computer graphics.3D computer graphics in contrast to 2D computer graphics are graphics that use a three-dimensional representation of geometric data that is stored in the computer for the purposes0 of performing calculations and rendering 2D images.

1.3 Applications of Computer Graphics

Some of the applications of computer graphics are listed below:

- Computational biology
- Computational physics
- Computer-aided design
- Computer simulation
- Digital art
- Education
- Graphic design
- Video Games
- Virtual reality
- Web design

1.4 Introduction to OpenGL

As a software interface for graphics hardware, OpenGL's main purpose is to render two- and three-dimensional objects into a frame buffer. These objects are described as sequences of vertices (which define geometric objects) or pixels (which define images). OpenGL performs several processing steps on this data to convert it to pixels to form the final desired image in the frame buffer.

Basic OpenGL Operation

The figure shown below gives an abstract, high-level block diagram of how OpenGL processes data. In the diagram, commands enter from the left and proceed through what can be thought of as

a processing pipeline. Some commands specify geometric objects to be drawn, and others control how the objects are handled during the various processing stages.

1.5 Introduction to GLUT

GLUT is the OpenGL Utility Toolkit, a window system independent toolkit for writing OpenGL programs. It implements a simple windowing application programming interface (API) for OpenGL. GLUT makes it considerably easier to learn about and explore OpenGL programming. GLUT provides a portable API so you can write a single OpenGL program that works on both Win32 PCs and X11 workstations.

GLUT is designed for constructing small to medium sized OpenGL programs. While GLUT is well-suited to learning OpenGL and developing simple OpenGL applications, GLUT is not a full-featured toolkit so large applications requiring sophisticated user interfaces are better off using native window system toolkits like Motif. GLUT is simple, easy, and small. My intent is to keep GLUT that way.

The GLUT library supports the following functionality:

- Multiple windows for OpenGL rendering.
- Call back driven event processing.
- An `idle' routine and timers.
- Utility routines to generate various solid and wire frame objects.
- Support for bitmap and stroke fonts.
- Miscellaneous window management functions.

1.6 Overview of the project

OpenGL is a software interface to graphics hardware. This interface consists of about many distinct commands that you use to specify the objects and operations needed to produce interactive three-dimensional applications. This project, named Snake Game uses OpenGL software interface and develops 2D game. Snake Game is game in which food pops up randomly on the screen which is eaten by the Snake. Task is to direct the worm using Arrow keys for up, down, left and right directions respectively. The Snake moves and eats the food as soon as the head of the Snake touches the food. With each food the Snake eats, it grows bigger by one matrix

1.7 Aim of the project

The aim of this project is to create a 2D Snake Game. The Snake moves freely on the window, moves upward, downward, leftward and rightward directions when respective keys are pressed.

CHAPTER 2

SOFTWARE REQUIREMENT SPECIFICATION

2.1 Minimum Software Requirements

- Operating System : Windows 98/XP or Higher
- Programming Language : C,C++
- Microsoft Visual Studio 2005 or higher: This Software package containing visual basics in C++ language is required.
- Toolkit : GLUT Toolkit, VC++

2.2 Minimum Hardware Requirements

This package has been developed on:

- Processor : Pentium Processor
- Processor Speed : 333 MHz
- RAM : 32 MB or Higher
- Graphics Card : 512MB
- Monitor : Color
- Keyboard : Low Profile, Dispatch able Type
- I/O Parts : Mouse, Monitor

CHAPTER 3

DESIGN

3.1 Initialization

Initialize the interaction with the windows. Initialize the display mode, double buffer and depth buffer. Initialize the various callback functions for drawing and redrawing arrows and target

blocks, for mouse interface. Initialize the window position and size and create the window to display the output.

