PROJECT REPORT

CSE-422

Computer Graphics Lab

Submitted to:

Tanim Ahmed

Lecturer,Department of CSE

Daffodil International University

Submitted by:

Md.Jannatul Ferdous (191-15-2497)

Mahin Ahmed (191-15-1029)

Ibrahim Hossain Tushar (192-15-1064)

Jannatul Mawa (192-15-1063)

Naimur Rahman (192-15-1069)

Section: PC-A

Department of CSE

Daffodil International University

Submission Date: 11/10/2022

PREFACE

This report has been prepared to the **Natural Scenery named "Beauty Of Nature"**. This document is the final report of the findings and recommendations of the Computer Graphics Lab (CSE-422). Computer graphics remains one of the most exciting and rapidly growing computer fields. Computer graphics has now become a common element in user interfaces, data visualization, television commercial, motion pictures, and many-many other application. With the advantages of Graphics, modern subject, we can be able to design different applications and tools which can be beneficial in our daily life. Besides, some entertaining applications can also be designed using Graphics such as Games, Photo Effects etc. We are trying to apply the Graphics applications on designing Natural Scenery. It will display the scenery of sea.

ACKNOWLEDGEMENT

Many people have contributed to this project in variety of ways. To all of them we would like to express our appreciation. We also acknowledge the many helpful comments from our friends and co-workers on many ideas. We are indebted to all those who provided reviews or suggestion for improving the material covered in this report and we extend our apologies to anyone who may have failed to mention. Firstly we would like to give lots of thanks to the course maker of TANIM AHMED for giving us such a wonderful opportunity. Secondly also to our university Daffodil International University and CSE

Department for giving us this platform and at last but the most important to our course teacher TANIM AHMED for his motivation and encouragement in our project.

INTRODUCTION

Computer has become a powerful tool for the rapid and economical production of pictures. Computer Graphics remains one the most exciting and rapidly growing fields. Old Chinese saying" One picture is worth of thousand words" can be modified in this computer era into "One picture is worth of many kilobytes of data". It is natural to expect that graphical communication will often be more convenient when computers are utilized for this purpose. Many people for different domain of applications use interactive graphics. For example structural engineering use for efficient design of structures on the basis of the analysis of stress in various elements of the structure. From the survey it is evident that in future, engineers, designers etc. will be using computer graphics quite extensively. There is virtually no area in which graphical displays cannot be used to some advantage, and so it is not surprising to find the use of computer graphics so widespread. Today, we find Computer Graphics used routinely in such diverse areas such as science, engineering, medicine, business, industry, government, art, entertainment, advertising, education, training, etc.

So for understanding the depth of this subject and for gaining sound knowledge in this field we had an attempted to the first step on this current field. We tried to make a graphically designated Natural Scenery. The Graphic Designated Natural Scenario is made with the application of Graphics codes. This project applied the subject and made the Natural Scenario.

OBJECTIVES

The main objectives of this project are:

- ❖ To implement the features of graphics.
- ❖ To interface the applications of graphics to the real world.
- ❖ To become familiarization with Graphics and its logical coding.

HEADER FILE USED

1) <math.h>:

Math.h header file (<math.h>) of c++ programming language contains constants and functions to perform mathematical operations. You can use functions of math.h in your c programs to calculating absolute value of a number, calculating logarithms and using trigonometric functions to calculate sine, cosine of an angle.

Math.h functions used in our project:

- Cos
- Sin

2) <GL/gl.h>:

GL/gl. h are the **base OpenGL headers**, which give you OpenGL-1.1 function and token declarations, any maybe more. For anything going beyond version 1.1 you must use the OpenGL extension mechanism.

3) <GL/glut.h>

The OpenGL Utility Toolkit (GLUT) is **a library of utilities for OpenGL programs, which primarily perform system-level I/O with the host operating system**. Functions performed include window definition, window control, and monitoring of keyboard and mouse input.

Project name:

- Beauty of nature.

Tools has been used:

- CodeBlocks, Opengl, Glut.

