# Computer Graphics (UCS505)

# Project on

# Demonstration of Land Breeze and Sea Breeze in Day and Night

**Submitted By**

Arjun Bhateja 102003042

**B.E. Third Year – COE-2**

**Submitted To:**

**Dr. Harpreet Singh**

**Assistant Professor**

**CSED TIET, Patiala**



**Computer Science and Engineering Department Thapar Institute of Engineering and Technology Patiala – 147001**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Description** | **Page No.** |
| 1. | Introduction to Project |  |
| 2. | Computer Graphics concepts used |  |
| 3. | User Defined Functions |  |
| 4. | Code |  |
| 5. | Output/ Screen shots |  |

**Introduction:**

The program creates an interactive visual display of land breeze and sea breeze during the day and night, with various elements in the scene moving and changing in response to user input.

The program responds to various keyboard inputs, such as 's' to start the program, 'b' to start the boat animation, 'e' to stop the boat animation, 'h' to start the horse animation, 'f' to stop the horse animation, and 'd' to set the scene to daytime. It also initializes several color variables to represent various elements in the scene, such as the sky, sun, water, land, and trees.

The main function initializes OpenGL and sets up the display mode, window size, and other settings. It then calls the glutMainLoop() function, which enters the main event processing loop and calls various functions to update the display and respond to user input.

**Computer Graphics concepts used:**

* OpenGL: The code uses the OpenGL graphics API to create the 2D and 3D graphics. Transformation: The code applies various transformations such as rotation and translation to the objects in the scene to create the animation.
* Modeling: The code models the objects in the scene using primitives such as rectangles (obstacles).
* Color: The code sets the color of the objects in the scene using RGB values.
* Rendering: The code renders the graphics on the screen using the display function and the glutSwapBuffers function.
* Animation: The code animates the objects in the scene by updating their position.

**User Defined Functions:**

The program includes several functions that are called throughout the main function to draw various elements in the scene. The welcome() and instructions() functions display introductory messages and user instructions, respectively. The draw\_Boat() function draws a boat on the water and moves it back and forth if the boatStatus flag is set to 1. The draw\_tree() and draw\_house() functions draw trees and houses on the land, respectively. The draw\_scene() function draws the overall scene, including the sky, sun, water, land, and various elements on the land. The draw\_horse() function draws a horse on the land and moves it back and forth if the horseStatus flag is set to 1. The draw\_dArrow() and draw\_uArrow() functions draw down and up arrows, respectively, to indicate the direction of the breezes. The plot() function plots a point on the screen, which is used in the midPointCircleAlgo() function to draw a circle. This function is called to draw a circle in the scene.

**Code:**

#include <windows.h>

#include <stdio.h>

#include <iostream>

#include <GL/glut.h>

using namespace std;

//variable declarations

int boatStatus = 0, horseStatus = 0;

int sprgm = 0; //start program flag

int ua1 = 1000, ua2 = 900, da1 = 200, da2 = 100;//x values for up and down arrows

int lefY = 170, rigY = 120; //y values for left and right labels

//initial color values during day

float R = 1, G = 1, B = 1; //sky gradient : white

float r = 1, g = 1, b = 0; //sky color : yellow

float r1 = 1.0, g1 = 0.5, b1 = 0.0; //celestial obj : orange

float r2 = 7.0, g2 = 0.0, b2 = 0.0; //down arrow : red

float r3 = 0.0, g3 = 0.0, b3 = 1.0; //up arrow : blue

float r4 = 0.2, g4 = 0.5, b4 = 0.8; //river : light blue

float r5 = 0.0, g5 = 0.7, b5 = 0.0; //land : bright green

float r6 = 0.0, g6 = 0.6, b6 = 0.0; //tree : light green

float r7 = 0.0, g7 = 0.0, b7 = 0.0; //instruction : black

float boatX = 0;

float boatY = 0;

float horseX = 0;

float horseY = 0;

float dArrowX = 0;

float dArrowY = 0;

float uArrowX = 0;

float uArrowY = 0;

int pntX1 = 1100, pntY1 = 650, radius = 40; //circle variables

//text variables

int i, s, m, y; //i is loop variable, s is space, m in index var for msg[], y axis value

string msg[6] = { "COMPUTER GRAPHICS PROJECT","TOPIC: DEMONSTRATION OF LAND & SEA BREEZE"," DURING DAY AND NIGHT","C++","OPENGL/GLUT","PRESS [S] TO START" };

string ins[6] = { "D: DAY","N: NIGHT","H: START HORSE","F: STOP","B: START BOAT","E: STOP" };

string t, lef = "WARM LAND BREEZE", rig = "COOL SEA BREEZE"; //temp string, left label, right label

