**COMPUTER GRAPHICS (UCS505)**

**Project Report**

**On**

**Train Station Simulator**

**Submitted By:**

|  |  |
| --- | --- |
| Shashwat Singh | 101803385 |
| Hrithik Raina | 101803388 |
|  |  |

**COE -18**

**B.E. Third Year - COE**

**Submitted To:**

Dr. Shatrughan Modi



**Computer Science and Engineering Department**

**Thapar Institute Of Engineering and Technology**

**Patiala-147001**

**June 2021**

**Table of Contents**

S. No Title Page number

1. Introduction of project 2

2. Concepts used 3

3. Functions used 4

4. Code 5

5. Snapshots 20

**INTRODUCTION:**

Computer Graphics is based on technology to access. The process transforms and presents information in a visual form. So g[raphics](https://www.geeksforgeeks.org/introduction-to-computer-graphics/geeksforgeeks.org/computer-graphics-2/) is defined as any sketch or a drawing or a special network that pictorially represents some meaningful information. In today's life, computer graphics has now become a common element in user interfaces, T.V. commercial motion pictures. Computer Graphics is the creation of pictures with the help of a computer. The end product of computer graphics is a picture; it may be a business graph, drawing, and engineering.

In computer graphics, two or three-dimensional pictures can be created that are used for research. Many hardware devices’ algorithms have been developed for improving the speed of picture generation with the passage of time. It includes the creation, storage of models and images of objects. These models have been used in various fields like engineering, mathematics and so on.

Today computer graphics have completely transformed from its initial version. Graphics provides one of the most natural means of communicating with a computer since our highly developed 2D or 3D pattern-recognition abilities allow us to perceive and process pictorial data rapidly. Computers have become a powerful medium for the rapid and economical production of pictures. It is one of the most interactive ways in which a user can control or use a system.

The idea behind this project is to display the working of the train station using computer graphics. This project demonstrates the scene in which one train and its bogies are moving from left to right, according to the traffic signal. This project consists of the station, the sky, the sun and moon, a comet, the traffic signal, the platform, a bench, trees and the grass.

After pressing any key on the prompt, which has detailed instructions on how to run our graphical interface, we enter the demo. We start the program by pressing the ‘t’ button and the train will start to move, until we press the ‘r’ button, for red light, which stops the train at the traffic light. Now we can press the ‘g’ button, for green light and the train starts to move again. If we press the red signal key before the train has crossed the station it will stop at the end of the window and if we press the red signal button key after the train has passed the signal it stops immediately.

**CONCEPT:**

Computer graphics is one of the most exciting and rapidly growing computer fields. It is also an extremely effective medium for communication between man and computer; a human being can understand the information content of a displayed diagram or perspective view much faster than he can understand a table of numbers or text containing the same information.

OpenGL (open graphics library) is a standard specification defining a cross-language across platform API for writing applications that produce 2D and 3D computer graphics.

OpenGL is an application program interface (API) offering various functions to implement primitives, models and images. This offers functions to create and manipulate render lighting, colouring, viewing the models. OpenGL offers different coordinate systems and frames. OpenGL offers translation, rotation and scaling of objects.

The main aim of this mini project is to illustrate the concepts and usage of pre-built functions in OpenGL. It simulates two trains where two tracks are not connected but trains work simultaneously with constant speed.

The best method for building the objects is by placing the drawing code for each object in a separate function. Each object is defined using a coordinate system that makes modelling convenient.

After we finish the drawings, our next aim would be to give motion to the train. As we are going to draw a simple train so our track will be straight. Train engines and bogies are made of many rectangles while wheels are circular. The trainhas a bogie and an engine. We have defined the main function in such a way that the train works according to the traffic signal.

We have represented the stoppage of the train by using traffic lights. Press 'r' or 'R' to change the signal light to red. Press 'g' or 'G' to change the signal light to green. Press 'd' or 'D' to make it day. Press 'n' or 'N' to make it night. Press 't' or 'T' Train to arrive at the station. Press the RIGHT MOUSE BUTTON to display the menu. Press LEFT MOUSE BUTTON to quit the program. We have used the keyboard to interact with the program.

