Lab Taks-6

Submission Guidelines-

- Rename the file to your id only. If your id is 18-XXXXX-1, then the file name must be 18-XXXXX-1.docx.
- Must include resources for all the section in the table

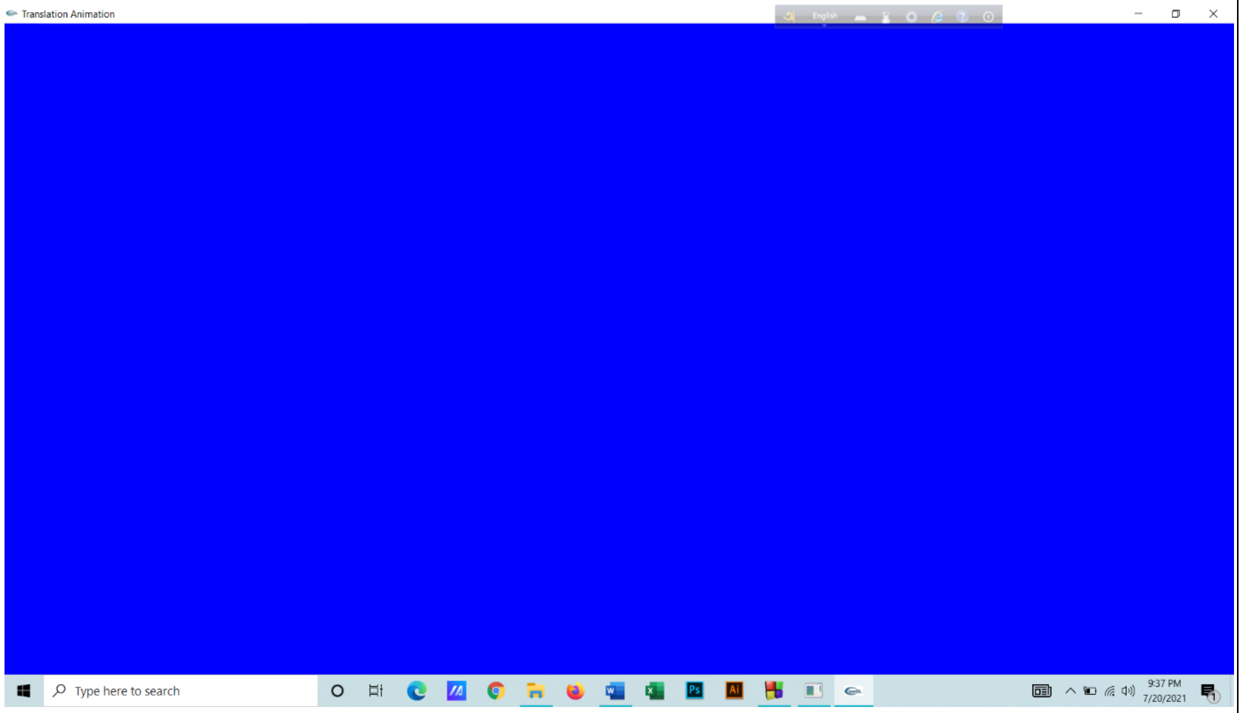
Question-
Develop an animation that will change the background color of the window after 20ms. Use at least two different colors.

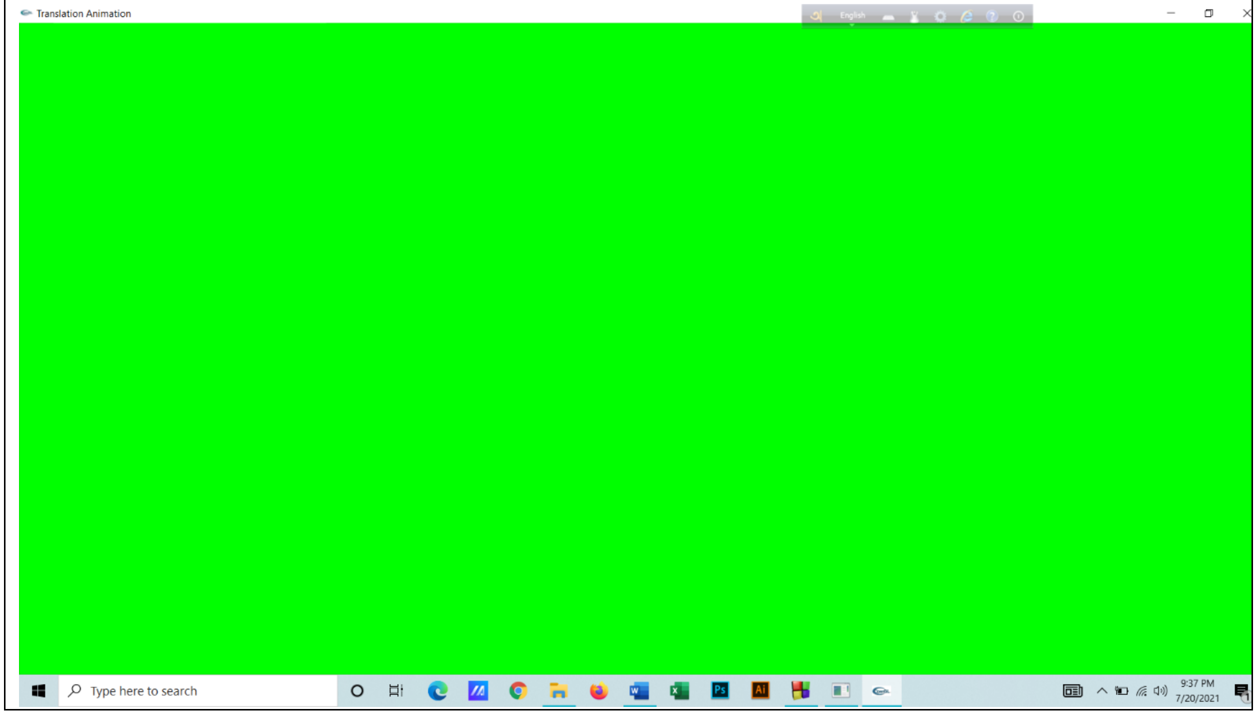
Code-

```
#include<cstdio>

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


GLfloat position = 0.0f;
GLfloat position1 = 0.0f;

GLfloat speed = 0.1f;
void dis();
void display();

void update(int value) {

    if(position <-1.5)
        position = 1.0f;

    position -= speed;

            glutPostRedisplay();


            glutTimerFunc(20,update,0);
}


void update1(int value) {
    if(position1 >1.0)
        position1 = -1.0f;

    position1 += speed;

            glutPostRedisplay();


            glutTimerFunc(20,update1,0);
}

void init() {
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
}

void disback(int val)
{
    glutDisplayFunc(display);
}

void display5()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glClearColor(0.0f, 0.0f, 1.0f, 1.0f);

   glPopMatrix();
   glutTimerFunc(1500,disback,0);
   glFlush();
}


void display4(int val) {

 glutDisplayFunc(display5);



}




void display3()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glClearColor(0.0f, 1.0f, 0.0f, 1.0f);

   glPopMatrix();
   glutTimerFunc(1500,display4,0);
   glFlush();
}

void display2(int val) {

 glutDisplayFunc(display3);


}
void display() {
   glClear(GL_COLOR_BUFFER_BIT);
   glLoadIdentity();
glClearColor(1.0f, 1.0f, 1.0f, 1.0f);


glPushMatrix();

glPopMatrix();

glutTimerFunc(1500,display2,0);
glFlush();

}

void dis()
{
       glutDisplayFunc(display);
}

int main(int argc, char** argv) {
   glutInit(&argc, argv);
   glutInitWindowSize(320, 320);
   glutInitWindowPosition(50, 50);
   glutCreateWindow("Translation Animation");
   glutDisplayFunc(dis);
   init();

   glutTimerFunc(20, update, 0);
     glutTimerFunc(20, update1, 0);
```

```
  glutMainLoop();
  return 0;
}
```

Output Screenshot (Full Screen)-





Question-
Develop an animation that will call four objects separately, each after 20ms.