//function declarations

void welcome();

void instructions();

void draw\_Boat();

void draw\_tree(int, int);

void draw\_house(int, int);

void draw\_scene();

void draw\_horse();

void draw\_dArrow(int, int);

void draw\_uArrow(int, int);

void plot(int, int);

void midPointCircleAlgo();

//keyboard entries and the corresponding change in value of various variables

void keyboard(unsigned char key, int x, int y)

{

switch (key) {

case 's':

case 'S':

sprgm = 1;

break;

case 'b': //boat start

case 'B':

boatStatus = 1;

break;

case 'e': //boat stop

case 'E':

boatStatus = 0;

break;

case 'H': //horse start

case 'h':

horseStatus = 1;

break;

case 'F': //horse stop

case 'f':

horseStatus = 0;

break;

case 'D': //day

case 'd':

{

//sky

R = 1, G = 1, B = 1;

r = 1, g = 1, b = 0;

//sun

r1 = 1, g1 = 0.5, b1 = 0.0;

//landbreeze down arrow: red

r2 = 7.0, g2 = 0.0, b2 = 0.0;

//seabreeze up arrow: blue

r3 = 0.0, g3 = 0.0, b3 = 7.0;

//river

r4 = 0.2, g4 = 0.5, b4 = 0.8;

//land

r5 = 0.0, g5 = 0.7, b5 = 0.0;

//tree

r6 = 0.0, g6 = 0.6, b6 = 0.0;

//ins

r7 = 0.0, g7 = 0.0, b7 = 0.0;

lef = "WARM LAND BREEZE";

rig = "COOL SEA BREEZE";

ua1 = 1000, ua2 = 900, da1 = 200, da2 = 100;

lefY = 170, rigY = 120;

break;

}

case 'N': //night

case 'n':

{

//sky

R = 0, G = 0, B = 0.5;

r = 0, g = 0, b = 0;

//moon

r1 = 1, g1 = 1, b1 = 1;

//seabreeze down arrow: blue

r2 = 0.0, g2 = 0.0, b2 = 7.0;

//landbreeze up arrow: red

r3 = 7.0, g3 = 0.0, b3 = 0.0;

//river

r4 = 0.1, g4 = 0.2, b4 = 0.3;

//land

r5 = 0.2, g5 = 0.5, b5 = 0.1;

//tree

r6 = 0.2, g6 = 0.4, b6 = 0.1;

//ins

r7 = 1.0, g7 = 1.0, b7 = 1.0;

lef = "WARM SEA BREEZE";

rig = "COOL LAND BREEZE";

ua1 = 100, ua2 = 200, da1 = 900, da2 = 1000;

lefY = 120, rigY = 170;

break;

}

default:

break;

}

}

//welcome screen

void welcomeBG()

{

glBegin(GL\_POLYGON);

glColor3f(.1, .2, .3);

glVertex2i(0, 600);

glVertex2i(1200, 600);

glVertex2i(1200, 750);

glVertex2i(0, 750);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.8, 0.8, 0.8);

glVertex2i(0, 50);

glVertex2i(0, 600);

glVertex2i(1200, 600);

glVertex2i(1200, 50);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(1, 1, 0.5);

glVertex2i(0, 0);

glVertex2i(0, 50);

glVertex2i(1200, 50);

glVertex2i(1200, 0);

glEnd();

}

void welcome()

{

s = 25;

t = msg[0];

for (i = 0; i < t.length(); i++)

{

glColor3f(1, 1, 1);

glRasterPos2i(i \* s + 270, 660);

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24, t.at(i));

}

m = 1;

y = 500;

while (m < 3)

{

t = msg[m];

for (i = 0; i < t.length(); i++)

{

glColor3f(0, 0, 0);

glRasterPos2i(i \* s + 150, y);

