**PRAKTIKUM GRAFIKA KOMPUTER**

Pertemuan ke-3 (Kurva)



**Yoga Agustiansyah**2206050

Jurusan Ilmu Komputer

Program Studi Teknik Informatika

Institut Teknologi Garut

Jl. Mayor Syamsu No. 1 Jayaraga Garut 44151 Indonesia

# **I. PENDAHULUAN**

Dalam lingkungan OpenGL, sebuah kurva adalah representasi visual dari data atau fungsi yang terdiri dari serangkaian titik atau segmen garis. Dalam praktikum ini, kurva digunakan untuk memvisualisasikan berbagai bentuk fungsi matematika dengan menggunakan mode gambar OpenGL, seperti GL\_POINTS, GL\_LINES, GL\_LINE\_STRIP, dan GL\_LINE\_LOOP. Setiap mode memiliki efek visual yang berbeda terhadap tampilan kurva yang dihasilkan.

# **II. PEMBAHASAN**

## **A. Program asal menggunakan GL\_POINTS**

**1. Source Code**

**a) Program 1 GL\_POINTS**

|  |
| --- |
| #include <GL/glut.h>  void display(void) {  glClearColor(0.078, 0.178, 0.294, 1.0);  glClear(GL\_COLOR\_BUFFER\_BIT);  float t = 0.0;  glColor3f(0.0, 0.850, 1);  glBegin(GL\_POINTS);  for (t = -1.0; t <= 1.0; t += 0.01) {  glVertex3f(t, -0.5 + t \* t, 0.0);  }  glEnd();  glFlush();  }  void kunci(unsigned char key, int x, int y) {  switch (key) {  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char \*argv[]) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah - 2206050 (Program 1)");  glutDisplayFunc(display);  glutKeyboardFunc(kunci);  glutMainLoop();  return 0;  } |

**b) Program 2 GL\_POINTS**

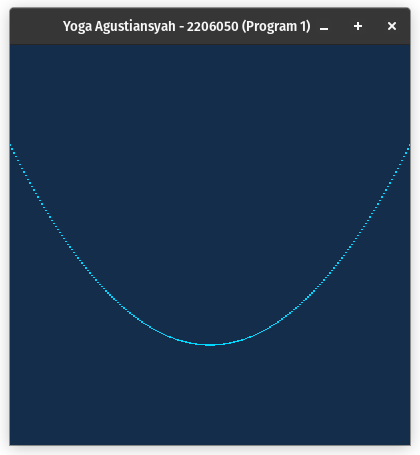
|  |
| --- |
| #include <GL/glut.h>  void myinit() {  glClearColor(0.078, 0.178, 0.294, 1.0);  glColor3f(1.0, 0.0, 0.0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-10.0, 10.0, -10.0, 10.0);  glMatrixMode(GL\_MODELVIEW);  }  void display(void) {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0.0, 0.850, 1);  float t = 0.0;  //f(x)=1/14(x+4)(x+1)(x-1)(x-3)+0.5  glPointSize(2);  glBegin(GL\_POINTS);  for (t = -10.0; t <= 10.0; t += 0.1) {  glVertex3f(t, (t + 4) \* (t + 1) \* (t - 1) \* (t - 3) / 14 + 0.5, 0.0);  }  glEnd();  glBegin(GL\_LINES);  glVertex3f(-10.0, 0.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);  glVertex3f(0.0, -10.0, 0.0);  glVertex3f(0.0, 10.0, 0.0);  glEnd();  glFlush();  }  void kunci(unsigned char key, int x, int y) {  switch (key) {  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char \*argv[]) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah – 2206050 (Program 2)”);  glutDisplayFunc(display);  myinit();  glutKeyboardFunc(kunci);  glutMainLoop();  return 0;  } |