Language has been used:

- C++

Project github link: **Click here**

Project Code:

```
//
Header
file:


          #include<GL/gl.h>
          #include<GL/glut.h>
          #include<math.h>


          void reshape(int,int);
          void timer(int);


          //user define functions:



          void circle(){
              //sun:
              glColor3f(1.0, 1.0, 0.0);
              glBegin(GL_POLYGON);
              float a;
              for(int i=0;i<360;i++){
                  a=i*3.142/180;
                  glVertex2f(0.40+0.05*cos(a),0.75+0.09*sin(a));
              }
              glEnd();
          }

          void field(){
```

```
    // grass:
    glColor3f(0.0, 1.0, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.00,0.00,0.0);
    glVertex3f(0.67,0.00,0.0);
    glVertex3f(0.66,0.03,0.0);
    glVertex3f(0.65,0.07,0.0);
    glVertex3f(0.62,0.11,0.0);
    glVertex3f(0.58,0.12,0.0);
    glVertex3f(0.55,0.12,0.0);
    glVertex3f(0.52,0.14,0.0);
    glVertex3f(0.50,0.17,0.0);
    glVertex3f(0.45,0.19,0.0);
    glVertex3f(0.43,0.21,0.0);
    glVertex3f(0.40,0.21,0.0);
    glVertex3f(0.35,0.23,0.0);
    glVertex3f(0.20,0.23,0.0);
    glVertex3f(0.15,0.25,0.0);
    glVertex3f(0.00,0.25,0.0);
    glEnd();
}

void home(){
    //first home first wall
    glColor3f(0.5, 0.5, 0.5);
    glBegin(GL_POLYGON);
    glVertex3f(0.12,0.10,0.0);
    glVertex3f(0.20,0.10,0.0);
    glVertex3f(0.20,0.15,0.0);
    glVertex3f(0.12,0.15,0.0);
    glEnd();

    //first home triangle
    glColor3f(0.5, 0.0, 0.5);
    glBegin(GL_TRIANGLES);
    glVertex3f(0.12,0.15,0.0);
    glVertex3f(0.20,0.15,0.0);
    glVertex3f(0.16,0.19,0.0);
    glEnd();

    //first home left roof
    glColor3f(0.5, 0.5, 0.2);
    glBegin(GL_POLYGON);
    glVertex3f(0.10,0.15,0.0);
    glVertex3f(0.12,0.15,0.0);
    glVertex3f(0.16,0.19,0.0);
```

```
glVertex3f(0.15,0.20,0.0);
glEnd();

//first home right roof
glColor3f(0.5, 0.7, 0.5);
glBegin(GL_POLYGON);
glVertex3f(0.20,0.15,0.0);
glVertex3f(0.25,0.15,0.0);
glVertex3f(0.20,0.20,0.0);
glVertex3f(0.15,0.20,0.0);
glEnd();

//first home 2nd wall
glColor3f(0.5, 0.5, 0.0);
glBegin(GL_POLYGON);
glVertex3f(0.20,0.10,0.0);
glVertex3f(0.24,0.10,0.0);
glVertex3f(0.24,0.15,0.0);
glVertex3f(0.20,0.15,0.0);
glEnd();

//first home left wall left window
glColor3f(1.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.13,0.12,0.0);
glVertex3f(0.15,0.12,0.0);
glVertex3f(0.15,0.14,0.0);
glVertex3f(0.13,0.14,0.0);
glEnd();

//first home left wall door
glColor3f(1.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.16,0.10,0.0);
glVertex3f(0.19,0.10,0.0);
glVertex3f(0.19,0.14,0.0);
glVertex3f(0.16,0.14,0.0);
glEnd();

//first home right wall right window
glColor3f(1.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.21,0.12,0.0);
glVertex3f(0.23,0.12,0.0);
glVertex3f(0.23,0.14,0.0);
glVertex3f(0.21,0.14,0.0);
```

```
glEnd();

//first home left under decoration
glColor3f(0.2, 0.3, 0.4);
glBegin(GL_POLYGON);
glVertex3f(0.11,0.09,0.0);
glVertex3f(0.21,0.09,0.0);
glVertex3f(0.20,0.10,0.0);
glVertex3f(0.12,0.10,0.0);
glEnd();



//first home right under decoration
glColor3f(0.2, 0.2, 0.1);
glBegin(GL_POLYGON);
glVertex3f(0.21,0.09,0.0);
glVertex3f(0.25,0.09,0.0);
glVertex3f(0.24,0.10,0.0);
glVertex3f(0.20,0.10,0.0);
glEnd();


//second home first wall
glColor3f(0.5, 0.5, 0.5);
glBegin(GL_POLYGON);