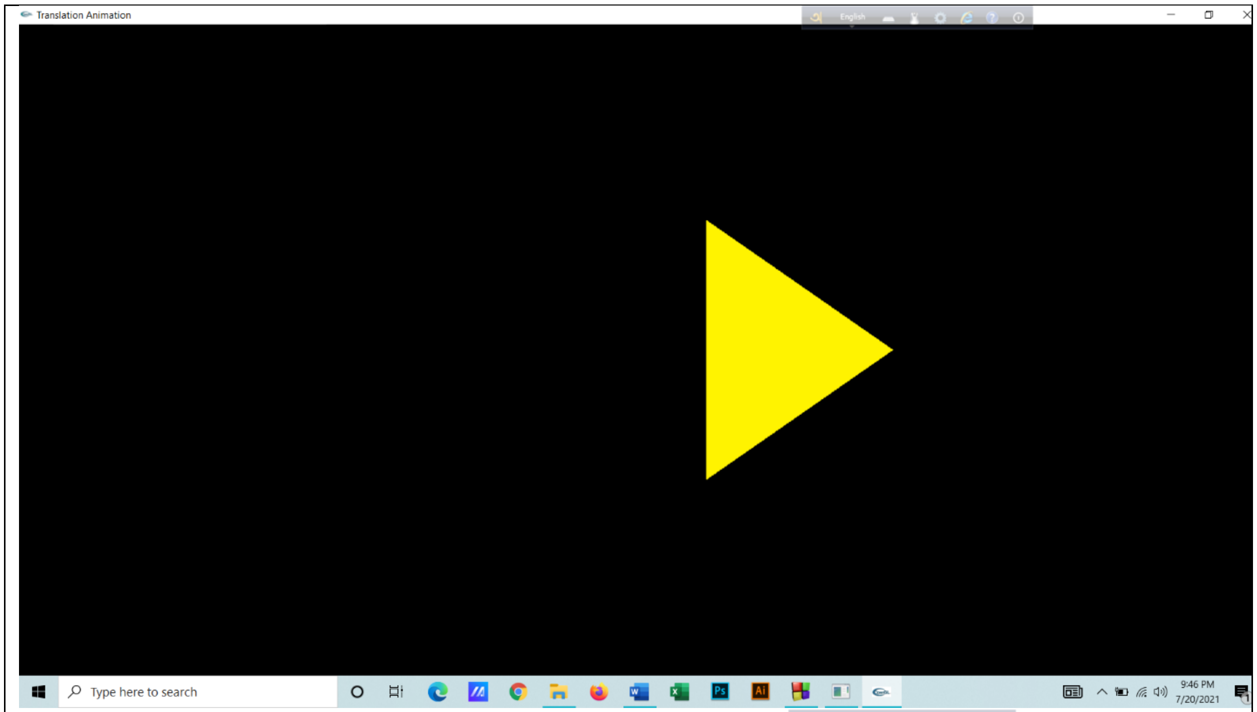
Code-

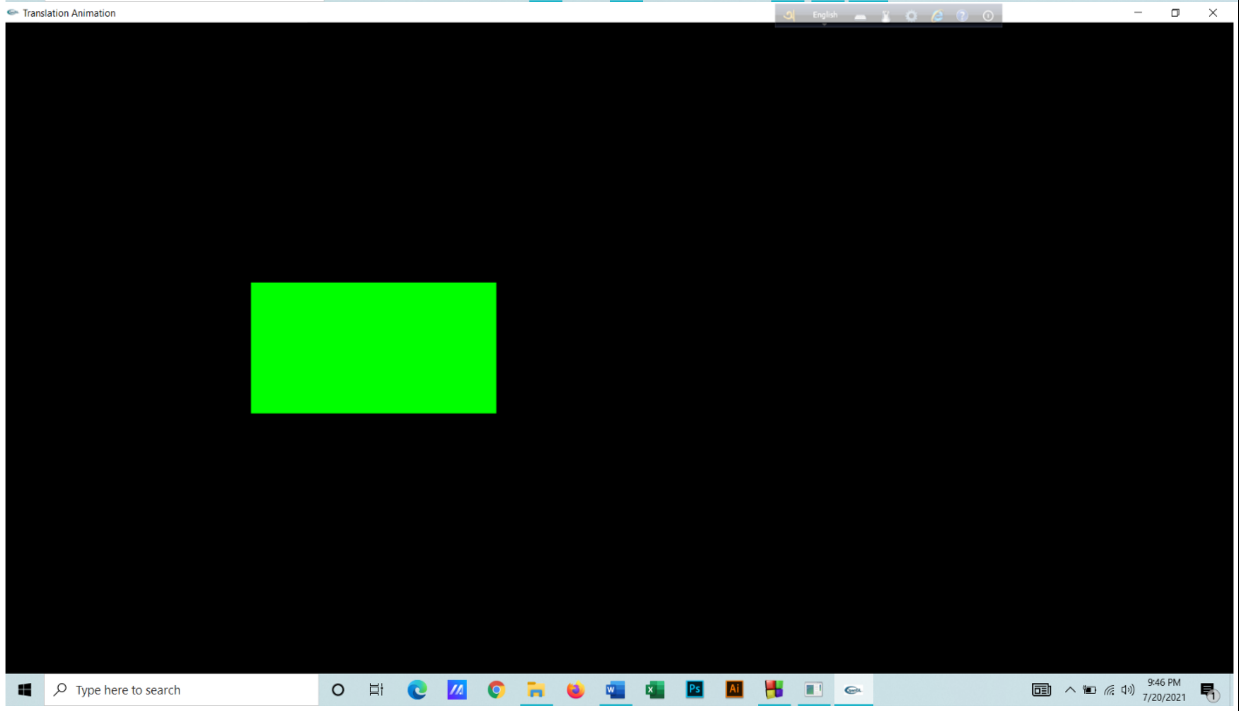
```
#include<cstdio>

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



GLfloat position = 0.0f;
GLfloat position1 = 0.0f;
GLfloat speed = 0.1f;
void dis();
void display();

void update(int value) {

   if(position <-1.5)
      position = 1.0f;

   position -= speed;

          glutPostRedisplay();


          glutTimerFunc(20,update,0);
}


void update1(int value) {

   if(position1 >1.0)
      position1 = -1.0f;

   position1 += speed;

          glutPostRedisplay();


          glutTimerFunc(20,update1,0);
}
```