**c) Program 3 GL\_POINTS**

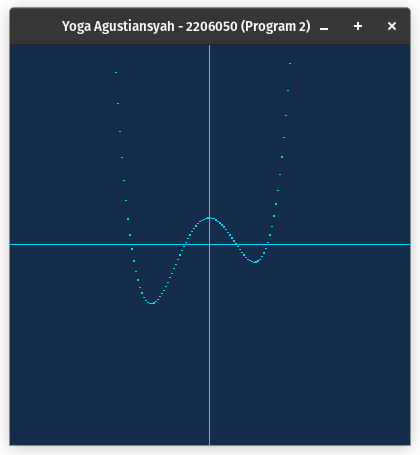
|  |
| --- |
| #include <GL/glut.h>  #include <math.h>  void myInit() {  glClearColor(0.078, 0.178, 0.294, 1.0);  glColor3f(1, 0, 0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-1.0, 10.5, -2.0, 2.0);  glMatrixMode(GL\_MODELVIEW);  }  void display(void) {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0.0, 0.850, 1);  float x = 0.0;  glBegin(GL\_POINTS);  for(x = 0.0; x <= 1.5\*6.28; x+=0.01){  glVertex2f(x, sin(x));  }  glEnd();  glColor3f(0.0, 1.0, 0.45);  glLineWidth(2);  glBegin(GL\_LINES);  glVertex3f(-10.0, 0.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);  glVertex3f(0.0, -10.0, 0.0);  glVertex3f(0.0, 10.0, 0.0);  glVertex3f(0.0, 0.0, 10.0);  glVertex3f(0.0, 0.0, -10.0);  glEnd();  glFlush();  }  void kunci (unsigned char key, int x, int y){  switch(key){  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char\* argv[]) {  glutInit (&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah - 2206050 (Program 3)");  glutDisplayFunc(display);  glutKeyboardFunc(kunci);  myInit();  glutMainLoop();  return 0;  } |

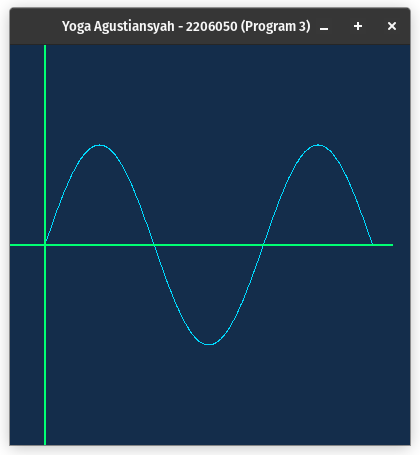
**2. Output**

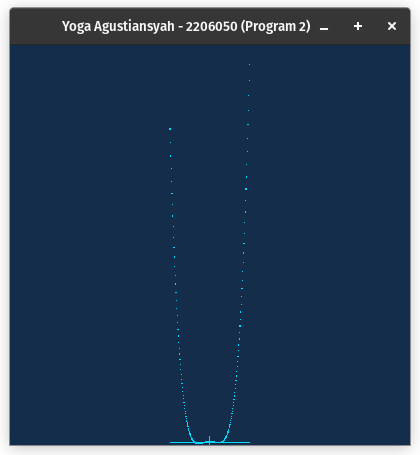
**a) Program 1**



**2) Program 2**



**Program 2 jika dilihat dari ujung ke ujung**



**3) Program 3**

**3, Penjelasan**

Dengan menggunakan **glBegin(GL\_POINTS)**, program akan menggambar setiap titik yang telah ditentukan oleh **glVertex3f**. Hasilnya adalah kurva yang terdiri dari banyak titik individual yang mewakili nilai fungsi pada interval titik koordinat tertentu.

## **B. Program menggunakan GL\_LINES**

**1. Source Code**

**a) Program 1 GL\_LINES**

|  |
| --- |
| #include <GL/glut.h>  void display(void) {  glClearColor(0.078, 0.178, 0.294, 1.0);  glClear(GL\_COLOR\_BUFFER\_BIT);  float t = 0.0;  glColor3f(0.0, 0.850, 1);    glBegin(GL\_LINES);  for (t = -1.0; t <= 1.0; t += 0.01) {  glVertex3f(t, -0.5 + t \* t, 0.0);  }  glEnd();  glFlush();  }  void kunci(unsigned char key, int x, int y) {  switch (key) {  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char \*argv[]) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah - 2206050 (Program 1)");  glutDisplayFunc(display);  glutKeyboardFunc(kunci);  glutMainLoop();  return 0;  } |