glVertex3f(0.24,0.10,0.0);
glVertex3f(0.30,0.10,0.0);
glVertex3f(0.30,0.15,0.0);
glVertex3f(0.24,0.15,0.0);
glEnd();



//second home second wall
glColor3f(0.5, 0.5, 0.0);
glBegin(GL_POLYGON);
glVertex3f(0.30,0.10,0.0);
glVertex3f(0.34,0.10,0.0);
glVertex3f(0.34,0.15,0.0);
glVertex3f(0.30,0.15,0.0);
glEnd();


//second home right roof
glColor3f(0.5, 0.7, 0.5);
```

```
glBegin(GL_POLYGON);
glVertex3f(0.30,0.15,0.0);
glVertex3f(0.35,0.15,0.0);
glVertex3f(0.30,0.20,0.0);
glVertex3f(0.25,0.20,0.0);
glEnd();


//second home left roof
glColor3f(0.5, 0.5, 0.2);
glBegin(GL_POLYGON);
glVertex3f(0.24,0.16,0.0);
glVertex3f(0.26,0.19,0.0);
glVertex3f(0.25,0.20,0.0);
glVertex3f(0.23,0.17,0.0);
glEnd();


//second home triangle
glColor3f(0.5, 0.0, 0.5);
glBegin(GL_POLYGON);
glVertex3f(0.25,0.15,0.0);
glVertex3f(0.30,0.15,0.0);
glVertex3f(0.26,0.19,0.0);
glVertex3f(0.24,0.16,0.0);
glEnd();


//second home left window
glColor3f(1.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.24,0.12,0.0);
glVertex3f(0.25,0.12,0.0);
glVertex3f(0.25,0.14,0.0);
glVertex3f(0.24,0.14,0.0);
glEnd();


//second home door
glColor3f(1.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.26,0.10,0.0);
glVertex3f(0.29,0.10,0.0);
glVertex3f(0.29,0.14,0.0);
glVertex3f(0.26,0.14,0.0);
glEnd();
```

```
    //second home right window
    glColor3f(1.0, 1.0, 1.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.31,0.12,0.0);
    glVertex3f(0.33,0.12,0.0);
    glVertex3f(0.33,0.14,0.0);
    glVertex3f(0.31,0.14,0.0);
    glEnd();


    //second home left under decoration
    glColor3f(0.2, 0.3, 0.4);
    glBegin(GL_POLYGON);
    glVertex3f(0.25,0.09,0.0);
    glVertex3f(0.31,0.09,0.0);
    glVertex3f(0.30,0.10,0.0);
    glVertex3f(0.24,0.10,0.0);
    glEnd();



    //second home right under decoration
    glColor3f(0.2, 0.2, 0.1);
    glBegin(GL_POLYGON);
    glVertex3f(0.31,0.09,0.0);
    glVertex3f(0.35,0.09,0.0);
    glVertex3f(0.34,0.10,0.0);
    glVertex3f(0.30,0.10,0.0);
    glEnd();
}

void river(){
    //river first portion from left
    glColor3f(0.0, 1.0, 1.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.00,0.25,0.0);
    glVertex3f(0.15,0.25,0.0);
    glVertex3f(0.15,0.45,0.0);
    glVertex3f(0.00,0.45,0.0);
    glEnd();

    //river second portion from left
    glColor3f(0.0, 1.0, 1.0);
    glBegin(GL_POLYGON);
```

```
glVertex3f(0.15,0.25,0.0);
glVertex3f(0.20,0.23,0.0);
glVertex3f(0.20,0.45,0.0);
glVertex3f(0.15,0.45,0.0);
glEnd();

//river 3rd portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.20,0.23,0.0);
glVertex3f(0.35,0.23,0.0);
glVertex3f(0.35,0.45,0.0);
glVertex3f(0.20,0.45,0.0);
glEnd();


//river 4th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.35,0.23,0.0);
glVertex3f(0.40,0.21,0.0);
glVertex3f(0.40,0.45,0.0);
glVertex3f(0.35,0.45,0.0);
glEnd();


//river 5th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.40,0.21,0.0);
glVertex3f(0.43,0.21,0.0);
glVertex3f(0.43,0.45,0.0);
glVertex3f(0.40,0.45,0.0);
glEnd();


//river 6th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.43,0.21,0.0);
glVertex3f(0.45,0.19,0.0);
glVertex3f(0.45,0.45,0.0);
glVertex3f(0.43,0.45,0.0);
glEnd();
```

```
//river 7th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.45,0.19,0.0);
glVertex3f(0.50,0.17,0.0);
glVertex3f(0.50,0.45,0.0);
glVertex3f(0.45,0.45,0.0);
glEnd();