3.2Flowchart

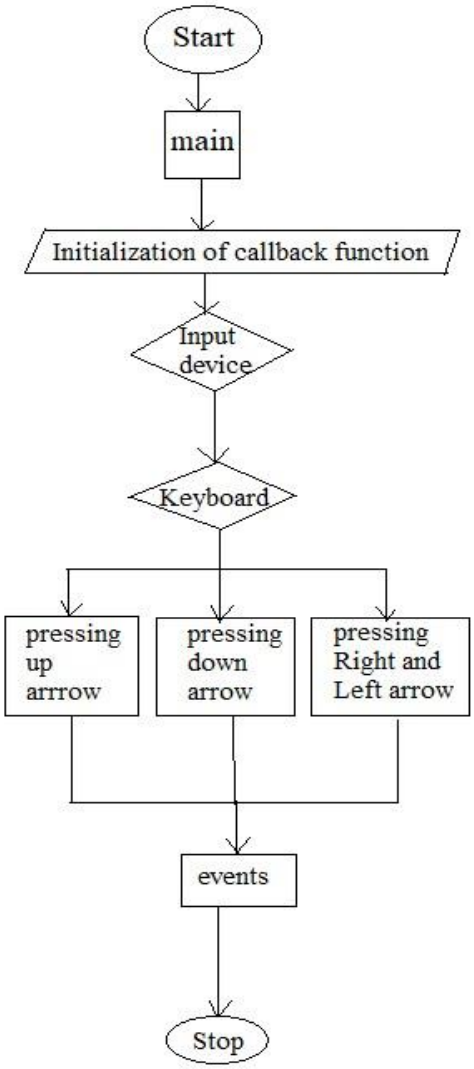

Fig 3.1: Flowchart for Snake Game program.

3.3 Flow of control

The flow of control in the flowchart is with respect to the Snake Game. For any program flowchart is compulsory to understand the program. We consider the flowchart for the snake game in which the flow starts from the start and proceeds to the main function after which it comes to the

initialization of callback functions and further it proceeds to keyboard function, the flow comes to quit when Snake is unable to pass through wall which is the end of the flowchart.

CHAPTER 4

IMPLEMENTATION

4.1 INBUILT FUNCTIONS

4.1.1 BASIC FUNCTIONS

- **glColor3f Function**

  Sets the current color.

  SYNTAX: Void glColor3f (GLfloat red, GLfloat green, GLfloat blue);

PARAMETERS:

- Red - The new red value for the current color.
- Green - The new green value for the current color.
- Blue -The new blue value for the current color.

- **glBegin, glEnd Function**

  The glBegin and glEnd function delimit the vertices of a primitive or a group of like primitives.

  SYNTAX: void glBegin, glEnd(GLenum mode); PARAMETERS:

- mode - The primitive or primitives that will be created from vertices presented between glBegin and the subsequent glEnd. The following are accepted symbolic constants and their meanings.

GL_LINES-

Treats each pair of vertices as an independent line segment. Vertices 2n-1 and 2n define line n.N/2 lines are drawn.

GL_QUADS-

Treats each group of four vertices as an independent quadrilateral. Vertices 4n-3, 4n-2, 4n-1 and 4n defined quadrilaterals are drawn.

- **glVertex2f Function**

  Specifies a vertex.

SYNTAX: Void glVertex2f(GLfloat x,GLfloat y,);

PARAMETERS:

- X- Specifies the x-coordinate of a vertex.
- Y- Specifies the y-coordinate of a vertex. EXAMPLE- glVertex2f(-3.0,3.0);

- **glLineWidth Function**

  specify the width of rasterized lines.

  SYNTAX: void glLineWidth(GLfloat *width*);

- **glRectf Function**

  The glRectf function draws a rectangle.

  SYNTAX: void

```
WINAPI glRectf(
  GLfloat x1,
  GLfloat y1,
  GLfloat x2,
  GLfloat y2
);
```

- **glRectd Function**

  The glRectd function draws a rectangle.

  SYNTAX: void

```
WINAPI glRectd(
  GLdouble x1,
  GLdouble y1,
  GLdouble x2,
  GLdouble y2
);
```

4.1.2 FUNCTIONS USED TO DISPLAY

- **glClear Function**

  The glClear function clears buffers to preset values.

  SYNTAX:

  glClear(GLbitfield mask); PARAMETERS:

- mask- Bitwise OR operations of masks that indicate the buffers to be cleared. The four masks are as follows.

Value	Meaning
GL_COLOR_BUFFER_BIT	The buffers currently enable for colorwriting
GL_DEPTH_BUFFER_BIT	The depth buffer.

Eg: glclear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);

- **glMatrixMode Function**

The gLMatrix Mode function specifies which matrix is the current matrix.

SYNTAX :

Void gLMatrix Model(GLenum mode); PARAMETERS :

- Mode – The matrix stack that is the target for subsequent matrix operations. The mode parameter can assume one of three values:

Value	Meaning
GL_MODEL VIEW	Applies sub sequent matrix operations to the Modelview matrix stack.

Eg: gLMatrix Mode(GL_MODELVIEW) :

- **glLoadIdentity Function**

The glLadIdentity function replaces the current matrix with the identity matrix.

SYNTAX :

Void glLoadIdentity(void); PARAMETERS :

This function has no parameters.

- **glClear Color Function**

glclearColor used for clearing the color buffers.