**FUNCTIONS USED:**

* **draw\_pixel()** and **plotpixels()-** These are functions called by draw\_circle and are necessary to plot a circle, using the midpoint method
* **draw\_circle()**- This function has the program to draw a circle. Circles can be created using small sets of lines. We used a bunch of lines, which are closed together to go round forming a circle. For drawing circles in opengl, this method also used the simple sine and cosine maths functions, but we have not used pie instead go for the degree for angles.
* **train\_station()-** This Function contains all the visual elements of the code. the station , platform and the train itself along with its buggies, the basic frame of the traffic signal/stop light and the various visual elements provided like the trees and the bench.

* **traffic\_light()-** This is responsible for all of the actions performed by the traffic signal. This takes 2 inputs “r” for red light and ”g” for green light, which in turn change the value of the ’light’ variable to 1 or 0, and lets us change the color shown on the traffic lights.
* **idle()-** One of the main functions responsible for all the motion in the scene. This function has various if-else statements that change the value of the position variables as the main loops, or we interrupt with the aforementioned interrupts.
* **mouse()-** It is used for completely stopping the program on clicking the left mouse button.
* **inp\_keyboard()-** All the input variables are converted to booleans to use for other internal functions, using switch statements, called directly in the main function so that its atop the stack of execution, thus always available, and responsiveness to our inputs is at its peak
* **my\_init()-** This function is used to initialize the variables

* **display()-** This function is used to display and combine the visual and working elements of the program, into a single call for main to call the visual elements (train\_station()) and the working elements (idle() and traffic\_light())

**CODE:**

#include<stdio.h>

#include<GL/glut.h>

#include <GL/gl.h>

#include <stdlib.h>

#include<iostream>

using namespace std;

#define SPEED 30.0

float i = 0.0, m = 0.0, n = 0.0, o = 0.0, c = 0.0;

int light = 1, day = 1, plane = 0, comet = 0, xm = 900, train = 0;

char ch;

void draw\_pixel(GLint cx, GLint cy)

{

glBegin(GL\_POINTS);

glVertex2i(cx, cy);

glEnd();

}

void plotpixels(GLint h, GLint k, GLint x, GLint y)

{

draw\_pixel(x + h, y + k);

draw\_pixel(-x + h, y + k);

draw\_pixel(x + h, -y + k);

draw\_pixel(-x + h, -y + k);

draw\_pixel(y + h, x + k);

draw\_pixel(-y + h, x + k);

draw\_pixel(y + h, -x + k);

draw\_pixel(-y + h, -x + k);

}

void draw\_circle(GLint h, GLint k, GLint r)

{

GLint d = 1 - r, x = 0, y = r;

while (y > x)

{

plotpixels(h, k, x, y);

if (d < 0) d += 2 \* x + 3;

else

{

d += 2 \* (x - y) + 5;

--y;

}

++x;

}

plotpixels(h, k, x, y);

}

void train\_station()

{

int l;

if (day == 1)

{

//sky

glColor3f(0.0, 0.9, 0.9);

glBegin(GL\_POLYGON);

glVertex2f(0, 380);

glVertex2f(0, 700);

glVertex2f(1100, 700);

glVertex2f(1100, 380);

glEnd();

//sun

for (l = 0; l <= 35; l++)

{

glColor3f(1.0, 0.9, 0.0);

draw\_circle(100, 625, l);

}

//cloud1

for (l = 0; l <= 20; l++)

{

glColor3f(1.0, 1.0, 1.0);

draw\_circle(160 + m, 625, l);

}

for (l = 0; l <= 35; l++)

{

glColor3f(1.0, 1.0, 1.0);

draw\_circle(200 + m, 625, l);

draw\_circle(225 + m, 625, l);

}

for (l = 0; l <= 20; l++)

{

glColor3f(1.0, 1.0, 1.0);

draw\_circle(265 + m, 625, l);

}

//cloud2

for (l = 0; l <= 20; l++)

{

glColor3f(1.0, 1.0, 1.0);

draw\_circle(370 + m, 615, l);

}

for (l = 0; l <= 35; l++)