//river 8th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.50,0.17,0.0);
glVertex3f(0.52,0.14,0.0);
glVertex3f(0.52,0.45,0.0);
glVertex3f(0.50,0.45,0.0);
glEnd();


//river 9th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.52,0.14,0.0);
glVertex3f(0.55,0.12,0.0);
glVertex3f(0.55,0.45,0.0);
glVertex3f(0.52,0.45,0.0);
glEnd();


//river 10th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.55,0.12,0.0);
glVertex3f(0.58,0.12,0.0);
glVertex3f(0.58,0.45,0.0);
glVertex3f(0.55,0.45,0.0);
glEnd();


//river 11th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.58,0.12,0.0);
glVertex3f(0.62,0.11,0.0);
glVertex3f(0.62,0.45,0.0);
```

```
glVertex3f(0.58,0.45,0.0);
glEnd();


//river 12th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.62,0.11,0.0);
glVertex3f(0.65,0.07,0.0);
glVertex3f(0.65,0.45,0.0);
glVertex3f(0.62,0.45,0.0);
glEnd();


//river 13th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.65,0.07,0.0);
glVertex3f(0.66,0.03,0.0);
glVertex3f(0.66,0.45,0.0);
glVertex3f(0.65,0.45,0.0);
glEnd();


//river 14th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.66,0.03,0.0);
glVertex3f(0.67,0.00,0.0);
glVertex3f(0.67,0.45,0.0);
glVertex3f(0.66,0.45,0.0);
glEnd();


//river 15th portion from left
glColor3f(0.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f(0.67,0.00,0.0);
glVertex3f(0.80,0.00,0.0);
glVertex3f(0.80,0.45,0.0);
glVertex3f(0.67,0.45,0.0);
glEnd();



//river last portion from left
```

```
    glColor3f(0.0, 1.0, 1.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.80,0.00,0.0);
    glVertex3f(1.00,0.00,0.0);
    glVertex3f(1.00,0.45,0.0);
    glVertex3f(0.67,0.45,0.0);
    glEnd();
}

float a = 0.00, b=0.06, c=0.02, d=0.07 ;

void boat(){


    //boat under decoration
    glColor3f(0.3, 0.1, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(a,0.33,0.0);
    glVertex3f(a+0.03,0.30,0.0);
    glVertex3f(a+0.10,0.30,0.0);
    glVertex3f(a+0.12,0.33,0.0);
    glEnd();


    //boat stick decoration
    glColor3f(0.3, 0.1, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(b,0.33,0.0);
    glVertex3f(b+0.01,0.33,0.0);
    glVertex3f(b+0.01,0.43,0.0);
    glVertex3f(b,0.43,0.0);
    glEnd();


    //boat left sail decoration
    glColor3f(1.0, 1.0, 1.0);
    glBegin(GL_TRIANGLES);
    glVertex3f(c,0.34,0.0);
    glVertex3f(c+0.04,0.34,0.0);
    glVertex3f(c+0.04,0.41,0.0);
    glEnd();


    //boat right sail decoration
    glColor3f(8.0, 0.0, 0.0);
    glBegin(GL_TRIANGLES);
```

```
    glVertex3f(d,0.34,0.0);
    glVertex3f(d+0.03,0.34,0.0);
    glVertex3f(d,0.42,0.0);
    glEnd();

}

void hill(){
    //1st hill decoration
    glColor3f(0.3, 0.1, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex3f(0.00,0.45,0.0);
    glVertex3f(0.20,0.45,0.0);
    glVertex3f(0.10,0.60,0.0);
    glEnd();


    //2nd hill decoration
    glColor3f(0.4, 0.2, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.20,0.45,0.0);
    glVertex3f(0.27,0.45,0.0);
    glVertex3f(0.20,0.59,0.0);
    glVertex3f(0.16,0.51,0.0);
    glEnd();


    //3rd hill decoration
    glColor3f(0.3, 0.1, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.27,0.45,0.0);
    glVertex3f(0.35,0.45,0.0);
    glVertex3f(0.28,0.62,0.0);
    glVertex3f(0.23,0.53,0.0);
    glEnd();


    //4th hill decoration
    glColor3f(0.4, 0.2, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.35,0.45,0.0);
    glVertex3f(0.45,0.45,0.0);