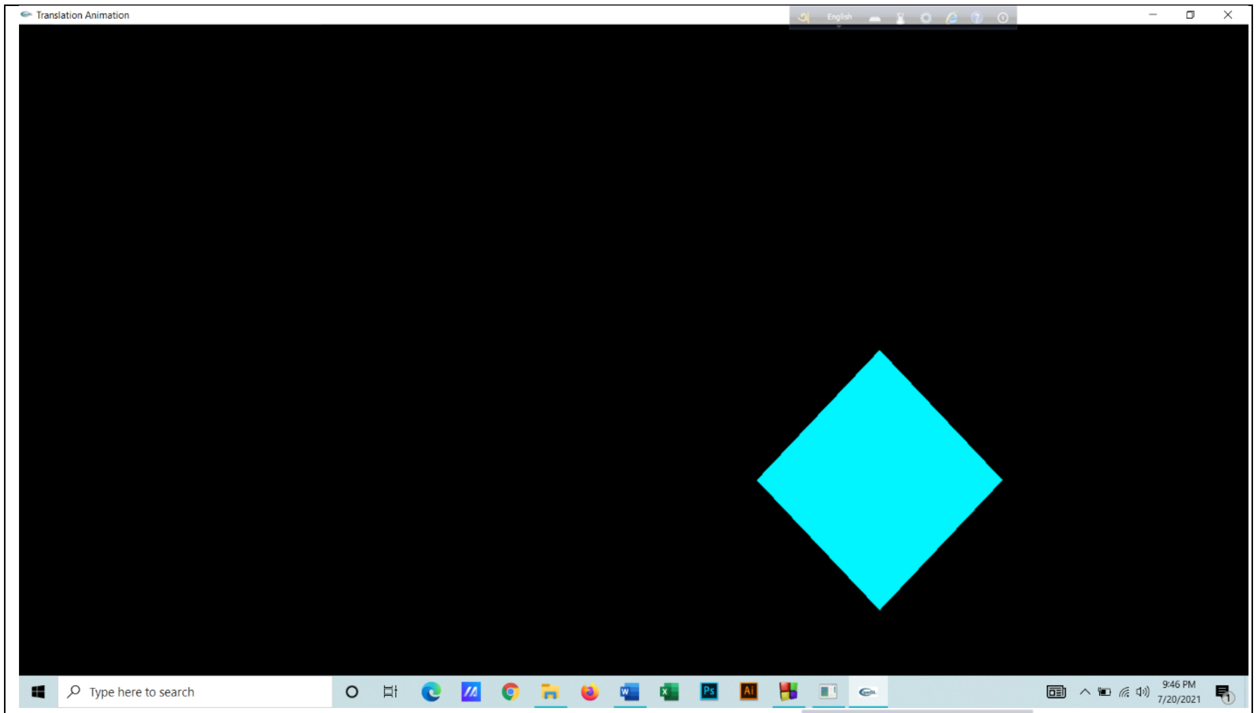
```
void init() {
  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
}

void disback(int val)
{
   glutDisplayFunc(display);
}

void display7()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glPushMatrix();
glTranslatef(0.0f,position, 0.0f);

   glBegin(GL_QUADS);           // These vertices form a closed polygon
           glColor3ub(248, 0, 255  );
           glVertex2f(-0.5f, 0.1f);
           glVertex2f(0.0f, 0.1f);
           glVertex2f(-0.1f, 0.5f);
   glVertex2f(-0.4f, 0.5f);
   glEnd();
  glPopMatrix();
  glutTimerFunc(1500,disback,0);
  glFlush();
}

void display6(int val) {

 glutDisplayFunc(display7);



}

void display5()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glPushMatrix();
glTranslatef(0.0f,position, 0.0f);
    glBegin(GL_QUADS);           // These vertices form a closed polygon
           glColor3ub(0, 245, 255 );
           glVertex2f(0.2f, 0.2f);
```

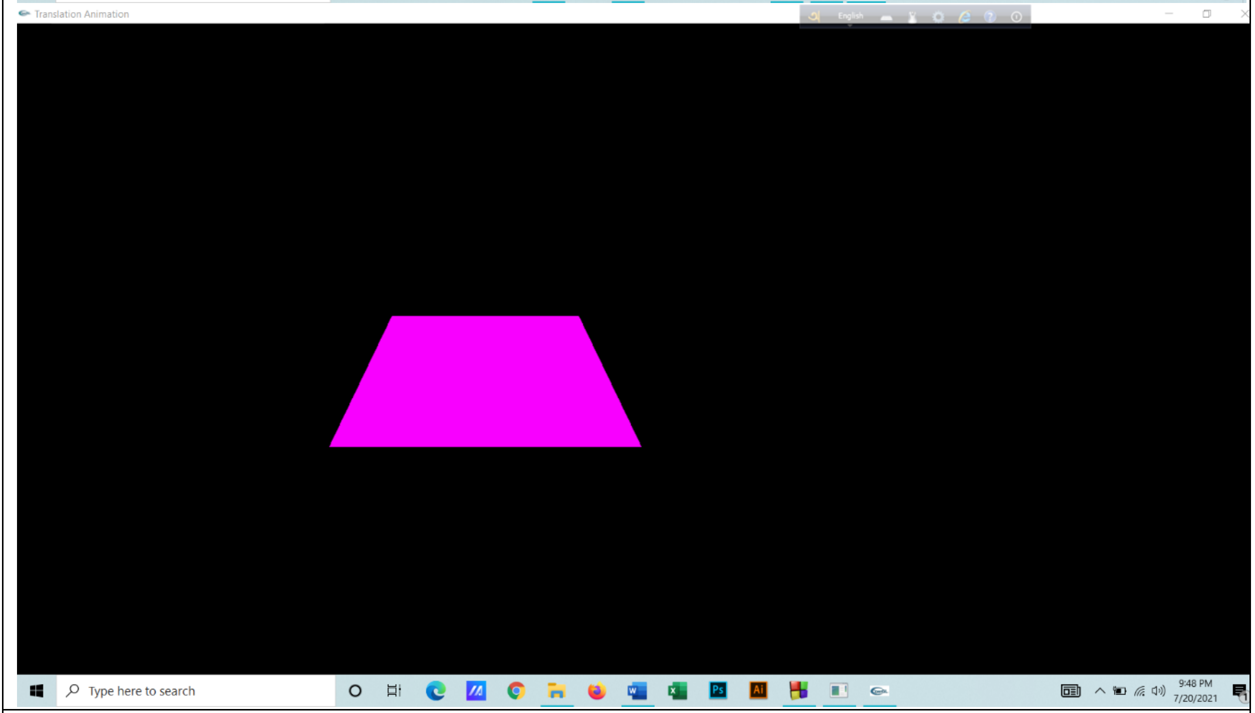
```
            glVertex2f(0.4f, -0.2f);
            glVertex2f(0.6f, 0.2f);
    glVertex2f(0.4f, 0.6f);
    glEnd();
   glPopMatrix();
   glutTimerFunc(1500,display6,0);
   glFlush();

}

void display4(int val) {

 glutDisplayFunc(display5);


}



void display3()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
    glPushMatrix();
glTranslatef(position1,0.0f, 0.0f);
   glBegin(GL_QUADS);
      glColor3f(0.0f, 1.0f, 0.0f);
      glVertex2f(-0.2f, -0.2f);
      glVertex2f( 0.2f, -0.2f);
      glVertex2f( 0.2f,  0.2f);
      glVertex2f(-0.2f,  0.2f);
   glEnd();
   glPopMatrix();
   glutTimerFunc(1500,display4,0);
   glFlush();
}

void display2(int val) {

 glutDisplayFunc(display3);


}
void display() {
  glClear(GL_COLOR_BUFFER_BIT);
  glLoadIdentity();



glPushMatrix();
glTranslatef(position,0.0f, 0.0f);

  glBegin(GL_TRIANGLES);
  glColor3ub(255, 243, 0 );
  glVertex2f(0.4f,-0.4f);
  glVertex2f(0.7f, 0.0f);
  glVertex2f(0.4f, 0.4f);
  glEnd();

glPopMatrix();

glutTimerFunc(1500,display2,0);
glFlush();

}

void dis()
{
     glutDisplayFunc(display);
}

int main(int argc, char** argv) {
  glutInit(&argc, argv);
  glutInitWindowSize(320, 320);
  glutInitWindowPosition(50, 50);
  glutCreateWindow("Translation Animation");
  glutDisplayFunc(dis);
  init();

  glutTimerFunc(20, update, 0);
   glutTimerFunc(20, update1, 0);
  glutMainLoop();
  return 0;
}
```

Output Screenshot (Full Screen)-


Translation Animation
English
Type here to search
9:46 PM
7/20/2021


Translation Animation
English
Type here to search
9:46 PM
7/20/2021





Question-

Develop a code that will have four different objects (keep it simple). The objects will move to the left, right, up and down in a loop.