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24, t.at(i));

}

m++;

y = y - 60;

}

y = 200;

s = 22;

while (m < 5)

{

t = msg[m];

for (i = 0; i < t.length(); i++)

{

glColor3f(.1, 0.2, 0.3);

glRasterPos2i(i \* s + 330, y);

glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_18, t.at(i));

}

m++;

y = y - 50;

}

t = msg[5];

for (i = 0; i < t.length(); i++)

{

glColor3f(0, 0, 0);

glRasterPos2i(i \* s + 390, 15);

glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_12, t.at(i));

}

}

//displays instructions and labels on screen

void instructions()

{

m = 0;

y = 700;

s = 15;

while (m < 6)

{

t = ins[m];

for (i = 0; i < t.length(); i++)

{

glColor3f(r7, g7, b7);

glRasterPos2i((i \* s) + 20, y);

glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_12, t.at(i));

}

m++;

y = y - 20;

}

//left label

for (i = 0; i < lef.length(); i++)

{

glColor3f(r7, g7, b7);

glRasterPos2i((i \* s) + 50, lefY);

glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_18, lef.at(i));

}

//right label

for (i = 0; i < rig.length(); i++)

{

glColor3f(r7, g7, b7);

glRasterPos2i((i \* s) + 850, rigY);

glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_18, rig.at(i));

}

}

//

//drawing circle using MidPoint Circle Algorithm

void plot(int x, int y)

{

//plot point

glColor3f(r1, g1, b1);

glPointSize(2.0);

glBegin(GL\_POINTS);

glVertex2i(x + pntX1, y + pntY1);

glEnd();

//draw a line from centre to plotted point

glColor3f(r1, g1, b1);

glLineWidth(2);

glBegin(GL\_LINES);

glVertex2i(pntX1, pntY1);

glVertex2i(x + pntX1, y + pntY1);

glEnd();

}

void midPointCircleAlgo()

{

int x = 0;

int y = radius;

float decision = 5 / 4 - radius;

plot(x, y);

while (y > x)

{

if (decision < 0)

{

x++;

decision += 2 \* x + 1;

}

else

{

y--;

x++;

decision += 2 \* (x - y) + 1;

}

plot(x, y);

plot(x, -y);

plot(-x, y);

plot(-x, -y);

plot(y, x);

plot(-y, x);

plot(y, -x);

plot(-y, -x);

}

}

//

//

//methods defining coordinate values for objects

//

//

void draw\_scene()

{

glBegin(GL\_POLYGON); //Lower Sky

glColor3f(r, g, b);

glVertex2i(0, 600);

glVertex2i(1200, 600);

glColor3f(R, G, B);

glVertex2i(1200, 50);

glVertex2i(0, 50);

glEnd();

glBegin(GL\_POLYGON); //Upper Sky

glColor3f(r, g, b);

glVertex2i(0, 600);

glVertex2i(1200, 600);

glVertex2i(1200, 750);

glVertex2i(0, 750);

glEnd();

glBegin(GL\_POLYGON); //Land

glColor3f(r5, g5, b5);

glVertex2i(0, 100);

glVertex2i(0, 280);

glVertex2i(200, 330);

glVertex2i(400, 360);

glVertex2i(600, 330);

glVertex2i(800, 320);

glVertex2i(1000, 300);

glColor3f(0.7, 0.2, 0.0);//Sand on ground

glVertex2i(1200, 270);

glVertex2i(1200, 100);

glEnd();

glBegin(GL\_POLYGON); // River

glColor3f(r4, g4, b4);

glVertex2i(0, 150);

glVertex2i(1200, 150);

glVertex2i(1200, 0);

glVertex2i(0, 0);

glEnd();

}

void draw\_tree(int x, int y)