    glVertex3f(0.38,0.57,0.0);
    glVertex3f(0.33,0.50,0.0);
    glEnd();
```

```
    //5th hill decoration
    glColor3f(0.3, 0.1, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.45,0.45,0.0);
    glVertex3f(0.59,0.45,0.0);
    glVertex3f(0.48,0.60,0.0);
    glVertex3f(0.42,0.50,0.0);
    glEnd();


    //6th hill decoration
    glColor3f(0.4, 0.2, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.59,0.45,0.0);
    glVertex3f(0.74,0.45,0.0);
    glVertex3f(0.60,0.63,0.0);
    glVertex3f(0.53,0.53,0.0);
    glEnd();


    //7th hill decoration
    glColor3f(0.3, 0.1, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f(0.74,0.45,0.0);
    glVertex3f(0.80,0.45,0.0);
    glVertex3f(0.75,0.54,0.0);
    glVertex3f(0.71,0.49,0.0);
    glEnd();
}


void cloud(){
    //Cloud 1st portion
    glColor3f(0.961, 0.961, 0.961);
    glBegin(GL_POLYGON);
    glVertex3f(0.00,0.45,0.0);
    glVertex3f(0.10,0.60,0.0);
    glVertex3f(0.10,1.00,0.0);
    glVertex3f(0.00,1.00,0.0);
    glEnd();


    //Cloud 2nd portion
    glColor3f(0.961, 0.961, 0.961);
    glBegin(GL_POLYGON);
```

```
glVertex3f(0.10,0.60,0.0);
glVertex3f(0.16,0.51,0.0);
glVertex3f(0.16,1.00,0.0);
glVertex3f(0.10,1.00,0.0);
glEnd();


//Cloud 3rd portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.16,0.51,0.0);
glVertex3f(0.20,0.59,0.0);
glVertex3f(0.20,1.00,0.0);
glVertex3f(0.16,1.00,0.0);
glEnd();


//Cloud 4th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.20,0.59,0.0);
glVertex3f(0.23,0.53,0.0);
glVertex3f(0.23,1.00,0.0);
glVertex3f(0.20,1.00,0.0);
glEnd();



//Cloud 5th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.23,0.53,0.0);
glVertex3f(0.28,0.62,0.0);
glVertex3f(0.28,1.00,0.0);
glVertex3f(0.23,1.00,0.0);
glEnd();



//Cloud 6th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.27,0.62,0.0);
glVertex3f(0.33,0.50,0.0);
glVertex3f(0.33,1.00,0.0);
glVertex3f(0.27,1.00,0.0);
```

```
glEnd();



//Cloud 7th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.33,0.50,0.0);
glVertex3f(0.38,0.57,0.0);
glVertex3f(0.38,1.00,0.0);
glVertex3f(0.33,1.00,0.0);
glEnd();



//Cloud 8th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.38,0.57,0.0);
glVertex3f(0.42,0.50,0.0);
glVertex3f(0.42,1.00,0.0);
glVertex3f(0.38,1.00,0.0);
glEnd();


//Cloud 9th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.42,0.50,0.0);
glVertex3f(0.48,0.60,0.0);
glVertex3f(0.48,1.00,0.0);
glVertex3f(0.42,1.00,0.0);
glEnd();



//Cloud 10th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.48,0.60,0.0);
glVertex3f(0.53,0.53,0.0);
glVertex3f(0.53,1.00,0.0);
glVertex3f(0.48,1.00,0.0);
glEnd();
```

```
//Cloud 11th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.53,0.53,0.0);
glVertex3f(0.60,0.63,0.0);
glVertex3f(0.60,1.00,0.0);
glVertex3f(0.53,1.00,0.0);
glEnd();



//Cloud 12th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.60,0.63,0.0);
glVertex3f(0.71,0.49,0.0);
glVertex3f(0.71,1.00,0.0);
glVertex3f(0.60,1.00,0.0);
glEnd();



//Cloud 13th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.71,0.49,0.0);
glVertex3f(0.75,0.54,0.0);
glVertex3f(0.75,1.00,0.0);
glVertex3f(0.71,1.00,0.0);
glEnd();



//Cloud 14th portion
glColor3f(0.961, 0.961, 0.961);
glBegin(GL_POLYGON);
glVertex3f(0.75,0.54,0.0);
glVertex3f(0.80,0.45,0.0);
glVertex3f(0.80,1.00,0.0);
glVertex3f(0.75,1.00,0.0);
glEnd();



//Cloud 15th portion
```