Code-

```
#include<cstdio>
```

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


GLfloat position = 0.0f;
GLfloat position1 = 0.0f;
GLfloat position2 = 0.0f;
GLfloat speed = 0.1f;
void dis();
void display();

void update(int value) {

   if(position <-1.5)
      position = 1.0f;

   position -= speed;

          glutPostRedisplay();


          glutTimerFunc(100,update,0);
}


void update1(int value) {

   if(position1 >1.0)
      position1 = -1.0f;

   position1 += speed;

          glutPostRedisplay();


          glutTimerFunc(100,update1,0);
}

void update2(int value) {

   if(position2 >-1.5)
      position2 = -1.0f;

   position2 += speed;
```

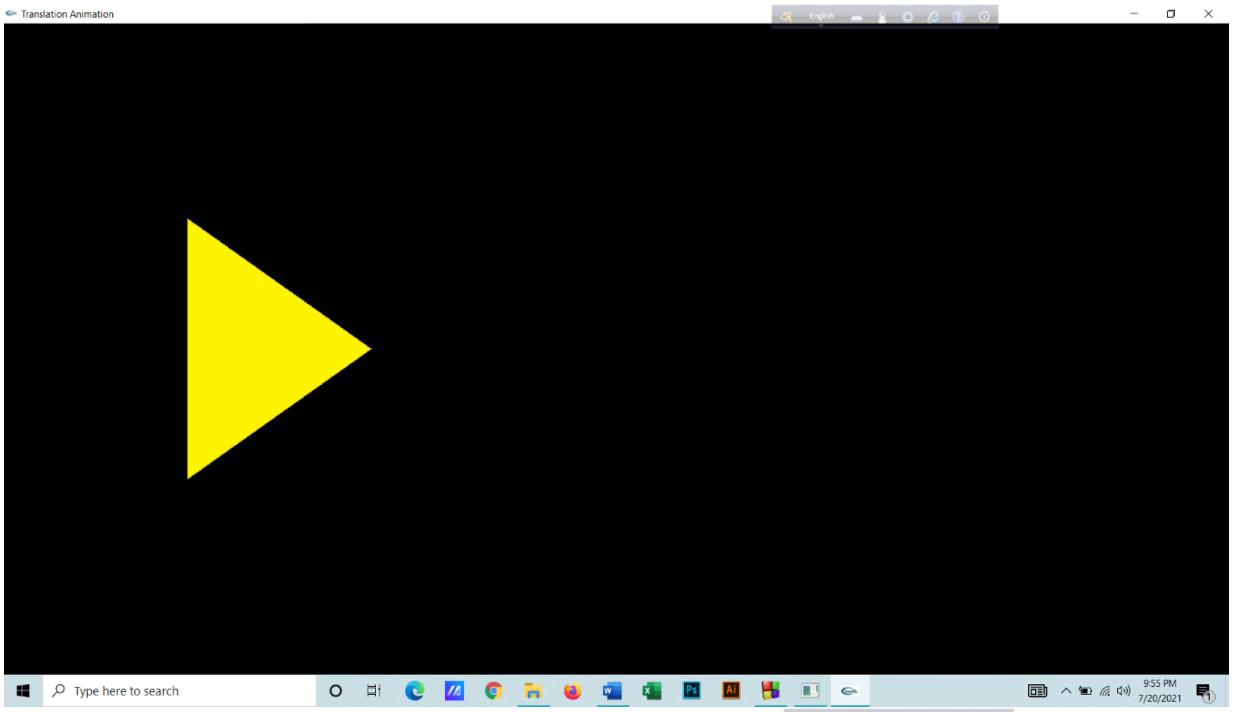
```
            glutPostRedisplay();


            glutTimerFunc(100,update2,0);
}

void init() {
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
}

void disback(int val)
{
    glutDisplayFunc(display);
}

void display7()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
    glPushMatrix();
glTranslatef(0.0f,position1, 0.0f);

    glBegin(GL_QUADS);            // These vertices form a closed polygon
            glColor3ub(248, 0, 255  );
            glVertex2f(-0.5f, 0.1f);
            glVertex2f(0.0f, 0.1f);
            glVertex2f(-0.1f, 0.5f);
    glVertex2f(-0.4f, 0.5f);
    glEnd();
   glPopMatrix();
   glutTimerFunc(1500,disback,0);
   glFlush();
}

void display6(int val) {

 glutDisplayFunc(display7);



}

void display5()
{
```

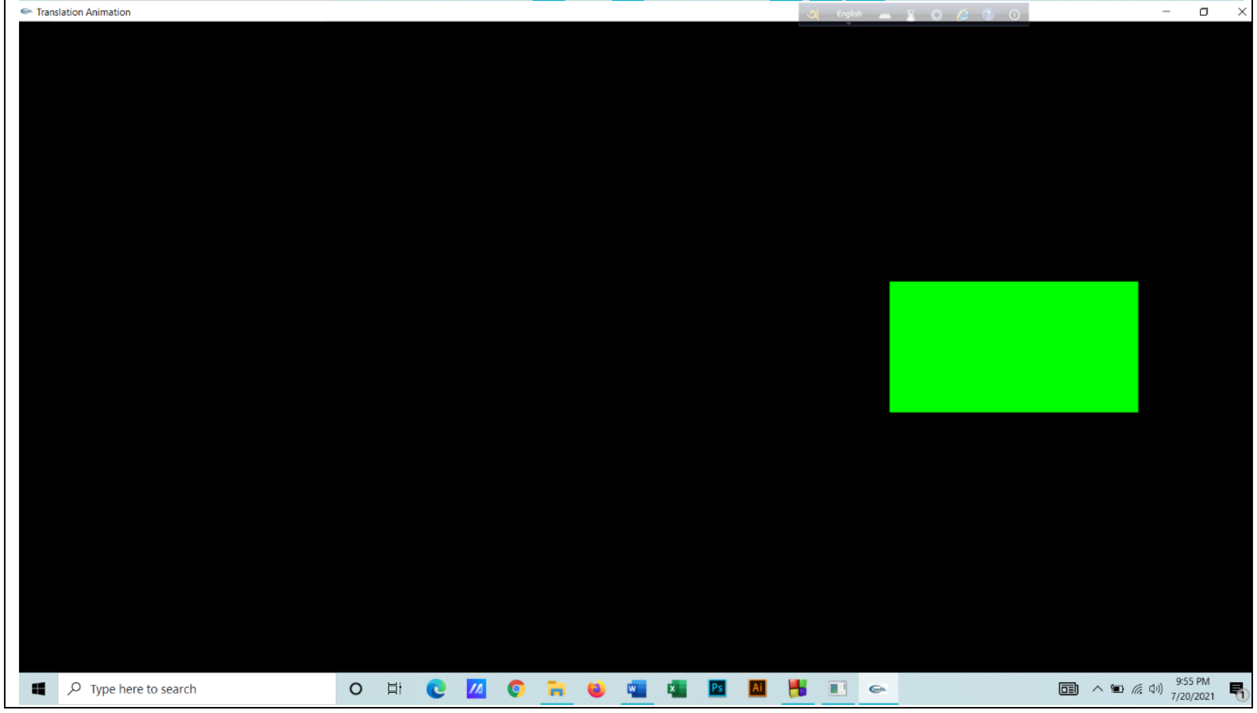
}

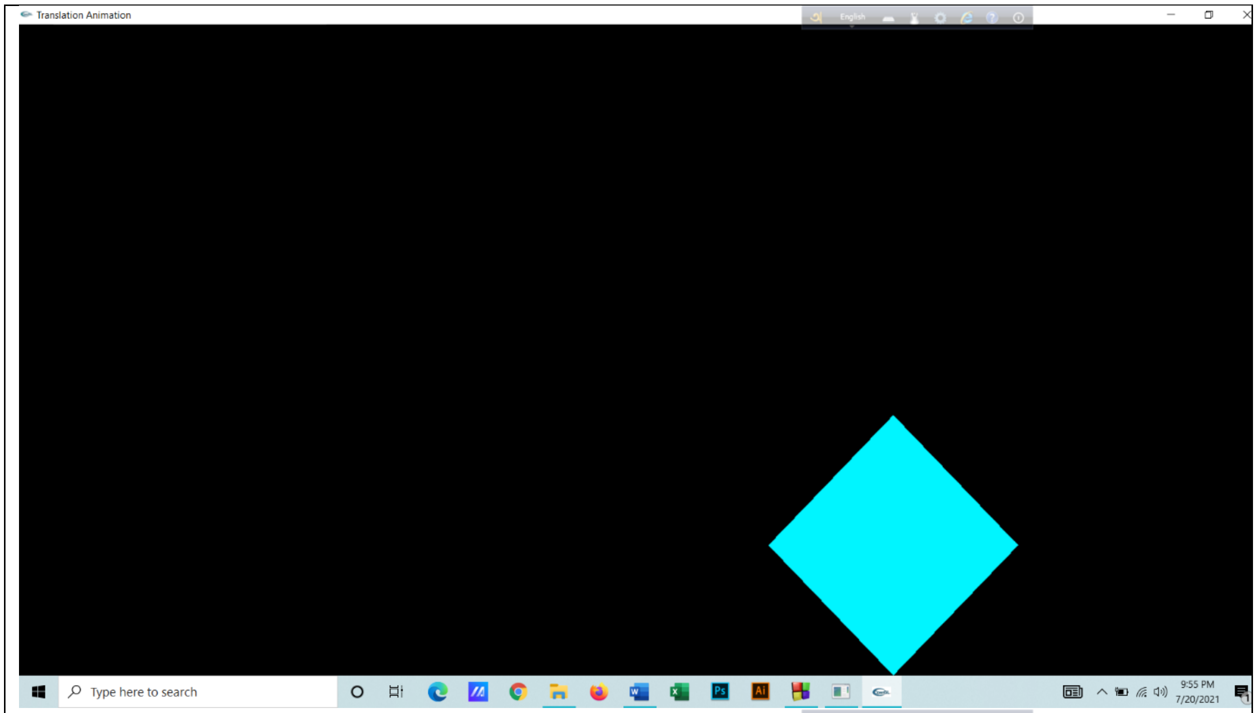
```
   glClear(GL_COLOR_BUFFER_BIT);
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glPushMatrix();
glTranslatef(0.0f,position, 0.0f);
    glBegin(GL_QUADS);           // These vertices form a closed polygon
           glColor3ub(0, 245, 255 );
           glVertex2f(0.2f, 0.2f);
           glVertex2f(0.4f, -0.2f);
           glVertex2f(0.6f, 0.2f);
   glVertex2f(0.4f, 0.6f);
   glEnd();
  glPopMatrix();
  glutTimerFunc(1500,display6,0);
  glFlush();