SYNTAX :

Void glclearColor(GLclampf r, GLclampf g, GLclampf b, GLclampf

a)Variables of type GLclampf are floating point numbers between 0.0 and 1.0.

- **glutPostRedisplay Function**

  This function requests the display callback be executed after the current callback returns.

  SYNTAX :

  Void glutPostRedisplay(void);

- **glutSwapBuffers Function**

  glutSwapBuffers swaps the buffers of the *current window* if double buffered.

  SYNTAX :

  void glutSwapBuffers(void);

4.1.3 FUNCTIONS USED TO SET THE VIEWING VALUE

- **glOrtho Function**

This function defines orthographic viewing volume with all parameters measured from the centre of projection.

Multiply the current matrix by a perspective matrix.

SYNTAX:

Void glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble nearVal, GLdouble farVal); PARAMETERS :

- Left, right – Specify the coordinates for the left and right vertical clipping planes.
- Bottom, top – Specify the coordinates for the bottom and top and horizontal clipping plane.
- near Val, far Val – Specify the coordinates for the bottom and top horizontal clipping planes.

- nearVal, farVal – Specify the distances to the nearer and farther depth clipping planes. These values are negative if the plane is to be behind the viewer.

- **glViewport Function**

Set the viewport.

SYNTAX:

void glViewport(GLint *x*, GLint *y*, GLsizei *width*, GLsizei *height*);

4.1.4 CALL BACK FUNCTIONS

- **glutDisplayFunc Function**

glutDisplayFunc sets the display callback for the current window.

SYNTAX:

Void glutDisplayFunc(void(*func)(void)); PARAMATERS:

- func - The new display callback function.

EG: glutDisplayFunc(display):

- **glutReshapeFunc Function**

glutReshapeFunc Function sets the reshape callback for the current window.

SYNTAX:

Void glutReshapeFunc(void(*func)(int width, int height));

PARAMATERS:

- Func - The new Idle callback function.

Eg: glutReshapeFunc(reshape);

4.1.5 MAIN FUNCTION

- **glutInit Function**

glutInit is used to initialize the GLUT library.

SYNTAX : gluInit(int *argcp, char

**argv);

PARAMATERS:

* argcp – A pointer to the program's unmodified argc variable from main. Upon return, the value pointer to by argcp will be updated, because gluInit extracts any command line options intended for the GLUT library.

* argv – The program's unmodified argv variable from main. Like argcp, the data for argv will be updated becaue glutInit extracts any command line options understood by the GLUT library. Eg: glutInit(&argc,argv);

- **glutInitDisplayMode Function**

glutInitDisplayMode sets the initial display mode.

SYNTAX :

Void glutInitDisplayMode(unsigned int mode); PARAMATERS:

* mode – Display mode, normally the bitwise OR-ing of GLUT display mode bit masks. See values below:

GLUT_RGB: An alias for GLUT_RGBA.

GLUT_DOUBLE: Bit mask to select a double buffered window. This overrides GLUT_SINGLE if it is also specified.

GLUT_DEPTH: Bit mak to select a window with a depth buffer.

Eg:

glutInitDisplayMode(GLUT_RGB/GLUT_DEPTH/GLUT_DOUBLE);

- **gluInitWindowPosition, glutInitWindowSize Functions**

glutInitWindowPosition and glutInitWindowsize set the initial window position and size respectively.

SYNTAX:

Void glutInitWindowSize(int width, int height);

Void glutInitWindowPosition(int x, int y); PARAMATERS:

* width – Width in pixels.

* height – Height in pixels.

* x – Window X location in pixels. * y – Window Y location in pixels.

Eg: glutInitWindowsSize(300,300);

- **glutCreateWindow Function**

glutCreateWindow creates a top-level window.

SYNTAX: int glutCreateWindow(char *name)' PARAMATERS:

* name – ASCII character string for use as window name. Eg: glutCreateWindow("Snake game"),

- **glutMainLoop Function**

glutMainLoop enters the GLUT event processing loop.

SYNTAX:

Void glutMainLoop(void),

Eg: glutMainLoop();

- **glutTimerfunc Function**

glutTimerFun registers a timer callback to be triggered in a specified number of milliseconds.

SYNTAX:

void glutTimerFunc(unsigned int msecs,
void (*func)(int value), value);

- **glutSpecialFunc Function**

glutSpecialFunc sets the special keyboard callback for the current window.