{

glBegin(GL\_POLYGON); //rectangular trunk

glColor3f(0.3, 0.2, 0.1);

glVertex2i(0 + x, 0 + y);

glVertex2i(0 + x, 120 + y);

glVertex2i(20 + x, 120 + y);

glVertex2i(20 + x, 0 + y);

glEnd();

glBegin(GL\_POLYGON); //triangle lower

glColor3f(r6, g6, b6);

glVertex2i(x - 60, 120 + y);

glVertex2i(x + 80, 120 + y);

glVertex2i(x + 10, 230 + y);

glEnd();

glBegin(GL\_POLYGON); //triangle middle

glColor3f(r6, g6, b6);

glVertex2i(x - 40, 190 + y);

glVertex2i(x + 60, 190 + y);

glVertex2i(x + 10, 300 + y);

glEnd();

glBegin(GL\_POLYGON); //triangle upper

glColor3f(r6, g6, b6);

glVertex2i(x - 20, 260 + y);

glVertex2i(x + 40, 260 + y);

glVertex2i(x + 10, 360 + y);

glEnd();

}

void draw\_house(int x, int y, float red, float green, float blue)

{

//red is intensity of red color

glBegin(GL\_POLYGON); //House

glColor3f(red, green, blue);

glVertex2i(0 + x, 0 + y);

glVertex2i(120 + x, 0 + y);

glVertex2i(120 + x, 116 + y);

glVertex2i(0 + x, 116 + y);

glEnd();

glBegin(GL\_POLYGON); //Roof

glColor3f(0.9, 0, 0);

glVertex2i(x - 10, 116 + y);

glVertex2i(x + 130, 116 + y);

glVertex2i(x + 100, 156 + y);

glVertex2i(x + 20, 156 + y);

glEnd();

glBegin(GL\_POLYGON); //Door

glColor3f(1, 0.7, 0.2);

glVertex2i(x + 40, y + 0);

glVertex2i(x + 80, y + 0);

glVertex2i(x + 80, y + 75);

glVertex2i(x + 40, y + 75);

glEnd();

}

void draw\_dArrow(int x, int y)

{

glBegin(GL\_POLYGON); //breeze

glColor3f(r2, g2, b2);

glVertex2i(0 + x, 50 + y);

glVertex2i(20 + x, 50 + y);

glVertex2i(20 + x, 150 + y);

glVertex2i(0 + x, 150 + y);

glEnd();

glBegin(GL\_POLYGON); //pointer

glColor3f(r2, g2, b2);

glVertex2i(x - 20, y + 50);

glVertex2i(x + 40, y + 50);

glVertex2i(x + 10, y + 0);

glEnd();

}

void draw\_uArrow(int x, int y)

{

glBegin(GL\_POLYGON); //breeze

glColor3f(r3, g3, b3);

glVertex2i(0 + x, 0 + y);

glVertex2i(20 + x, 0 + y);

glVertex2i(20 + x, 100 + y);

glVertex2i(0 + x, 100 + y);

glEnd();

glBegin(GL\_POLYGON); //pointer

glColor3f(r3, g3, b3);

glVertex2i(x - 20, y + 100);

glVertex2i(x + 40, y + 100);

glVertex2i(x + 10, y + 150);

glEnd();

}

void draw\_Boat()

{

glBegin(GL\_POLYGON); //boat

glColor3f(0.9, 0.5, 0.1);

glVertex2i(250, 20);

glVertex2i(220, 100);

glVertex2i(550, 100);

glVertex2i(500, 20);

glEnd();

glBegin(GL\_POLYGON); //man body

glColor3f(1, 1, 1);

glVertex2i(270, 100);

glVertex2i(330, 100);

glVertex2i(310, 150);

glVertex2i(290, 150);

glVertex2i(310, 200);

glVertex2i(270, 200);

glVertex2i(270, 100);

glEnd();

glBegin(GL\_POLYGON); //face

glColor3f(1.2, 0.5, 0.4);

glVertex2i(275, 200);

glVertex2i(295, 200);

glVertex2i(295, 220);

glVertex2i(300, 220);

glVertex2i(295, 250);

glVertex2i(275, 250);

glVertex2i(275, 200);

glEnd();

glBegin(GL\_POLYGON); //hair

glColor3f(0.0, 0.0, 0.0);

glVertex2i(295, 250);

glVertex2i(300, 270);

glVertex2i(275, 260);

glVertex2i(275, 250);

glVertex2i(295, 250);

glEnd();

glBegin(GL\_POLYGON); //oar

glColor3f(0.0, 0.0, 0.0);

glVertex2i(310, 150);

glVertex2i(250, 0);

glVertex2i(270, 0);

glVertex2i(310, 160);

glVertex2i(310, 150);

glEnd();