{

glColor3f(1.0, 1.0, 1.0);

draw\_circle(410 + m, 615, l);

draw\_circle(435 + m, 615, l);

draw\_circle(470 + m, 615, l);

}

for (l = 0; l <= 20; l++)

{

glColor3f(1.0, 1.0, 1.0);

draw\_circle(500 + m, 615, l);

}

//grass

glColor3f(0.0, 0.9, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(0, 160);

glVertex2f(0, 380);

glVertex2f(1100, 380);

glVertex2f(1100, 160);

glEnd();

}

else

{

//sky

glColor3f(0.0, 0.0, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(0, 380);

glVertex2f(0, 700);

glVertex2f(1100, 700);

glVertex2f(1100, 380);

glEnd();

//moon

int l;

for (l = 0; l <= 35; l++)

{

glColor3f(1.0, 1.0, 1.0);

draw\_circle(100, 625, l);

}

//comet

if (comet == 1)

{

for (l = 0; l <= 7; l++)

{

glColor3f(1.0, 1.0, 1.0);

draw\_circle(300 + c, 675, l);

}

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_TRIANGLES);

glVertex2f(200 + c, 675);

glVertex2f(300 + c, 682);

glVertex2f(300 + c, 668);

glEnd();

}

//grass

glColor3f(0.0, 0.3, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(0, 160);

glVertex2f(0, 380);

glVertex2f(1100, 380);

glVertex2f(1100, 160);

glEnd();

}

//platform

glColor3f(0.560784, 0.560784, 0.737255);

glBegin(GL\_POLYGON);

glVertex2f(0, 160);

glVertex2f(0, 250);

glVertex2f(1100, 250);

glVertex2f(1100, 160);

glEnd();

//table 1

glColor3f(1.0, 0.498039, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(140, 190);

glVertex2f(140, 210);

glVertex2f(155, 210);

glVertex2f(155, 190);

glEnd();

glColor3f(0.2, 0.2, 0.2);

glBegin(GL\_POLYGON);

glVertex2f(130, 210);

glVertex2f(130, 215);

glVertex2f(225, 215);

glVertex2f(225, 210);

glEnd();

glColor3f(1.0, 0.498039, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(200, 190);

glVertex2f(200, 210);

glVertex2f(215, 210);

glVertex2f(215, 190);

glEnd();

//below track

glColor3f(0.623529, 0.623529, 0.372549);

glBegin(GL\_POLYGON);

glVertex2f(0, 0);

glVertex2f(0, 150);

glVertex2f(1100, 150);

glVertex2f(1100, 0);

glEnd();

//tree 1

glColor3f(0.9, 0.2, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(50, 185);

glVertex2f(50, 255);

glVertex2f(65, 255);

glVertex2f(65, 185);

glEnd();

for (l = 0; l <= 30; l++)

{

glColor3f(0.31, 0.78, 0.47);

draw\_circle(40, 250, l);

draw\_circle(80, 250, l);

}

for (l = 0; l <= 25; l++)

{

glColor3f(0.31, 0.78, 0.47);

draw\_circle(50, 290, l);

draw\_circle(70, 290, l);

}

for (l = 0; l <= 20; l++)

{

glColor3f(0.31, 0.78, 0.47);

draw\_circle(60, 315, l);

}

//tree 2

glColor3f(0.9, 0.2, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(300, 185);

glVertex2f(300, 255);

glVertex2f(315, 255);

glVertex2f(315, 185);

glEnd();

for (l = 0; l <= 30; l++)

{

glColor3f(0.31, 0.78, 0.47);

draw\_circle(290, 250, l);

draw\_circle(330, 250, l);

}

for (l = 0; l <= 25; l++)

{

glColor3f(0.31, 0.78, 0.47);

draw\_circle(300, 290, l);

draw\_circle(320, 290, l);

}

for (l = 0; l <= 20; l++)

{

glColor3f(0.31, 0.78, 0.47);

draw\_circle(310, 315, l);

}

//railway station

glColor3f(0.647, 0.164, 0.164);

glBegin(GL\_POLYGON);

glVertex2f(400, 250);

glVertex2f(400, 450);

glVertex2f(950, 450);

glVertex2f(950, 250);

glEnd();