SYNTAX:

void glutSpecialFunc(void (*func)(int key, int x, int y));

CHAPTER 5

TESTING

Test Cases

Insert for Test Case	Expected Output	Observed Output	Remarks
UP KEY	It should Go in Upward Direction	It performs movement in Upward Direction.	Pass.
DOWN KEY	It should Go in Downward Direction	It performs movement in Downward Direction.	Pass.
LEFT KEY	It should Go in leftward Direction	It performs movement in Leftward Direction.	Pass
RIGHT KEY	It should Go in Rightward Direction	It performs movement in Rightward Direction.	Pass
Other key pressed	The program should get affected.	The program does not get affected.	Fail.

Fig 5.1: Testing

CHAPTER 6

RESULTS

With the left key the Snake moves in the left direction. With the right key the Snake moves in the right direction. With the upward key the Snake moves in the upward direction. With the downward key the Snake moves in the Downward direction. A 2D view of the Snake is created.

With the help of these keys we can make a few moves.

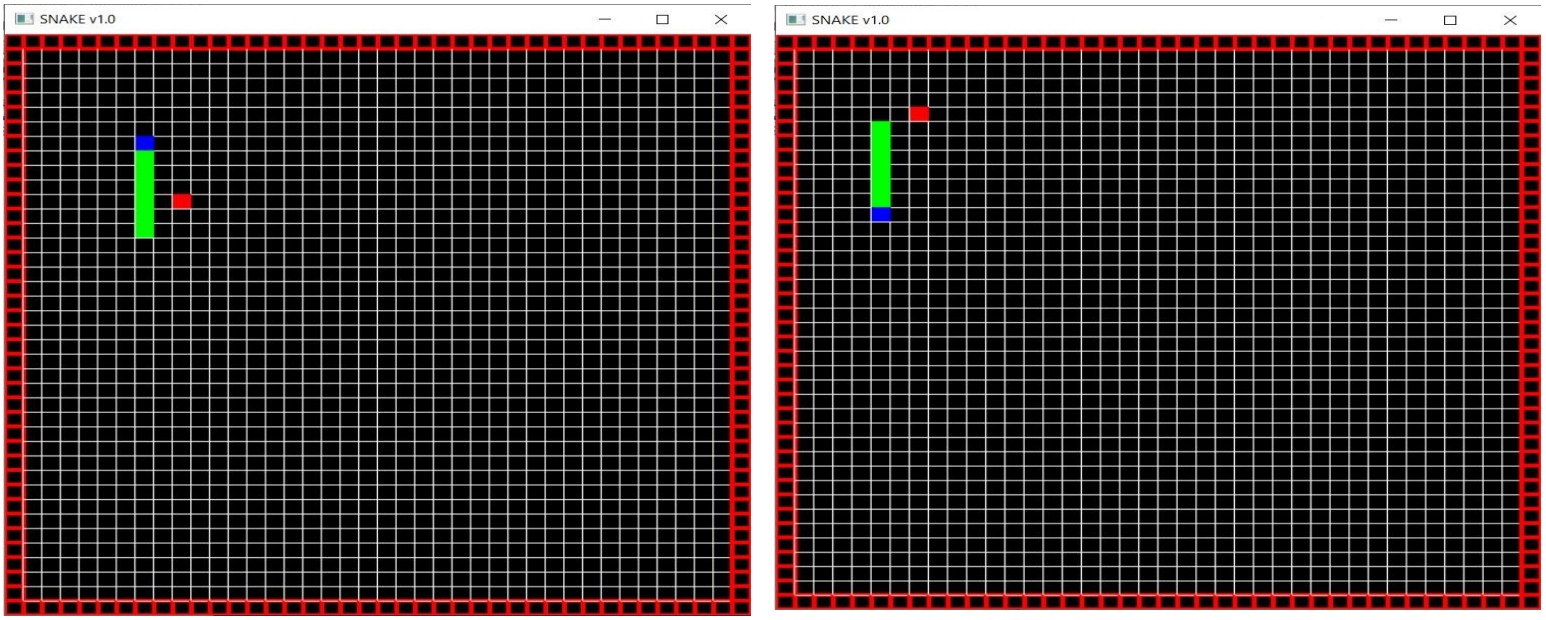

Fig 6.1: Upward And Downward movement using Arrow keys

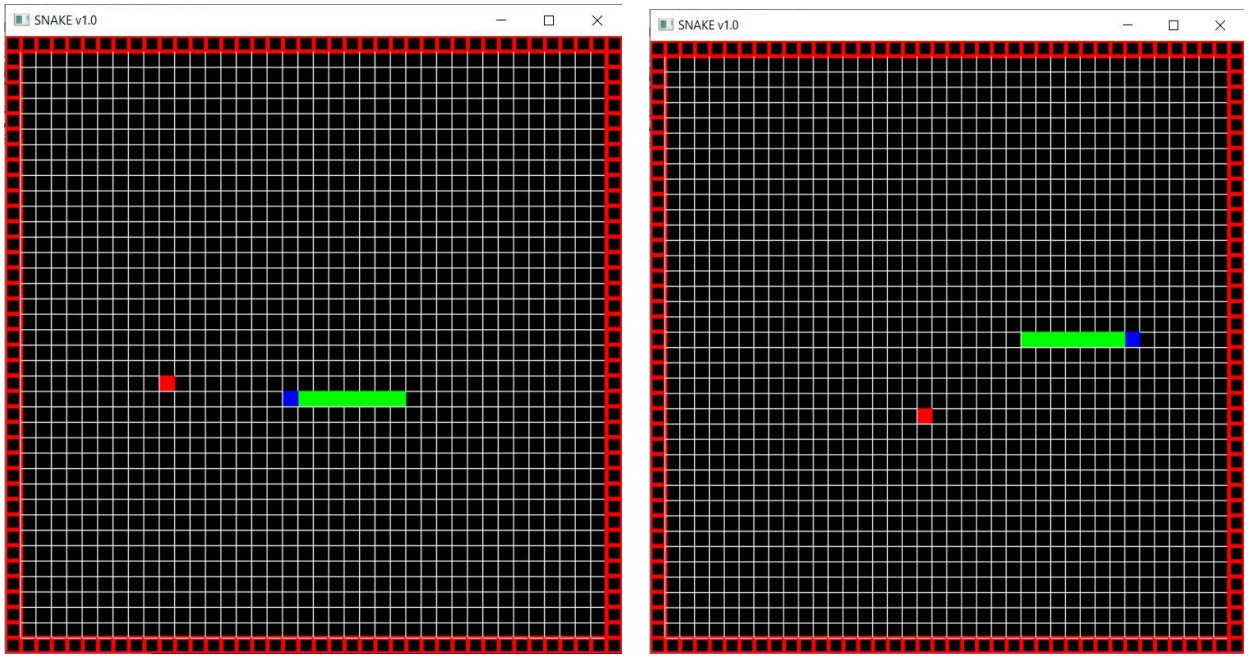