}

void draw\_horse()

{

int x = 400;

//front leg

glBegin(GL\_POLYGON);

glColor3f(1, 1, 1);

glVertex2i(100 + x, 200);

glVertex2i(110 + x, 200);

glVertex2i(110 + x, 250);

glVertex2i(100 + x, 250);

glEnd();

//back leg

glBegin(GL\_POLYGON);

glColor3f(1, 1, 1);

glVertex2i(190 + x, 200);

glVertex2i(180 + x, 200);

glVertex2i(180 + x, 250);

glVertex2i(190 + x, 250);

glEnd();

//body

glBegin(GL\_POLYGON);

glColor3f(0.9, 0.3, 0.1);

glVertex2i(100 + x, 250);

glVertex2i(190 + x, 250);

glVertex2i(190 + x, 300);

glVertex2i(100 + x, 300);

glEnd();

//neck and face

glBegin(GL\_POLYGON);

glColor3f(0.9, 0.3, 0.1);

glVertex2i(120 + x, 300);

glVertex2i(90 + x, 390);

glVertex2i(60 + x, 330);

glVertex2i(60 + x, 350);

glVertex2i(80 + x, 350);

glVertex2i(100 + x, 300);

glEnd();

//mane

glBegin(GL\_POLYGON);

glColor3f(1, 1, 1);

glVertex2i(150 + x, 300);

glVertex2i(90 + x, 390);

glVertex2i(120 + x, 300);

glVertex2i(150 + x, 300);

glEnd();

//muscle

glBegin(GL\_POLYGON);

glColor3f(0, 0, 0);

glVertex2i(70 + x, 325);

glVertex2i(70 + x, 351);

glVertex2i(72 + x, 352);

glVertex2i(72 + x, 324);

glEnd();

//tail

glBegin(GL\_POLYGON);

glColor3f(1, 1, 1);

glVertex2i(190 + x, 300);

glVertex2i(190 + x, 300);

glVertex2i(200 + x, 220);

glVertex2i(210 + x, 300);

glEnd();

//eye

glPointSize(4);

glBegin(GL\_POINTS);

glColor3f(0, 0, 0);

glVertex2i(85 + x, 360);

glEnd();

}

//

//

//methods defining motion of objects

//

//

//movement of down arrow (i.e. land breeze)

void dArrow()

{

dArrowY -= .5;

if (dArrowY < 0)

{

dArrowY = +200;

}

glPushMatrix();

glTranslatef(dArrowX, dArrowY, 0);

draw\_dArrow(da1, 0);

draw\_dArrow(da2, 0);

glPopMatrix();

}

//movement of up arrow (i.e. sea breeze)

void uArrow()

{

uArrowY += .5;

if (uArrowY > 200)

{

uArrowY = -200;

}

glPushMatrix();

glTranslatef(uArrowX, uArrowY, 0);

draw\_uArrow(ua1, 0);

draw\_uArrow(ua2, 0);

glPopMatrix();

}

//movement of boat

void boat()

{

if (boatStatus == 1)

{

boatX += .5;

}

if (boatX > 1000)

{

boatX = -600;

}

glPushMatrix();

glTranslatef(boatX, boatY, 0);

draw\_Boat();

glPopMatrix();

}

//movement of horse

void horse()

{

if (horseStatus == 1)

{

horseX -= .3;

}

if (horseX < -600)

{

horseX = +600;

}

glPushMatrix();

glTranslatef(horseX, horseY, 0);

draw\_horse();

glPopMatrix();

}

//

//

//

//

void init(void)

{

glClearColor(0.0, 0.0, 1.0, 0.0);

glColor3f(1.0, 1.0, 1.0);

glMatrixMode(GL\_PROJECTION);

gluOrtho2D(0.0, 1200.0, 0.0, 750.0);

}

void myDisplay(void)

{

if (sprgm == 1)

{

draw\_scene();

draw\_tree(50, 200);

draw\_tree(550, 320);

draw\_tree(850, 310);

draw\_house(320, 340, 0.5, 0.0, 0.5);

draw\_house(700, 310, 0.8, 0.3, 0.5);

midPointCircleAlgo();

horse();

dArrow();

uArrow();

boat();

instructions();

glFlush();

glutPostRedisplay();