**b) Program 2 GL\_LINES**

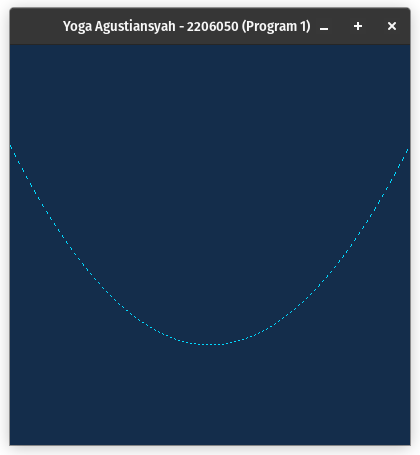
|  |
| --- |
| #include <GL/glut.h>  void myinit() {  glClearColor(0.078, 0.178, 0.294, 1.0);  glColor3f(1.0, 0.0, 0.0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-10.0, 10.0, -10.0, 10.0);  glMatrixMode(GL\_MODELVIEW);  }  void display(void) {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0.0, 0.850, 1);  float t = 0.0;  //f(x)=1/14(x+4)(x+1)(x-1)(x-3)+0.5  glBegin(GL\_LINES);  for (t = -10.0; t <= 10.0; t += 0.1) {  glVertex3f(t, (t + 4) \* (t + 1) \* (t - 1) \* (t - 3) / 14 + 0.5, 0.0);  }  glEnd();  glBegin(GL\_LINES);  glVertex3f(-10.0, 0.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);  glVertex3f(0.0, -10.0, 0.0);  glVertex3f(0.0, 10.0, 0.0);  glEnd();  glFlush();  }  void kunci(unsigned char key, int x, int y) {  switch (key) {  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char \*argv[]) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah – 2206050 (Program 2)”);  glutDisplayFunc(display);  myinit();  glutKeyboardFunc(kunci);  glutMainLoop();  return 0;  } |

**c) Program 3 GL\_LINES**

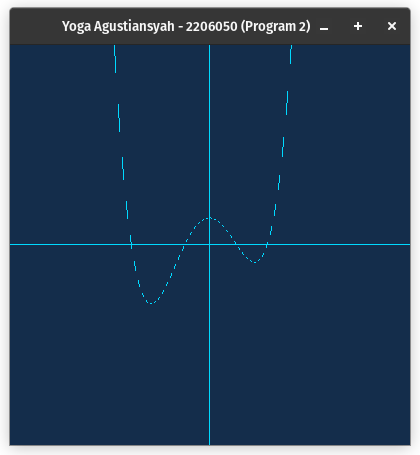
|  |
| --- |
| #include <GL/glut.h>  #include <math.h>  void myInit() {  glClearColor(0.078, 0.178, 0.294, 1.0);  glColor3f(1, 0, 0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-1.0, 10.5, -2.0, 2.0);  glMatrixMode(GL\_MODELVIEW);  }  void display(void) {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0.0, 0.850, 1);  float x = 0.0;  glBegin(GL\_LINES);  for(x = 0.0; x <= 1.5\*6.28; x+=0.01){  glVertex2f(x, sin(x));  }  glEnd();  glColor3f(0.0, 1.0, 0.45);  glLineWidth(2);  glBegin(GL\_LINES);  glVertex3f(-10.0, 0.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);  glVertex3f(0.0, -10.0, 0.0);  glVertex3f(0.0, 10.0, 0.0);  glVertex3f(0.0, 0.0, 10.0);  glVertex3f(0.0, 0.0, -10.0);  glEnd();  glFlush();  }  void kunci (unsigned char key, int x, int y){  switch(key){  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char\* argv[]) {  glutInit (&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah - 2206050 (Program 3)");  glutDisplayFunc(display);  glutKeyboardFunc(kunci);  myInit();  glutMainLoop();  return 0;  } |

