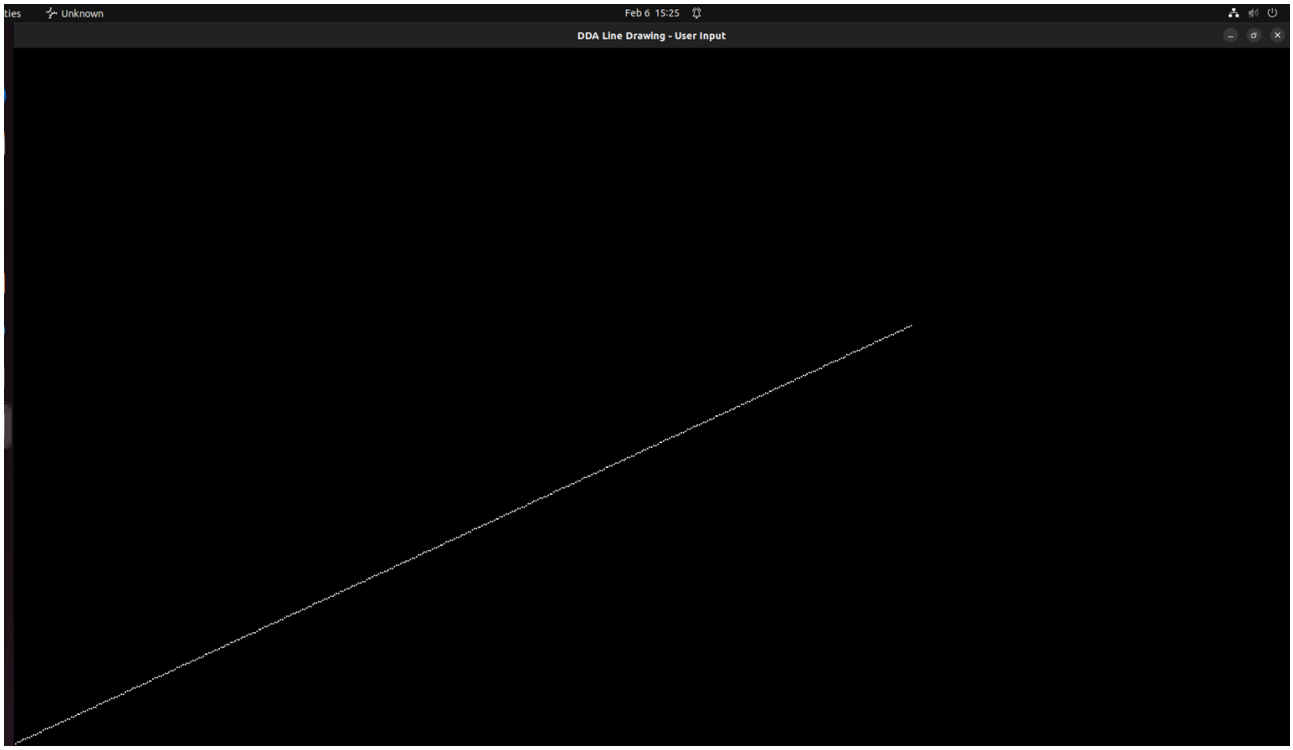
Lab - 3

Date - 06.02.2026

1) DDA algorithm

```
#include <GL/glut.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

int xStart, yStart, xEnd, yEnd;

void ddaLine(int x1, int y1, int x2, int y2)
{
   int dx = x2 - x1;
   int dy = y2 - y1;

   int steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy);

   float xInc = dx / (float)steps;
   float yInc = dy / (float)steps;

   float x = x1;
   float y = y1;

   glBegin(GL_POINTS);
   for (int i = 0; i <= steps; i++)
   {
      glVertex2i((int)round(x), (int)round(y));
      x += xInc;
      y += yInc;
   }
   glEnd();
}

void display()
{
   glClear(GL_COLOR_BUFFER_BIT);

   glColor3f(1.0, 1.0, 1.0);
   glPointSize(2.0);

   ddaLine(xStart, yStart, xEnd, yEnd);

   glFlush();
}

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

```
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0, 640, 0, 480);
}

int main(int argc, char** argv)
{
    printf("Enter x1 y1: ");
    scanf("%d %d", &xStart, &yStart);

    printf("Enter x2 y2: ");
    scanf("%d %d", &xEnd, &yEnd);

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(640, 480);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("DDA Line Drawing - User Input");

    init();
    glutDisplayFunc(display);
    glutMainLoop();

    return 0;
}
```



2) Bresenham's Line Generation Algorithm

```
#include <GL/glut.h>
#include <stdio.h>
#include <stdlib.h>

int xStart, yStart, xEnd, yEnd;

void bresenhamLine(int x1, int y1, int x2, int y2)
{
   int dx = abs(x2 - x1);
   int dy = abs(y2 - y1);

   int sx = (x1 < x2) ? 1 : -1;
   int sy = (y1 < y2) ? 1 : -1;

   int err = dx - dy;

   glBegin(GL_POINTS);
   while (1)
   {
      glVertex2i(x1, y1);

      if (x1 == x2 && y1 == y2)
         break;

      int e2 = 2 * err;

      if (e2 > -dy)
      {
         err -= dy;
         x1 += sx;
      }

      if (e2 < dx)
      {
         err += dx;
         y1 += sy;
      }
   }
   glEnd();
}

void display()
{
   glClear(GL_COLOR_BUFFER_BIT);

   glColor3f(1.0, 1.0, 1.0);
   glPointSize(2.0);
```

```
    bresenhamLine(xStart, yStart, xEnd, yEnd);

    glFlush();
}

void init()
{
    glClearColor(0.0, 0.0, 0.0, 1.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0, 640, 0, 480);
}

int main(int argc, char** argv)
{
    printf("Enter x1 y1: ");
    scanf("%d %d", &xStart, &yStart);

    printf("Enter x2 y2: ");
    scanf("%d %d", &xEnd, &yEnd);