}

void display4(int val) {

 glutDisplayFunc(display5);


}



void display3()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glPushMatrix();
glTranslatef(position1,0.0f, 0.0f);
  glBegin(GL_QUADS);
     glColor3f(0.0f, 1.0f, 0.0f);
     glVertex2f(-0.2f, -0.2f);
     glVertex2f( 0.2f, -0.2f);
     glVertex2f( 0.2f,  0.2f);
     glVertex2f(-0.2f,  0.2f);
  glEnd();
  glPopMatrix();
  glutTimerFunc(1500,display4,0);
  glFlush();
}
```

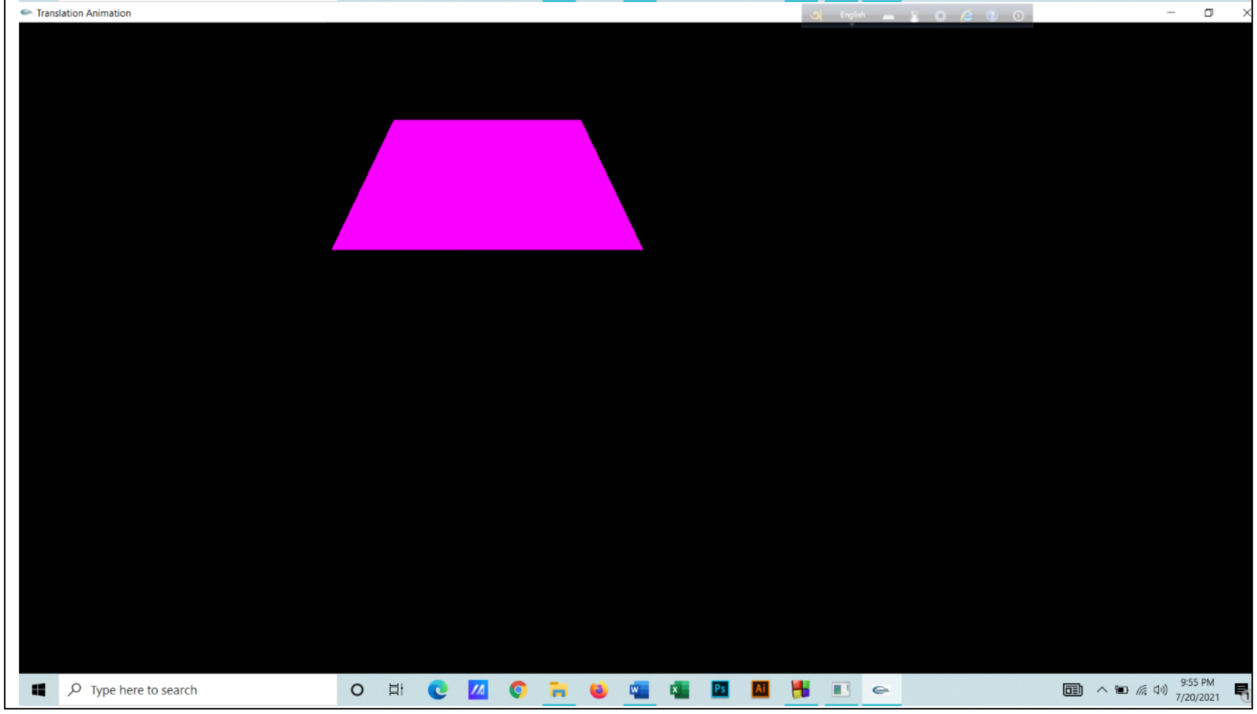
```
void display2(int val) {

 glutDisplayFunc(display3);


}
void display() {
  glClear(GL_COLOR_BUFFER_BIT);
  glLoadIdentity();



glPushMatrix();
glTranslatef(position,0.0f, 0.0f);

  glBegin(GL_TRIANGLES);
  glColor3ub(255, 243, 0 );
  glVertex2f(0.4f,-0.4f);
  glVertex2f(0.7f, 0.0f);
  glVertex2f(0.4f, 0.4f);
  glEnd();

glPopMatrix();

glutTimerFunc(1500,display2,0);
glFlush();

}

void dis()
{
     glutDisplayFunc(display);
}

int main(int argc, char** argv) {
  glutInit(&argc, argv);
  glutInitWindowSize(320, 320);
  glutInitWindowPosition(50, 50);
  glutCreateWindow("Translation Animation");
  glutDisplayFunc(dis);
  init();

  glutTimerFunc(100, update, 0);
    glutTimerFunc(100, update1, 0);
 glutTimerFunc(100, update2, 0);
   glutMainLoop();
   return 0;
}
```

Output Screenshot (Full Screen)-









Question-

Develop a code that will have four different objects (keep it simple). Four different keys will be dedicated each objects. The objects will move to the left, right, up and down in a loop as the keys are pressed individually.

Code-

```
#include<cstdio>

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


GLfloat position = 0.0f;
GLfloat position1 = 0.0f;

GLfloat speed = 0.1f;
void dis();
void display();

void update(int value) {

   if(position <-1.5)
      position = 1.0f;

   position -= speed;

         glutPostRedisplay();


         glutTimerFunc(100,update,0);
}


void update1(int value) {

   if(position1 >1.0)
      position1 = -1.0f;

   position1 += speed;

         glutPostRedisplay();