//roof

glColor3f(0.556863, 0.419608, 0.137255);

glBegin(GL\_POLYGON);

glVertex2f(350, 450);

glVertex2f(450, 500);

glVertex2f(900, 500);

glVertex2f(1000, 450);

glEnd();

//window 1

glColor3f(0.196, 0.6, 0.8);

glBegin(GL\_POLYGON);

glVertex2f(450, 300);

glVertex2f(450, 375);

glVertex2f(550, 375);

glVertex2f(550, 300);

glEnd();

//window 2

glColor3f(0.196, 0.6, 0.8);

glBegin(GL\_POLYGON);

glVertex2f(800, 300);

glVertex2f(800, 375);

glVertex2f(900, 375);

glVertex2f(900, 300);

glEnd();

//door

glColor3f(0.329412, 0.329412, 0.329412);

glBegin(GL\_POLYGON);

glVertex2f(625, 250);

glVertex2f(625, 375);

glVertex2f(725, 375);

glVertex2f(725, 250);

glEnd();

//signal

glColor3f(1.0, 0.0, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(1060, 160);

glVertex2f(1060, 350);

glVertex2f(1070, 350);

glVertex2f(1070, 160);

glEnd();

glColor3f(0.7, 0.7, 0.7);

glBegin(GL\_POLYGON);

glVertex2f(1040, 350);

glVertex2f(1040, 500);

glVertex2f(1090, 500);

glVertex2f(1090, 350);

glEnd();

for (l = 0; l <= 20; l++)

{

glColor3f(0.0, 0.0, 0.0);

draw\_circle(1065, 475, l);

glColor3f(1.0, 1.0, 0.0);

draw\_circle(1065, 425, l);

glColor3f(0.0, 0.0, 0.0);

draw\_circle(1065, 375, l);

}

if (train == 1)

{

// train carrier 1

glColor3f(0.26, 0.43, 0.26);

glBegin(GL\_POLYGON);

glVertex2f(300 + i - xm, 50);

glVertex2f(300 + i - xm, 300);

glVertex2f(-100 + i - xm, 300);

glVertex2f(-100 + i - xm, 50);

glEnd();

// carrier 1 Wheels

for (l = 0; l < 50; l++)

{

glColor3f(0.35, 0.16, 0.14);

draw\_circle(-25 + i - xm, 50, l);

draw\_circle(225 + i - xm, 50, l);

}

//train base

glColor3f(0.196078, 0.6, 0.8);

glBegin(GL\_POLYGON);

glVertex2f(350 + i - xm, 50);

glVertex2f(350 + i - xm, 125);

glVertex2f(800 + i - xm, 125);

glVertex2f(800 + i - xm, 50);

glEnd();

//train control chamber

glColor3f(0.26, 0.43, 0.26);

glBegin(GL\_POLYGON);

glVertex2f(360 + i - xm, 125);

glVertex2f(360 + i - xm, 325);

glVertex2f(560 + i - xm, 325);

glVertex2f(560 + i - xm, 125);

glEnd();

//train engine

glColor3f(0, 0, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(560 + i - xm, 125);

glVertex2f(560 + i - xm, 225);

glVertex2f(755 + i - xm, 225);

glVertex2f(755 + i - xm, 125);

glEnd();

//train smoke

glColor3f(0.196078, 0.6, 0.9);

glBegin(GL\_POLYGON);

glVertex2f(650 + i - xm, 225);

glVertex2f(650 + i - xm, 275);

glVertex2f(700 + i - xm, 275);

glVertex2f(700 + i - xm, 225);

glEnd();

//train head-light

glColor3f(1.0, 1.0, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(755 + i - xm, 225);

glVertex2f(765 + i - xm, 225);

glVertex2f(765 + i - xm, 185);

glVertex2f(755 + i - xm, 185);

glEnd();

// train connector

glColor3f(0.309804, 0.184314, 0.184314);

glBegin(GL\_POLYGON);

glVertex2f(350 + i - xm, 75);

glVertex2f(350 + i - xm, 95);

glVertex2f(300 + i - xm, 95);

glVertex2f(300 + i - xm, 75);

glEnd();

//train wheels

for (l = 0; l < 50; l++)