Fig 6.1: Leftward And Rightward movement using Arrow keys

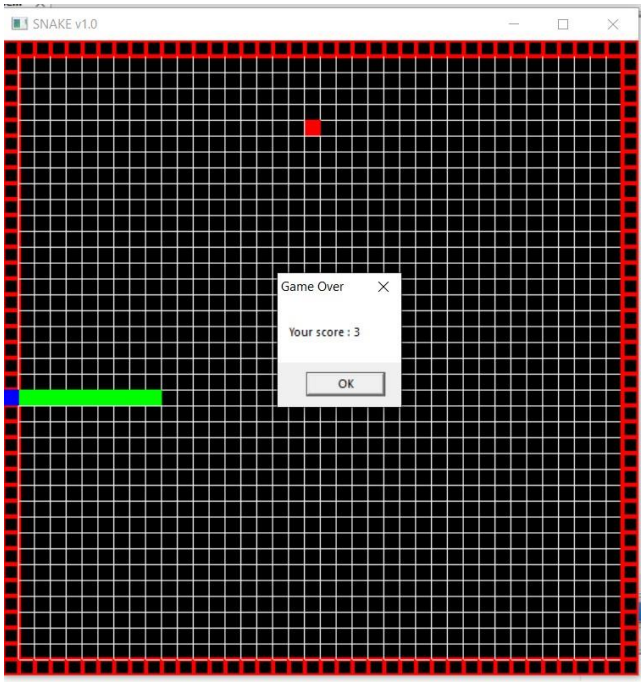

Fig 6.2: Game Over

Conclusion

A Snake Game Graphics package has been developed Using OpenGL.The illustration of graphical principles and OpenGl features are included and application program is efficiently developed.

The aim in developing this program was to design a simple program using Open GL application software by applying the skills we learnt in class, and in doing so, to understand the algorithms and the techniques underlying interactive graphics better.

The designed program will incorporate all the basic properties that a simple program must possess.

The program is user friendly as the only skill required in executing this program is the knowledge of graphics.

The Main idea of the program is to create an interactive game, fun to play.

References

[1]. Edward Angel *"Interactive Computer Graphics"* Pearson Education, 5th Edition.2008

[2]. Donald Hearn & Pauline baker *"Computer Graphics with OpenGL"* 3rd Edition. 2011

[3].www.opengl.org

[4]. www.freeglut.sourcefourge.net

[5]. www.github.com

[6]. www.openglcg.blogspot.in