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(640, 480);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Bresenham Line Drawing - User Input");

    init();
    glutDisplayFunc(display);
    glutMainLoop();

    return 0;
}
```


ties
Unknown
Feb 6 15:28
Bresenham Line Drawing - User Input

3) Implement line drawing algorithm to draw dotted line

```
#include <GL/glut.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

int xStart, yStart, xEnd, yEnd;

void ddaLineDotted(int x1, int y1, int x2, int y2)
{
   int dx = x2 - x1;
   int dy = y2 - y1;

   int steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy);

   float xInc = dx / (float)steps;
   float yInc = dy / (float)steps;

   float x = x1, y = y1;

   glBegin(GL_POINTS);
   for (int i = 0; i <= steps; i++)
   {
      if (i % 3 == 0)   // dotted pattern
         glVertex2i(round(x), round(y));

      x += xInc;
      y += yInc;
   }
   glEnd();
}

void display()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glPointSize(2.0);

   glColor3f(1.0, 1.0, 1.0);
   ddaLineDotted(xStart, yStart, xEnd, yEnd);

   glFlush();
}

void init()
{
   glClearColor(0.0, 0.0, 0.0, 1.0);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
```

```
    gluOrtho2D(0, 640, 0, 480);
}

int main(int argc, char** argv)
{
    printf("Enter x1 y1: ");
    scanf("%d %d", &xStart, &yStart);

    printf("Enter x2 y2: ");
    scanf("%d %d", &xEnd, &yEnd);

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(640, 480);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Dotted Line");

    init();
    glutDisplayFunc(display);
    glutMainLoop();

    return 0;
}
```

4) Implement line drawing algorithm to draw dashed-dotted line

```
#include <GL/glut.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

int xStart, yStart, xEnd, yEnd;

void ddaLineDashDot(int x1, int y1, int x2, int y2)
{
   int dx = x2 - x1;
   int dy = y2 - y1;

   int steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy);

   float xInc = dx / (float)steps;
   float yInc = dy / (float)steps;

   float x = x1, y = y1;

   glBegin(GL_POINTS);
   for (int i = 0; i <= steps; i++)
   {
      int pattern = i % 10;

      if (pattern < 4 || pattern == 6)
         glVertex2i(round(x), round(y));

      x += xInc;
      y += yInc;
   }
   glEnd();
}

void display()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glPointSize(2.0);

   glColor3f(1.0, 0.0, 0.0);
   ddaLineDashDot(xStart, yStart - 50, xEnd, yEnd - 50);

   glFlush();
}

void init()
{
```

```
    glClearColor(0.0, 0.0, 0.0, 1.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0, 640, 0, 480);
}

int main(int argc, char** argv)
{
    printf("Enter x1 y1: ");
    scanf("%d %d", &xStart, &yStart);

    printf("Enter x2 y2: ");
    scanf("%d %d", &xEnd, &yEnd);

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(640, 480);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Dash-Dotted Line");

    init();
    glutDisplayFunc(display);
    glutMainLoop();

    return 0;
}
```

5) Use any Line Drawing algorithm, print HELLO.

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

void drawLine(float x1, float y1, float x2, float y2)
{
   glBegin(GL_LINES);
      glVertex2f(x1, y1);
      glVertex2f(x2, y2);
   glEnd();
}

void display()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glColor3f(1.0, 1.0, 1.0); // white color

   // H
   drawLine(50, 50, 50, 150);
   drawLine(100, 50, 100, 150);
   drawLine(50, 100, 100, 100);

   // E
   drawLine(120, 50, 120, 150);
   drawLine(120, 150, 170, 150);
   drawLine(120, 100, 160, 100);
   drawLine(120, 50, 170, 50);

   // L
   drawLine(190, 50, 190, 150);
   drawLine(190, 50, 240, 50);

   // L
   drawLine(260, 50, 260, 150);
   drawLine(260, 50, 310, 50);

   // O
   drawLine(330, 50, 380, 50);
   drawLine(380, 50, 380, 150);
   drawLine(380, 150, 330, 150);
   drawLine(330, 150, 330, 50);

   glFlush();
}

void init()
{
   glClearColor(0.0, 0.0, 0.0, 0.0);
   glMatrixMode(GL_PROJECTION);
```

```
    glLoadIdentity();
    gluOrtho2D(0, 500, 0, 200);
}

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(600, 300);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("HELLO using OpenGL Line Drawing");

    init();
    glutDisplayFunc(display);
    glutMainLoop();

    return 0;
}
```