{

glColor3f(0.35, 0.16, 0.14);

draw\_circle(425 + i - xm, 50, l);

draw\_circle(700 + i - xm, 50, l);

}

}

glFlush();

}

void traffic\_light()

{

int l;

if (light == 1)

{

for (l = 0; l <= 20; l++)

{

glColor3f(0.0, 0.0, 0.0);

draw\_circle(1065, 475, l);

glColor3f(0.0, 0.7, 0.0);

draw\_circle(1065, 375, l);

}

}

else

{

for (l = 0; l <= 20; l++)

{

glColor3f(1.0, 0.0, 0.0);

draw\_circle(1065, 475, l);

glColor3f(0.0, 0.0, 0.0);

draw\_circle(1065, 375, l);

}

}

}

void idle()

{

glClearColor(1.0, 1.0, 1.0, 1.0);

if (light == 0 && (i >= 0 && i <= 1150))

{

i += SPEED / 10;

m += SPEED / 150;

n -= 2;

o += 0.2;

c += 2;

}

if (light == 0 && (i >= 2600 && i <= 3000))

{

i += SPEED / 10;

m += SPEED / 150;

n -= 2;

o += 0.2;

c += 2;

}

if (light == 0)

{

i = i;

m += SPEED / 150;

n -= 2;

o += 0.2;

c += 2;

}

else

{

i += SPEED / 10;

m += SPEED / 150;

n -= 2;

o += 0.2;

c += 2;

}

if (i > 3500)

i = 0.0;

if (m > 1100)

m = 0.0;

if (o > 75)

{

plane = 0;

}

if (c > 500)

{

comet = 0;

}

glutPostRedisplay();

}

void mouse(int btn, int state, int x, int y)

{

if (btn == GLUT\_LEFT\_BUTTON && state == GLUT\_UP)

exit(0);

}

void inp\_keyboard(unsigned char key, int x, int y)

{

switch (key)

{

case 'g':

case 'G':

light = 1;

break;

case 'r':

case 'R':

light = 0;

break;

case 'd':

case 'D':

day = 1;

break;

case 'n':

case 'N':

day = 0;

break;

case 't':

case 'T':

train = 1;

i = 0;

break;

};

}

void main\_menu(int index)

{

switch (index)

{

case 1:

if (index == 1)

{

plane = 1;

o = n = 0.0;

}

break;

case 2:

if (index == 2)

{

comet = 1;

c = 0.0;

}

break;

}

}

void myinit()

{

glClearColor(1.0, 1.0, 1.0, 1.0);

glColor3f(0.0, 0.0, 1.0);

glPointSize(2.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(0.0, 1100.0, 0.0, 700.0);

}

void display()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

train\_station();

traffic\_light();

glFlush();

}

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

{

int c\_menu,test\_lol;

printf(" Train Station Simulator \n\n\n");

printf("'r' to change the signal light to red. \n\n");

printf("'g' to change the signal light to green. \n\n");

printf("'d' for day. \n\n");

printf("'n' for night time \n\n");

printf("'t' for train To start moving\n\n");

printf("Right Click To show contextual menu \n\n");

printf("Left Click To end The Program \n\n\n");

printf("Press any key and Hit ENTER.\n");

cin >> test\_lol;

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(1100.0, 700.0);

glutInitWindowPosition(0, 0);

glutCreateWindow("Train Station");

glutDisplayFunc(display);

glutIdleFunc(idle);

glutKeyboardFunc(inp\_keyboard);

glutMouseFunc(mouse);

myinit();

//menu operations, aeroplane(rip) and comet

c\_menu = glutCreateMenu(main\_menu);

glutAddMenuEntry("Aeroplane", 1);

glutAddMenuEntry("Comet", 2);