}

else

{

welcomeBG();

welcome();

glFlush();

glutPostRedisplay();

}

}

void main(int argc, char\*\* argv)

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(1150, 600);

glutInitWindowPosition(0, 0);

glutCreateWindow("Sea Breeze & Land Breeze");

glutKeyboardFunc(keyboard);

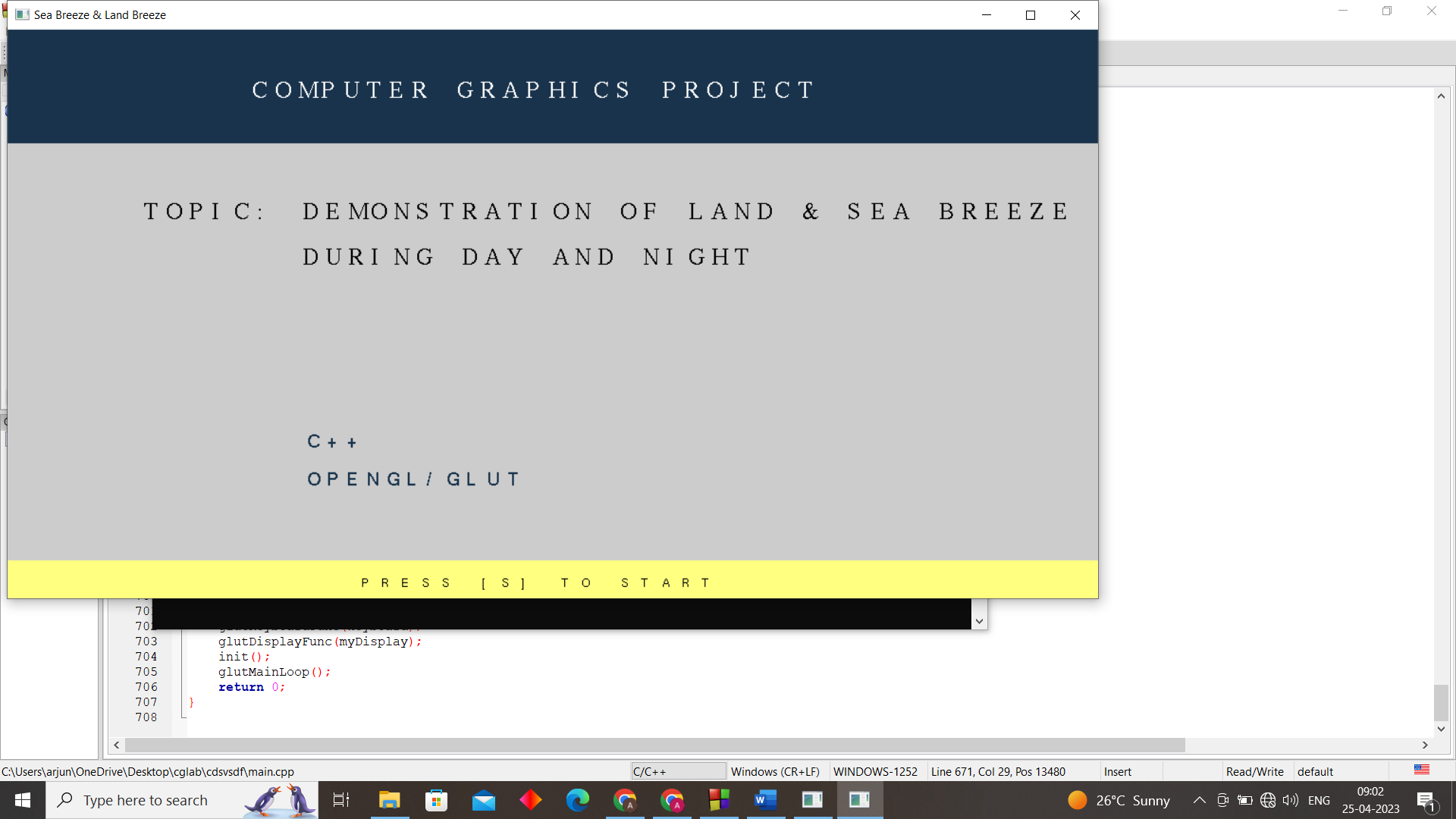
glutDisplayFunc(myDisplay);