**2. Output**

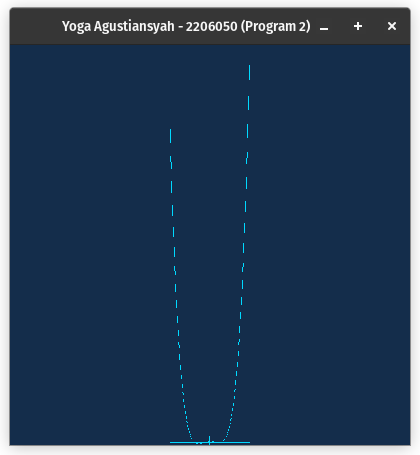
**a) Program 1**



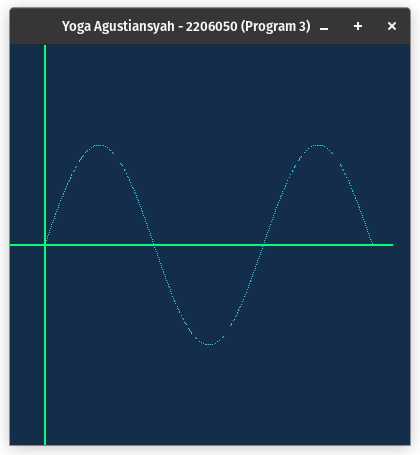
**2) Program 2**



**Program 2 jika dilihat dari ujung ke ujung**

****

**3) Program 3**

****

**3. Penjelasan**

Jika menggunakan **glBegin(GL\_LINES)**, program akan menghubungkan setiap dua titik yang ditentukan menggunakan **glVertex3f** dengan garis lurus. Ini akan menghasilkan serangkaian garis lurus yang menghubungkan titik-titik tersebut, menghasilkan kurva dengan sudut-sudut tajam di antara garis-garis tersebut.

## **C. Program menggunakan GL\_LINE\_STRIP**

**1. Source Code**

**a) Program 1 dengan GL\_LINE\_STRIP**

|  |
| --- |
| #include <GL/glut.h>  void display(void) {  glClearColor(0.078, 0.178, 0.294, 1.0);  glClear(GL\_COLOR\_BUFFER\_BIT);  float t = 0.0;  glColor3f(0.0, 0.850, 1);    glBegin(GL\_LINE\_STRIP);  for (t = -1.0; t <= 1.0; t += 0.01) {  glVertex3f(t, -0.5 + t \* t, 0.0);  }  glEnd();  glFlush();  }  void kunci(unsigned char key, int x, int y) {  switch (key) {  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char \*argv[]) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah - 2206050 (Program 1)");  glutDisplayFunc(display);  glutKeyboardFunc(kunci);  glutMainLoop();  return 0;  } |

**b) Program 2 GL\_LINE\_STRIP**

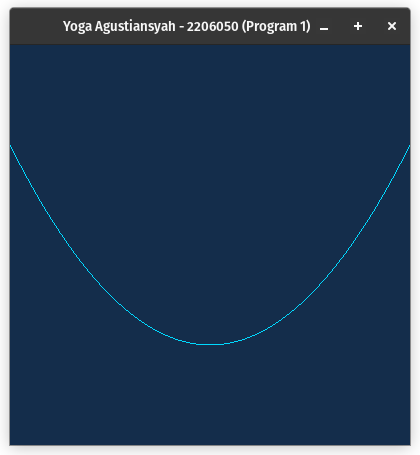
|  |
| --- |
| #include <GL/glut.h>  void myinit() {  glClearColor(0.078, 0.178, 0.294, 1.0);  glColor3f(1.0, 0.0, 0.0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-10.0, 10.0, -10.0, 10.0);  glMatrixMode(GL\_MODELVIEW);  }  void display(void) {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0.0, 0.850, 1);  float t = 0.0;  //f(x)=1/14(x+4)(x+1)(x-1)(x-3)+0.5  glBegin(GL\_LINE\_STRIP);  for (t = -10.0; t <= 10.0; t += 0.1) {  glVertex3f(t, (t + 4) \* (t + 1) \* (t - 1) \* (t - 3) / 14 + 0.5, 0.0);  }  glEnd();  glBegin(GL\_LINES);  glVertex3f(-10.0, 0.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);  glVertex3f(0.0, -10.0, 0.0);  glVertex3f(0.0, 10.0, 0.0);  glEnd();  glFlush();  }  void kunci(unsigned char key, int x, int y) {  switch (key) {  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char \*argv[]) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah – 2206050 (Program 2)”);  glutDisplayFunc(display);  myinit();  glutKeyboardFunc(kunci);  glutMainLoop();  return 0;  } |