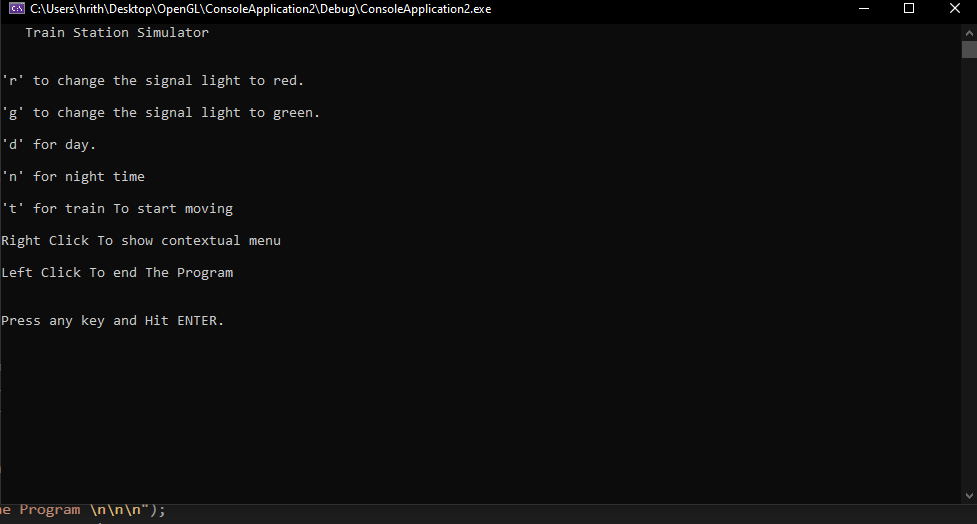
glutAttachMenu(GLUT\_RIGHT\_BUTTON);

glutMainLoop();

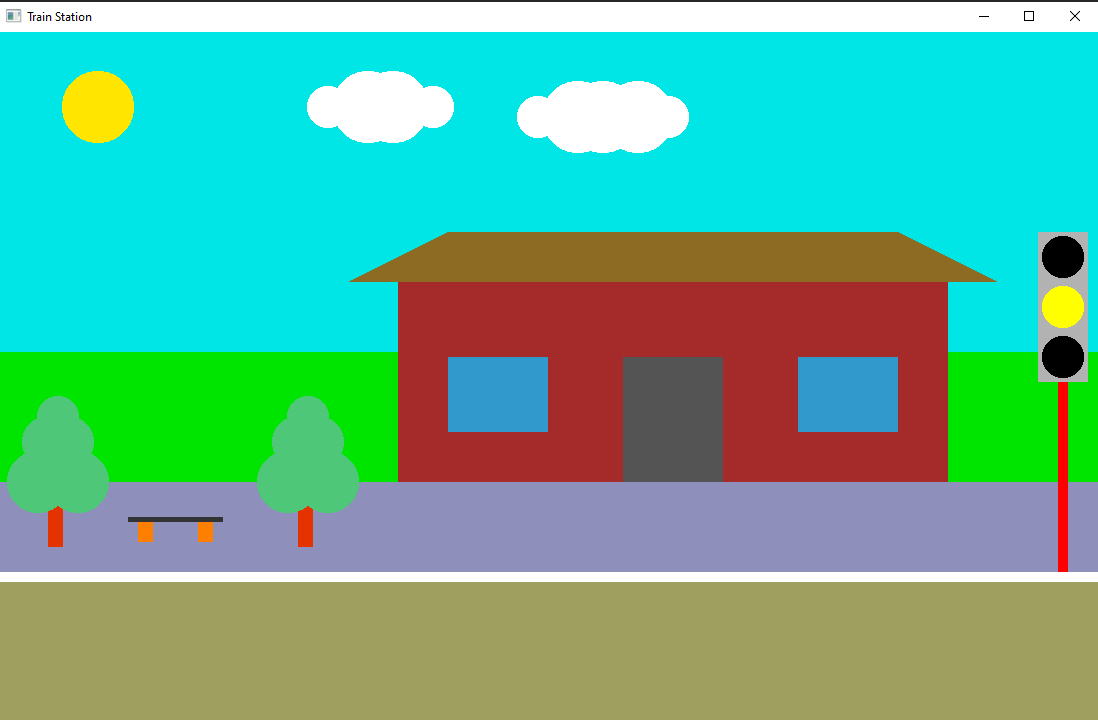
return 0;