         glutTimerFunc(100,update1,0);
}


void init() {
  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
```

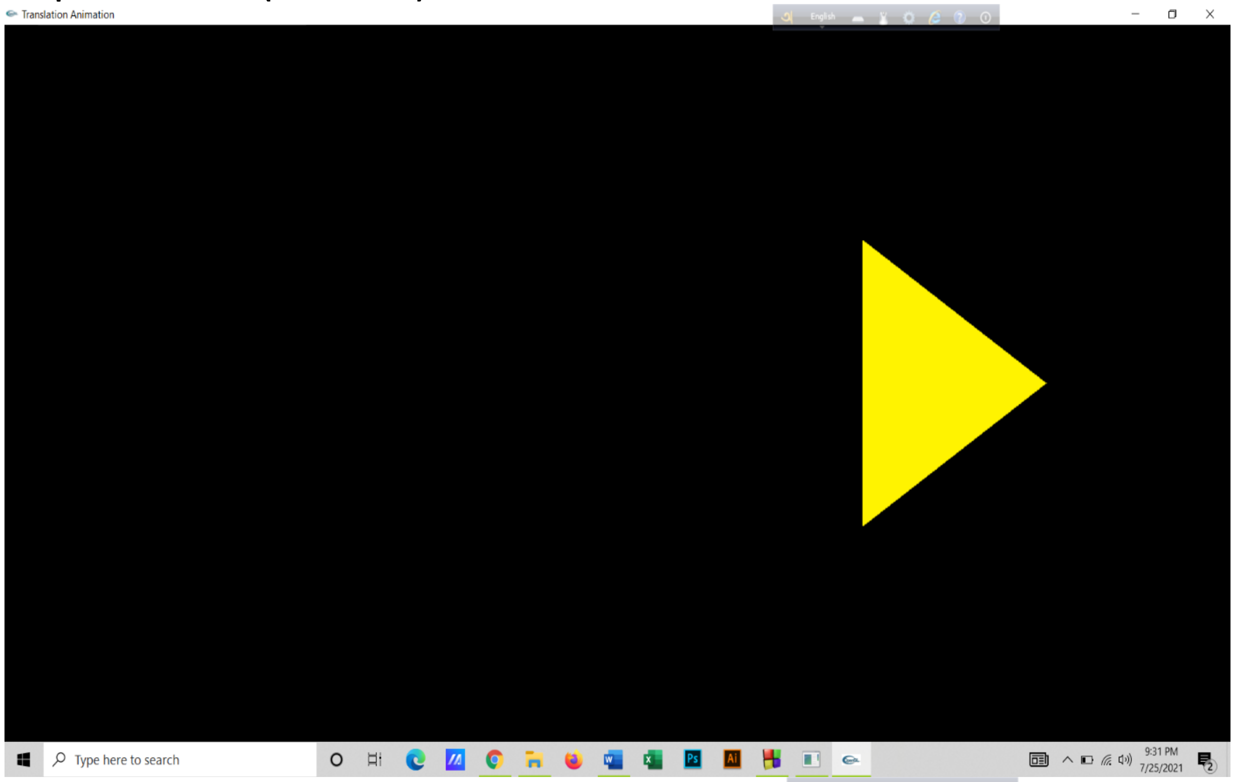
```
}

void disback(int val)
{
   glutDisplayFunc(display);
}

void display7()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glPushMatrix();
//glTranslatef(0.0f,position1, 0.0f);

   glBegin(GL_QUADS);           // These vertices form a closed polygon
         glColor3ub(248, 0, 255  );
         glVertex2f(-0.5f, 0.1f);
         glVertex2f(0.0f, 0.1f);
         glVertex2f(-0.1f, 0.5f);
   glVertex2f(-0.4f, 0.5f);
   glEnd();
  glPopMatrix();
  glutTimerFunc(1500,disback,0);
  glFlush();
}

void display6(int val) {

 glutDisplayFunc(display7);


}

void display5()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glPushMatrix();
//glTranslatef(0.0f,position, 0.0f);
    glBegin(GL_QUADS);           // These vertices form a closed polygon
         glColor3ub(0, 245, 255 );
         glVertex2f(0.2f, 0.2f);
         glVertex2f(0.4f, -0.2f);
         glVertex2f(0.6f, 0.2f);
    glVertex2f(0.4f, 0.6f);
    glEnd();
   glPopMatrix();
   glutTimerFunc(1500,display6,0);
   glFlush();

}

void display4(int val) {

 glutDisplayFunc(display5);



}

void display3()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
    glPushMatrix();
//glTranslatef(position1,0.0f, 0.0f);
   glBegin(GL_QUADS);
      glColor3f(0.0f, 1.0f, 0.0f);
      glVertex2f(-0.2f, -0.2f);
      glVertex2f( 0.2f, -0.2f);
      glVertex2f( 0.2f,  0.2f);
      glVertex2f(-0.2f,  0.2f);
   glEnd();
   glPopMatrix();
   glutTimerFunc(1500,display4,0);
   glFlush();
}

void display2(int val) {

 glutDisplayFunc(display3);



}
void display() {
   glClear(GL_COLOR_BUFFER_BIT);
   glLoadIdentity();
glPushMatrix();
//glTranslatef(position,0.0f, 0.0f);
```

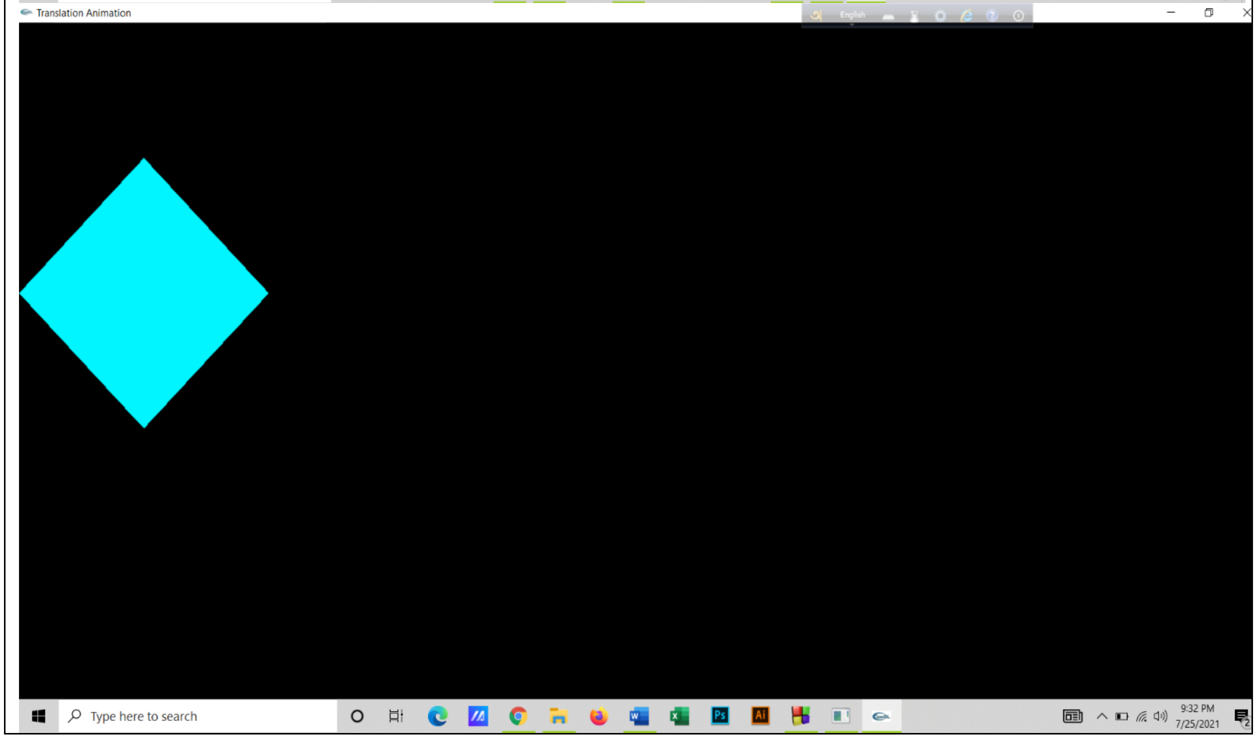
```
  glBegin(GL_TRIANGLES);
  glColor3ub(255, 243, 0 );
  glVertex2f(0.4f,-0.4f);
  glVertex2f(0.7f, 0.0f);
  glVertex2f(0.4f, 0.4f);
  glEnd();

glPopMatrix();

glutTimerFunc(1500,display2,0);
glFlush();