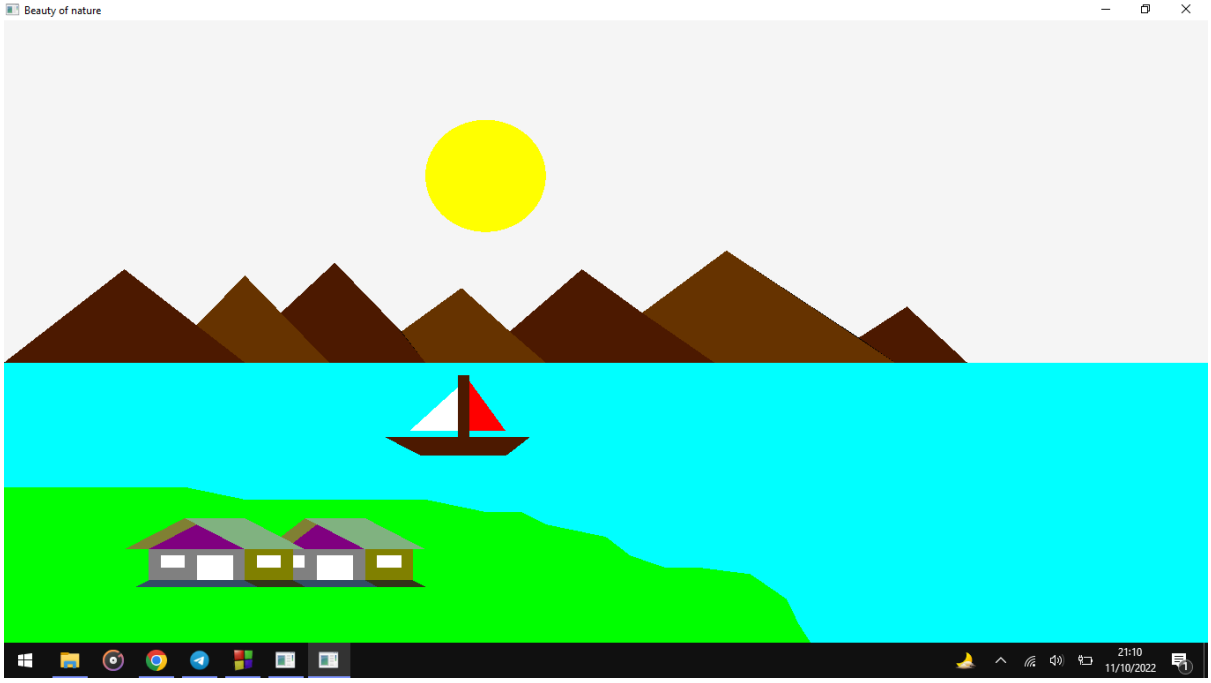
```
    glColor3f(0.961, 0.961, 0.961);
    glBegin(GL_POLYGON);
    glVertex3f(0.80,0.45,0.0);
    glVertex3f(1.00,0.45,0.0);
    glVertex3f(1.00,1.00,0.0);
    glVertex3f(0.80,1.00,0.0);
    glEnd();
}

// Display function:

void display(void){
    glClear(GL_COLOR_BUFFER_BIT);
    glLoadIdentity();
    field();
    home();
    river();
    boat();
    hill();
    cloud();
    circle();
    glutSwapBuffers();
}

    // Initialize function:

void init(void){
    glClearColor(0.0,0.0,0.0,0.0);  // fix windows background color
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0.0,1.0,0.0,1.0, 0.0,1.0);
}

// Main function:

int main(int argc, char** argv)
{
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
    glutInitWindowSize(1400,700);
    glutInitWindowPosition(0,0);
    glutCreateWindow("Beauty of nature");
    init();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutTimerFunc(1000,timer,0);
```

```
    glutMainLoop();
    return 0;
}




void reshape(int w,int h)
{
    glViewport(0,0,(GLsizei)w,(GLsizei)h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0.0,1.0,0.0,1.0);
    glMatrixMode(GL_MODELVIEW);
}


int state = 1;

void timer(int)
{
    glutPostRedisplay();        //opengl call the display function the next
time it gets the changes.
    glutTimerFunc(1000/700,timer,0);   //timer function calling itself 1000/700
times in 1 second.

    switch(state)
    {
    case 1:
        if(a<1.10 && b<1.10 && c<1.10 && d<1.10){
            a+=0.0001;
            b+=0.0001;
            c+=0.0001;
            d+=0.0001;
        }
    else
        state = -1;
        break;
    case -1:
        if(a>-0.15 && b>-0.15 && c>-0.15 && d>-0.15){
            a-=0.0001;
            b-=0.0001;
            c-=0.0001;
            d-=0.0001;
        }
        else
```