}

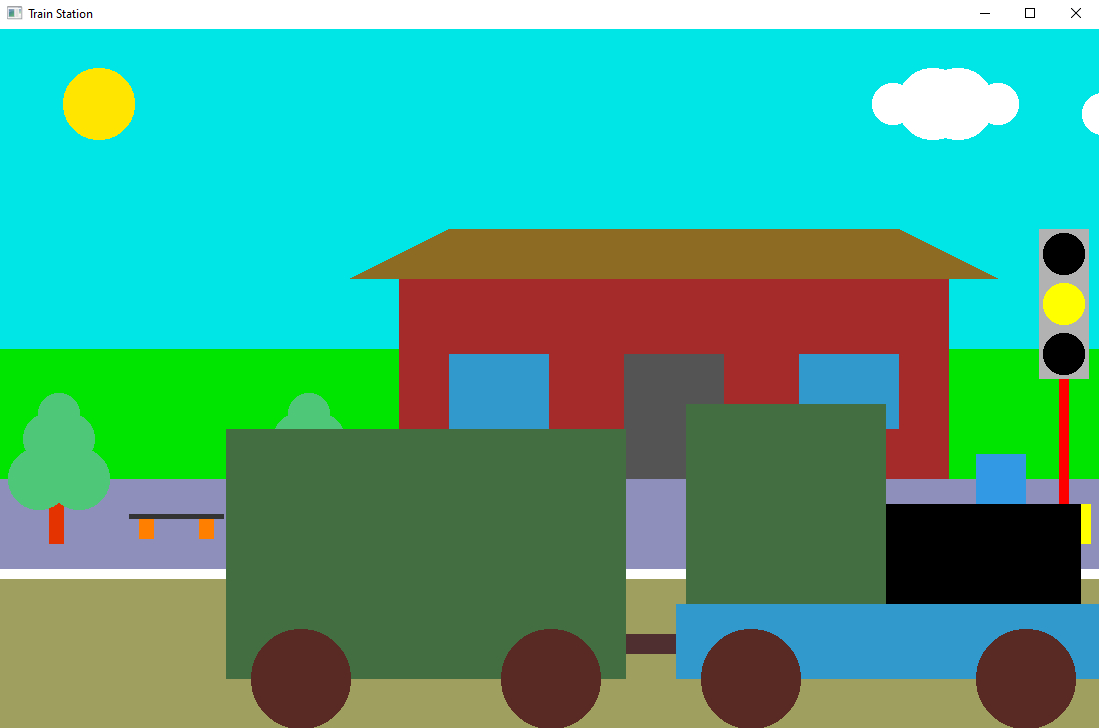
**SNAPSHOTS:**



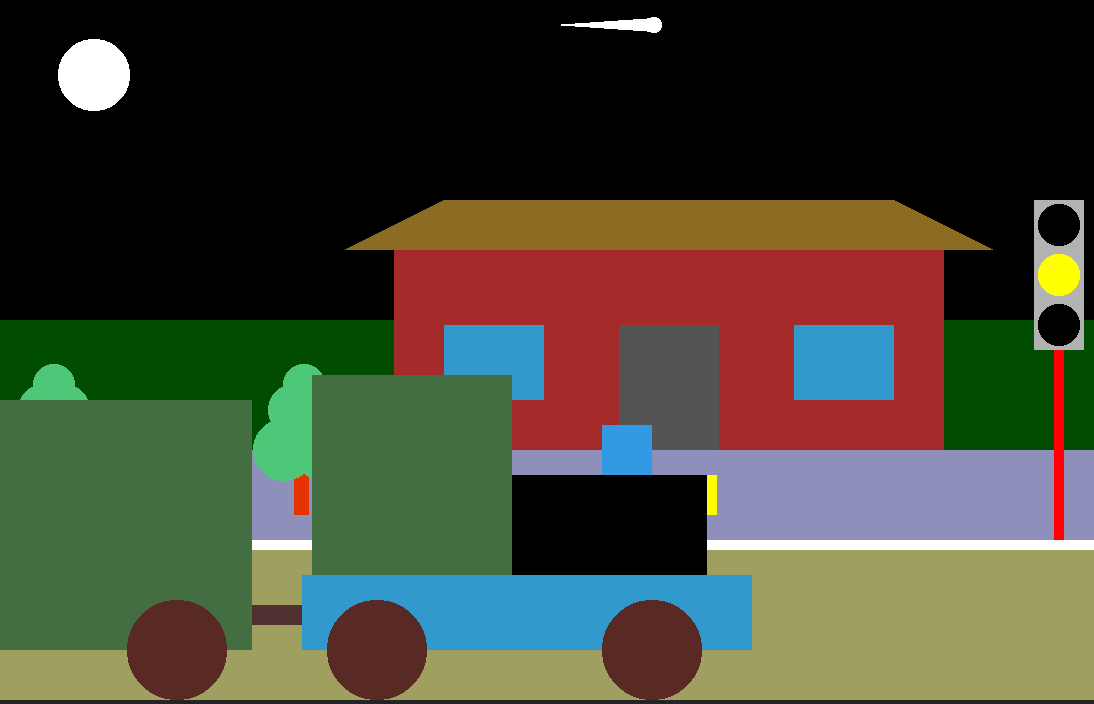
Initially, we are presented with this screen, detailing us with all the workings of the program, and what different buttons do. After entering any key, we enter the graphical application