**c) Program 3 GL\_LINE\_STRIP**

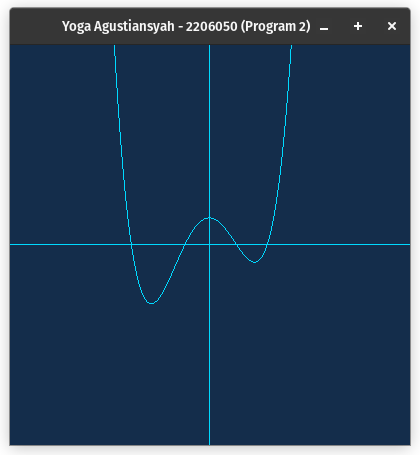
|  |
| --- |
| #include <GL/glut.h>  #include <math.h>  void myInit() {  glClearColor(0.078, 0.178, 0.294, 1.0);  glColor3f(1, 0, 0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-1.0, 10.5, -2.0, 2.0);  glMatrixMode(GL\_MODELVIEW);  }  void display(void) {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0.0, 0.850, 1);  float x = 0.0;  glBegin(GL\_LINE\_STRIP);  for(x = 0.0; x <= 1.5\*6.28; x+=0.01){  glVertex2f(x, sin(x));  }  glEnd();  glColor3f(0.0, 1.0, 0.45);  glLineWidth(2);  glBegin(GL\_LINES);  glVertex3f(-10.0, 0.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);  glVertex3f(0.0, -10.0, 0.0);  glVertex3f(0.0, 10.0, 0.0);  glVertex3f(0.0, 0.0, 10.0);  glVertex3f(0.0, 0.0, -10.0);  glEnd();  glFlush();  }  void kunci (unsigned char key, int x, int y){  switch(key){  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char\* argv[]) {  glutInit (&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah - 2206050 (Program 3)");  glutDisplayFunc(display);  glutKeyboardFunc(kunci);  myInit();  glutMainLoop();  return 0;  } |

**2. Output**

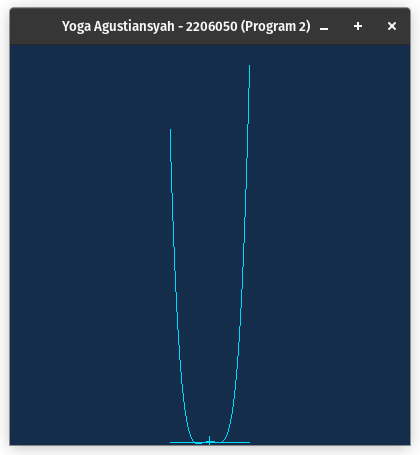
**a) Program 1**



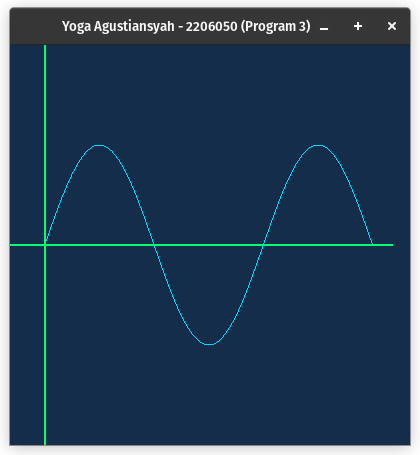
**2) Program 2**



**Program 2 jika dilihat dari ujung ke ujung**

****

**3) Program 3**

****

**3. Penjelasan**

Dalam mode **GL\_LINE\_STRIP**, program juga menghubungkan titik-titik yang ditentuksn dengan garis lurus, tetapi bedanya, kita hanya perlu mendefinisikan titik awal sekali. Setelah itu, setiap titik berikutnya secara otomatis terhubung ke titik sebelumnya. Ini akan menghasilkan kurva yang lebih mulus daripada GL\_LINES.