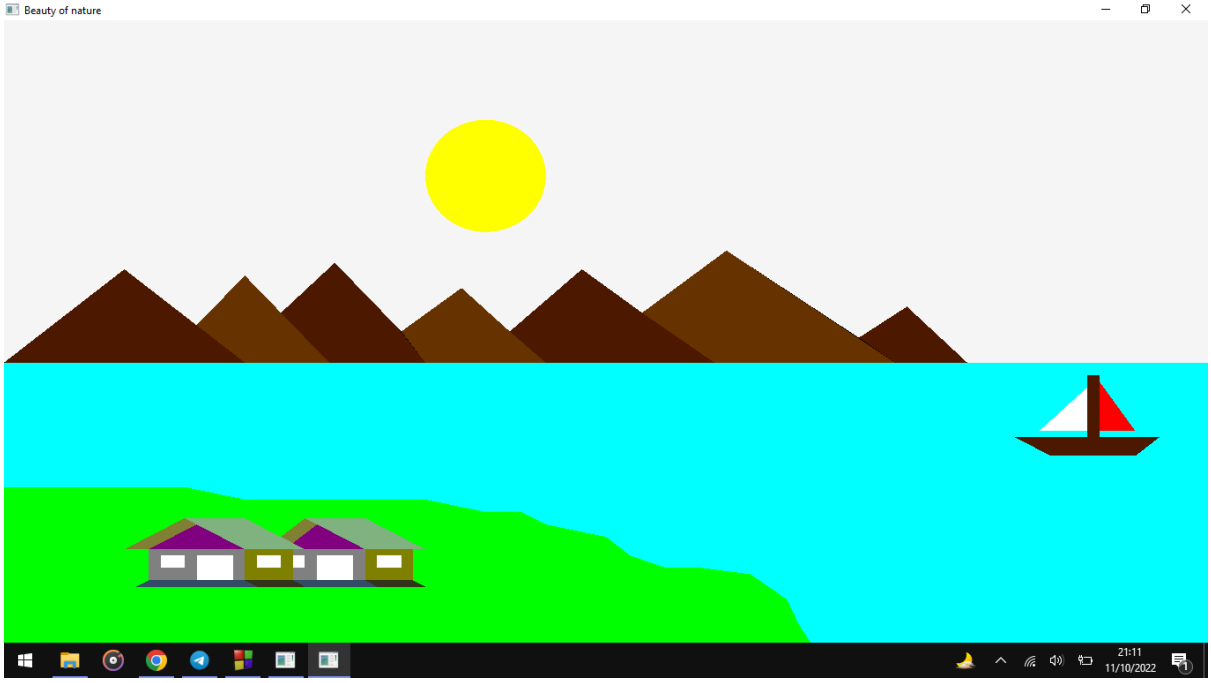
```
            state = 1;
        break;
}}
```

Project Output


Beauty of nature
21:10
11/10/2022


Beauty of nature
21:10
11/10/2022


Beauty of nature
21:11
11/10/2022


Beauty of nature
21:11
11/10/2022