}

void dis()
{
     glutDisplayFunc(display);
}

void SpecialInput(int key, int x, int y)
{
switch(key)
{
case GLUT_KEY_UP:
//do something here

glTranslatef(0.0f,position1, 0.0f);

break;
case GLUT_KEY_DOWN:
glTranslatef(0.0f,position, 0.0f);
break;
case GLUT_KEY_LEFT:
  //glClearColor(1.0f, 1.0f, 1.0f, 1.0f);

glTranslatef(position,0.0f, 0.0f);
break;
case GLUT_KEY_RIGHT:
//do something here
  //glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
  glTranslatef(position1,0.0f, 0.0f);
break;
}
```

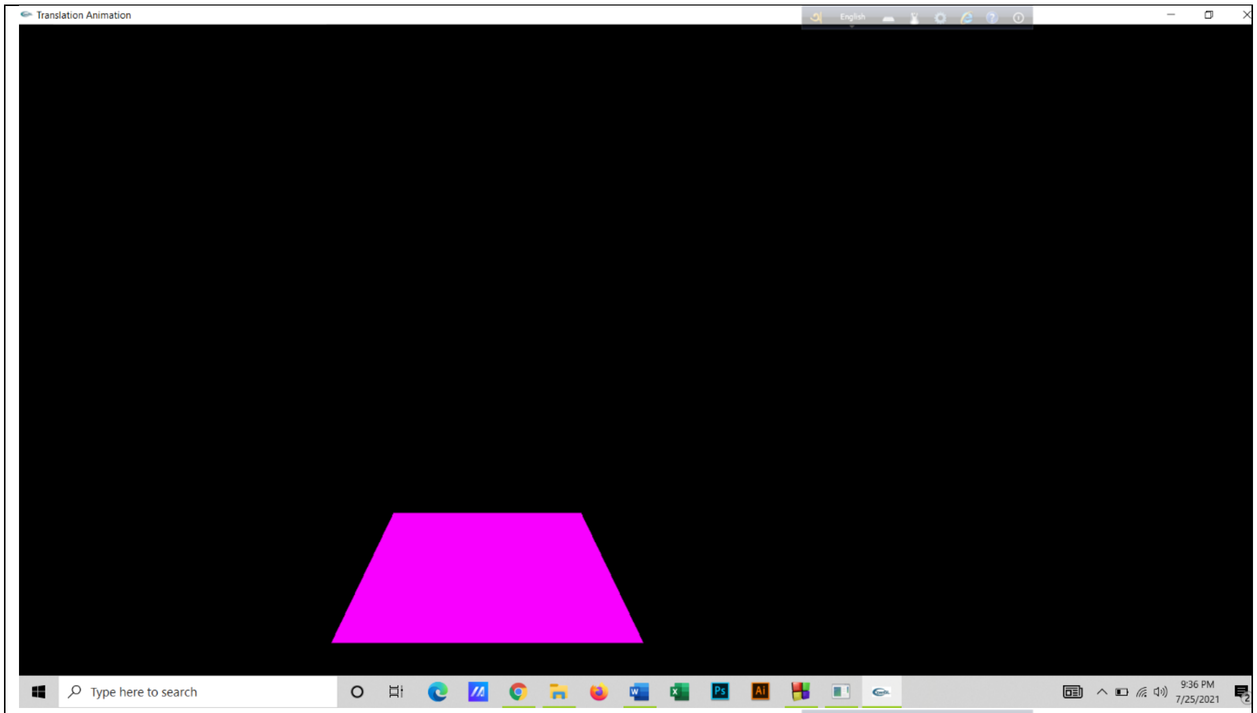
```
glutPostRedisplay();
}

void day()
{
   glClearColor(1.0,1.0,1.0,1.0);
glutPostRedisplay();
}
void night()
{
   glClearColor(0.0,0.0,0.0,1.0);
glutPostRedisplay();
}



void handleKeypress(unsigned char key, int x, int y) {

        switch (key) {

case 'd':

     glClearColor(1.0,1.0,1.0,1.0);
glutPostRedisplay();
  // glutDisplayFunc(day);
   //day();
   break;
case 'n':
 glClearColor(0.0,0.0,0.0,1.0);
glutPostRedisplay();
   break;

        }
}
int main(int argc, char** argv) {
  glutInit(&argc, argv);
  glutInitWindowSize(320, 320);
  glutInitWindowPosition(50, 50);
  glutCreateWindow("Translation Animation");
  glutDisplayFunc(dis);
  init();
glutSpecialFunc(SpecialInput);
  glutKeyboardFunc(handleKeypress);
  glutTimerFunc(100, update, 0);
```

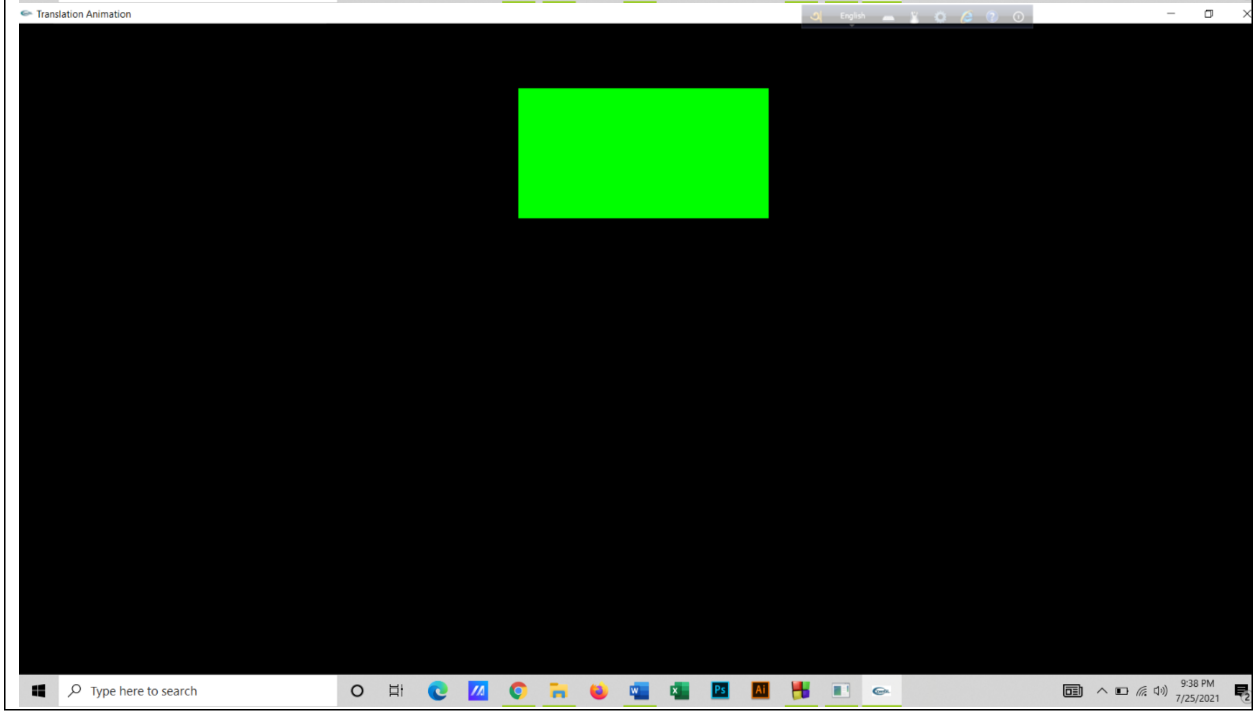
```
    glutTimerFunc(100, update1, 0);
  glutMainLoop();
  return 0;
}
```

Output Screenshot (Full Screen)-









Question-
Develop a code that will have four different objects (keep it simple). Two of the objects will move to the right as the right click is made on the mouse and two of the objects will move to the left as the left key is pressed on the mouse.

Code-

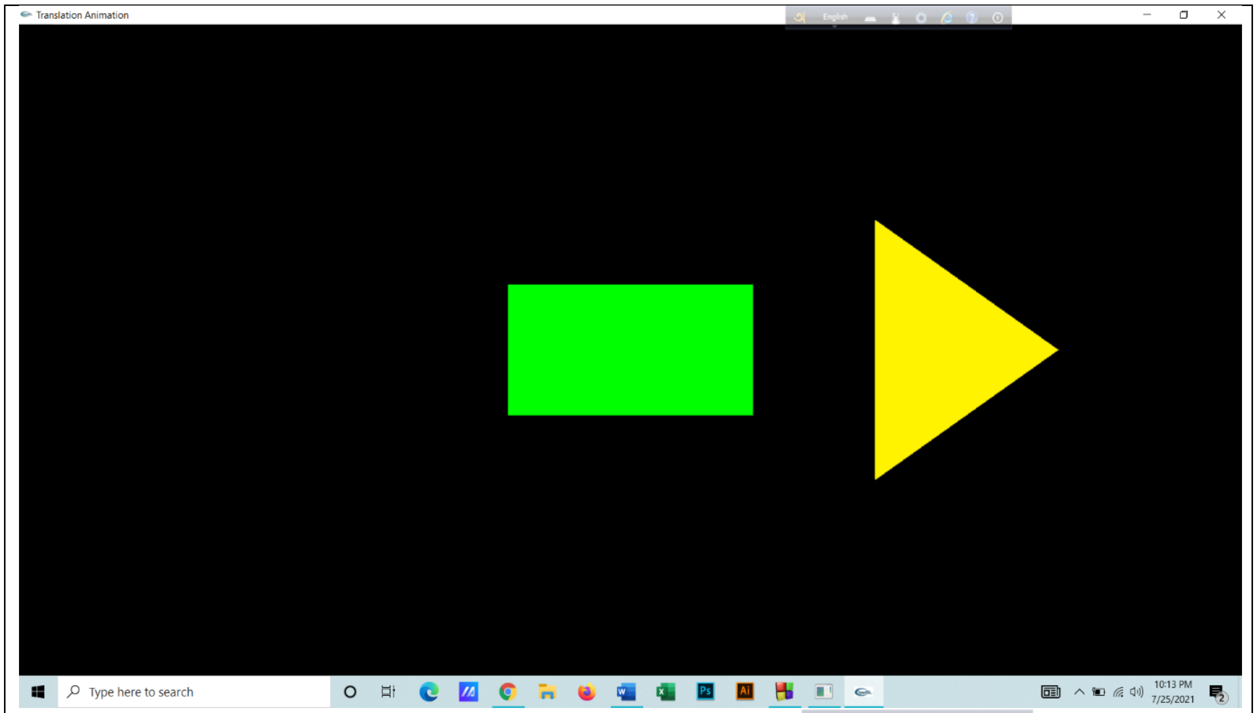
```
#include<cstdio>