## **D. Program menggunakan GL\_LINE\_LOOP**

**1. Source Code**

**a) Program 1 dengan GL\_LINE\_LOOP**

|  |
| --- |
| #include <GL/glut.h>  void display(void) {  glClearColor(0.078, 0.178, 0.294, 1.0);  glClear(GL\_COLOR\_BUFFER\_BIT);  float t = 0.0;  glColor3f(0.0, 0.850, 1);    glBegin(GL\_LINE\_LOOP);  for (t = -1.0; t <= 1.0; t += 0.01) {  glVertex3f(t, -0.5 + t \* t, 0.0);  }  glEnd();  glFlush();  }  void kunci(unsigned char key, int x, int y) {  switch (key) {  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char \*argv[]) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah - 2206050 (Program 1)");  glutDisplayFunc(display);  glutKeyboardFunc(kunci);  glutMainLoop();  return 0;  } |

**b) Program 2 GL\_LINE\_LOOP**

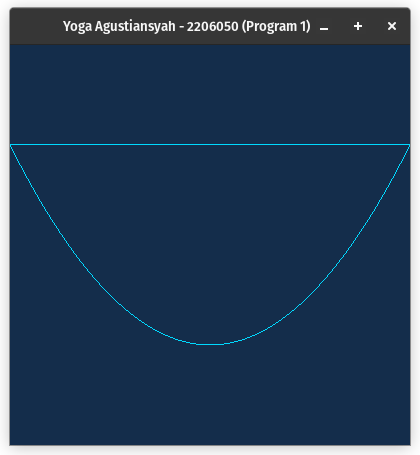
|  |
| --- |
| #include <GL/glut.h>  void myinit() {  glClearColor(0.078, 0.178, 0.294, 1.0);  glColor3f(1.0, 0.0, 0.0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-10.0, 10.0, -10.0, 10.0);  glMatrixMode(GL\_MODELVIEW);  }  void display(void) {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0.0, 0.850, 1);  float t = 0.0;  //f(x)=1/14(x+4)(x+1)(x-1)(x-3)+0.5  glBegin(GL\_LINE\_LOOP);  for (t = -10.0; t <= 10.0; t += 0.1) {  glVertex3f(t, (t + 4) \* (t + 1) \* (t - 1) \* (t - 3) / 14 + 0.5, 0.0);  }  glEnd();  glBegin(GL\_LINES);  glVertex3f(-10.0, 0.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);  glVertex3f(0.0, -10.0, 0.0);  glVertex3f(0.0, 10.0, 0.0);  glEnd();  glFlush();  }  void kunci(unsigned char key, int x, int y) {  switch (key) {  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char \*argv[]) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah – 2206050 (Program 2)”);  glutDisplayFunc(display);  myinit();  glutKeyboardFunc(kunci);  glutMainLoop();  return 0;  } |

**c) Program 3 GL\_LINE\_LOOP**

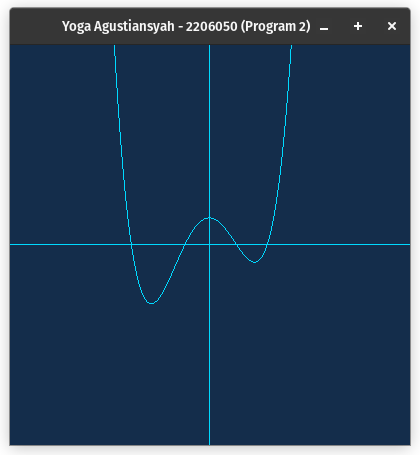
|  |
| --- |
| #include <GL/glut.h>  #include <math.h>  void myInit() {  glClearColor(0.078, 0.178, 0.294, 1.0);  glColor3f(1, 0, 0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-1.0, 10.5, -2.0, 2.0);  glMatrixMode(GL\_MODELVIEW);  }  void display(void) {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0.0, 0.850, 1);  float x = 0.0;  glBegin(GL\_LINE\_LOOP);  for(x = 0.0; x <= 1.5\*6.28; x+=0.01){  glVertex2f(x, sin(x));  }  glEnd();  glColor3f(0.0, 1.0, 0.45);  glLineWidth(2);  glBegin(GL\_LINES);  glVertex3f(-10.0, 0.0, 0.0);  glVertex3f(10.0, 0.0, 0.0);  glVertex3f(0.0, -10.0, 0.0);  glVertex3f(0.0, 10.0, 0.0);  glVertex3f(0.0, 0.0, 10.0);  glVertex3f(0.0, 0.0, -10.0);  glEnd();  glFlush();  }  void kunci (unsigned char key, int x, int y){  switch(key){  case 27:  case 'q':  exit(0);  break;  }  glutPostRedisplay();  }  int main(int argc, char\* argv[]) {  glutInit (&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Yoga Agustiansyah - 2206050 (Program 3)");  glutDisplayFunc(display);  glutKeyboardFunc(kunci);  myInit();  glutMainLoop();  return 0;  } |