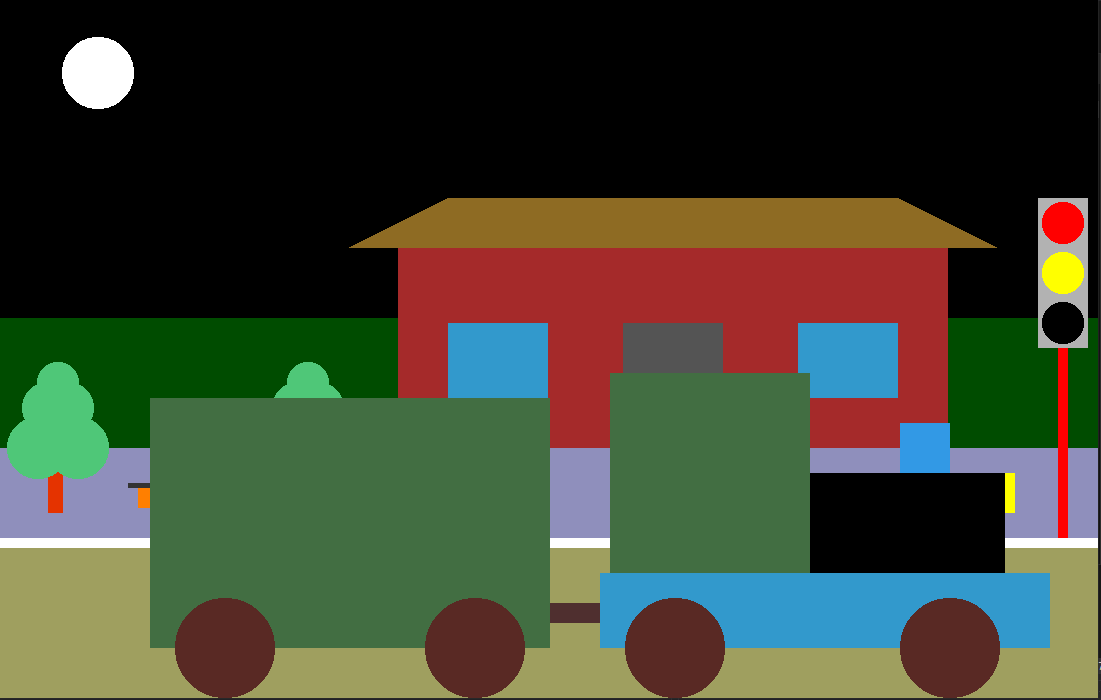
This is the initial environment. We can see moving clouds and the railway station with trees and benches, with the platform as well. Now we can enter t for the train to start moving.



So, the train started moving. We can also change the time of day to night by pressing the n key, and day by the d key. In the night mode, we can use the right click menu to see a comet we animated.



Now, we can use the r button to activate the red light, if in motion, the train stops at the end of screen, otherwise stops immediately if it has moved past the traffic light.



Pressing g key now starts the motion of the train again, as it turns the traffic light green again.