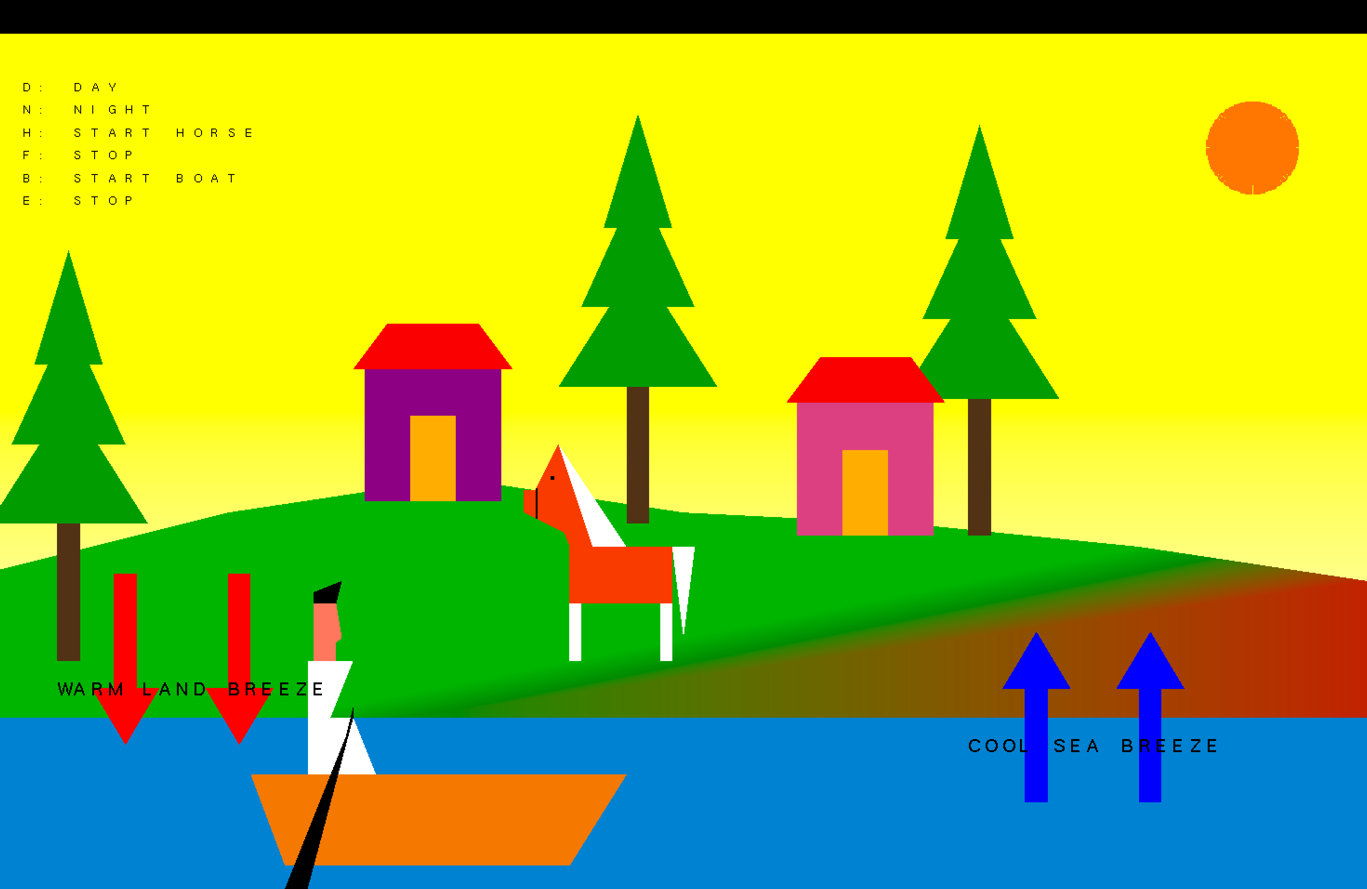
init();

glutMainLoop();

}

**Output Screenshots:**

****

****

****

****