**2. Output**

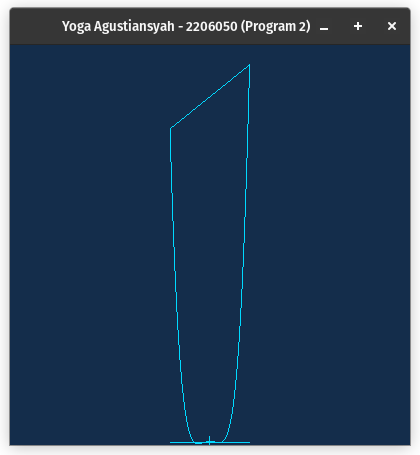
**a) Program 1**



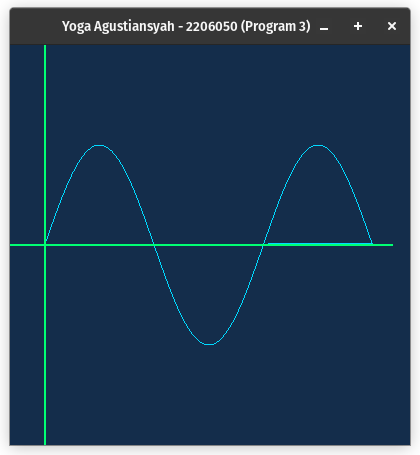
**2) Program 2**

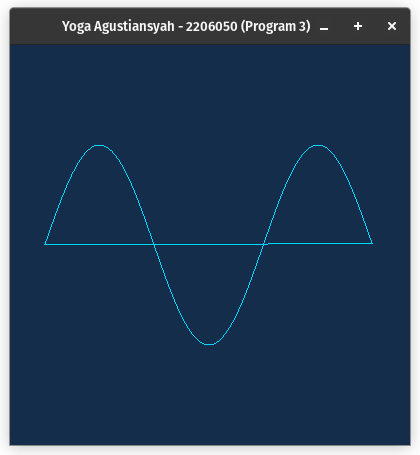


**Program 2 jika dilihat dari ujung ke ujung**

****

**3) Program 3**

 **Program 3 jika tanpa garis kartesius**

****

**3. Penjelasan**

Mode **GL\_LINE\_LOOP** ini mirip dengan **GL\_LINE\_STRIP**, hanya saja, mode juga menghubungkan titik terakhir kembali ke titik awal, sehingga kita akan mendapatkan kurva tertutup. Ini berguna jika kita ingin menggambar kurva yang kembali ke titik awal.

# **III. Kesimpulan**

Dalam praktikum ini, telah dijelaskan dan diuji empat mode yang berbeda dalam OpenGL untuk menggambar kurva, yaitu GL\_POINTS, GL\_LINES, GL\_LINE\_STRIP, dan GL\_LINE\_LOOP. Setiap mode memiliki efek visual yang berbeda pada tampilan kurva yang dihasilkan. Pemilihan mode tergantung pada tujuan visual yang ingin dicapai. Mode GL\_POINTS cocok untuk titik-titik individual, GL\_LINES untuk sudut tajam, GL\_LINE\_STRIP untuk kurva mulus, dan GL\_LINE\_LOOP untuk kurva tertutup. Dengan pemahaman ini, menggambar kurva dapat lebih fleksibel sesuai dengan kebutuhan kita.

Menurut pandangan pribadi saya, mode GL\_LINE\_STRIP adalah pilihan yang sangat cocok karena menghasilkan kurva yang lebih halus dan lebih bagus dalam dari segi visual. Sedangkan mode GL\_LINES, yang menghasilkan garis putus-putus, mungkin sebaiknya dihindari dalam pembuatan kurva, terutama jika tujuan adalah visualisasi yang mulus dan kontinu, karena garis putus-putus tidak lazim digunakan dalam pembuatan garis kurva. Namun, pemilihan mode selalu bergantung pada kebutuhan dan preferensi spesifik dalam penggambaran kurva.