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


GLfloat position = 0.0f;
GLfloat position1 = 0.0f;

GLfloat speed = 0.1f;
//void dis();
void display();

void update(int value) {

   if(position <-1.0)
      position = 1.0f;

   position -= speed;

          glutPostRedisplay();


          glutTimerFunc(100,update,0);
}


void update1(int value) {

   if(position1 >1.0)
      position1 = -1.0f;

   position1 += speed;

          glutPostRedisplay();


          glutTimerFunc(100,update1,0);
}


void init() {
  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
```

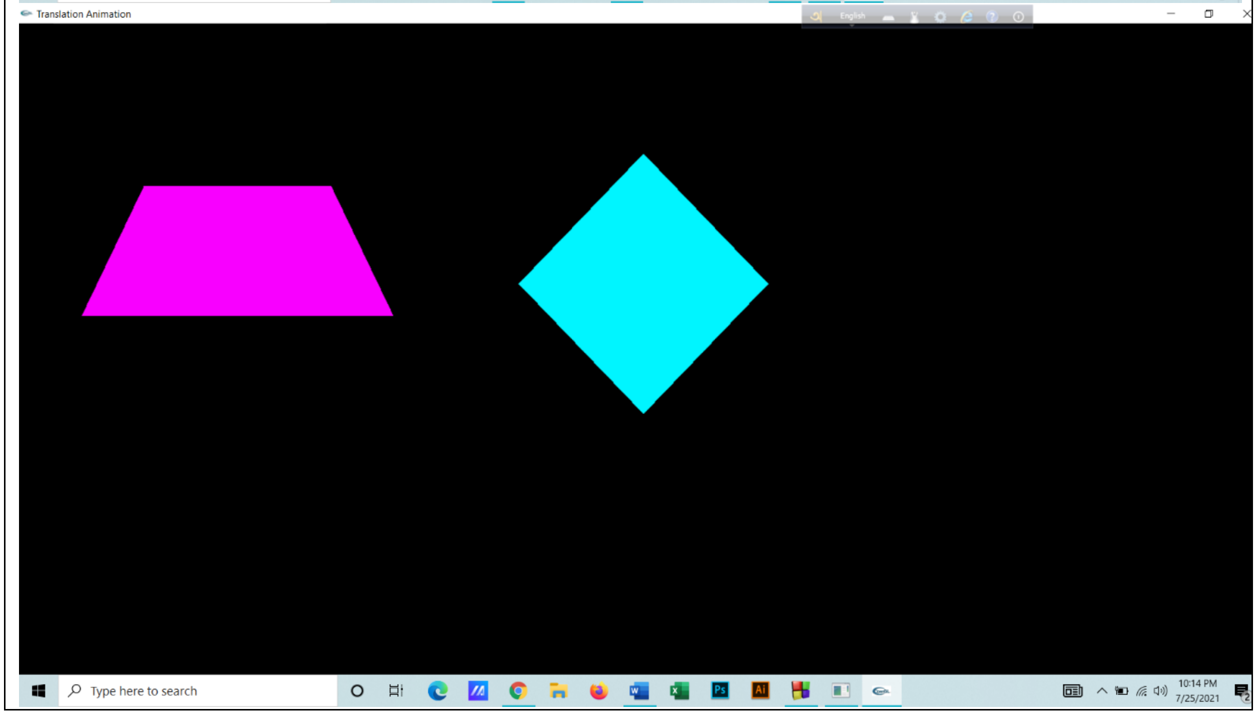
```
}

/*void disback(int val)
{
   glutDisplayFunc(display);
}
*/
void display6(int val) {

   glutDisplayFunc(display);

//glutTimerFunc(1500,display,0);

}

void display5()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glPushMatrix();
//glTranslatef(0.0f,position, 0.0f);
    glBegin(GL_QUADS);          // These vertices form a closed polygon
         glColor3ub(0, 245, 255 );//skyblue
         glVertex2f(0.2f, 0.2f);
         glVertex2f(0.4f, -0.2f);
         glVertex2f(0.6f, 0.2f);
   glVertex2f(0.4f, 0.6f);
   glEnd();
   glBegin(GL_QUADS);          // These vertices form a closed polygon
         glColor3ub(248, 0, 255  );//pink
         glVertex2f(-0.5f, 0.1f);
         glVertex2f(0.0f, 0.1f);
         glVertex2f(-0.1f, 0.5f);
   glVertex2f(-0.4f, 0.5f);
   glEnd();
  glPopMatrix();
  glutTimerFunc(1500,display6,0);
  glFlush();

}

void display2(int val) {

 glutDisplayFunc(display5);
```

```
}
void display() {
  glClear(GL_COLOR_BUFFER_BIT);
  glLoadIdentity();
glPushMatrix();
//glTranslatef(position,0.0f, 0.0f);

  glBegin(GL_TRIANGLES);
  glColor3ub(255, 243, 0 );//yellow
  glVertex2f(0.4f,-0.4f);
  glVertex2f(0.7f, 0.0f);
  glVertex2f(0.4f, 0.4f);
  glEnd();
   glBegin(GL_QUADS);
     glColor3f(0.0f, 1.0f, 0.0f);//green
     glVertex2f(-0.2f, -0.2f);
     glVertex2f( 0.2f, -0.2f);
     glVertex2f( 0.2f,  0.2f);
     glVertex2f(-0.2f,  0.2f);
  glEnd();
glPopMatrix();

glutTimerFunc(1500,display2,0);
glFlush();

}



void handleMouse(int button, int state, int x, int y) {
         if (button == GLUT_LEFT_BUTTON)
         {          glTranslatef(position,0.0f, 0.0f);



                                    }
if (button == GLUT_RIGHT_BUTTON)
         {
           glTranslatef(position1,0.0f, 0.0f);
          }
glutPostRedisplay();}
```

```
void handleKeypress(unsigned char key, int x, int y) {
        switch (key) {
case 'a':
   speed = 0.0f;
   break;
case 'w':
   speed = 0.1f;
   break;
glutPostRedisplay();
        }
        }




int main(int argc, char** argv) {
  glutInit(&argc, argv);
  glutInitWindowSize(320, 320);
  glutInitWindowPosition(50, 50);
  glutCreateWindow("Translation Animation");
  glutDisplayFunc(display);
  init();
  glutMouseFunc(handleMouse);
  glutKeyboardFunc(handleKeypress);

  glutTimerFunc(100, update, 0);
   glutTimerFunc(100, update1, 0);

  glutMainLoop();
  return 0;
}
```

Output Screenshot (Full Screen)-


Translation Animation
Type here to search
10:13 PM
7/25/2021


Translation Animation
Type here to search
10:14 PM
7/25/2021