**SSN COLLEGE OF ENGINEERING, KALAVAKKAM**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**UCS1712 – GRAPHICS AND MULTIMEDIA LAB**

**------------------------------------------------------------------------------------------------------------ Lab Exercise 7 : Cohen Sutherland Line clipping in C++ using OpenGL**

**Aim:**

* Apply Cohen Sutherland line clipping on a line (x1,y1) (x2,y2) with respect to a clipping window (XWmin,YWmin) (XWmax,YWmax).
* After clipping with respect to an edge, display the line segment with the calculated intermediate intersection points and the vertex list.
* Input: The clipping window co-ordinates and the line endpoints

Source Code :

| #include<GLUT/glut.h> #include<iostream> #include<string> #include<cmath> using namespace std; const double PI=3.14159265; void drawString(float x, float y, const char \*string) {  glRasterPos2f(x, y);  for (const char \*c = string; \*c != '\0'; c++) {  glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_12, \*c);  } } void myInit() {  glClearColor(1.0,0.6,0.5,0.0);  glPointSize(2);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(0,480.0,0,480.0); } void myDisplay() {  glClear(GL\_COLOR\_BUFFER\_BIT);  glColor3f(0, 0, 0);  glBegin(GL\_LINES);  glVertex2f(0,240);  glVertex2f(0,-240);  glVertex2f(240,0);  glVertex2f(-240,0);  glEnd();  int xmin,xmax,ymin,ymax,ch,rows,cols=4;  float x,y,x1,y1;  cout<<"Enter number of lines :";  cin >> rows;  int \*\*matrix = new int\*[rows],\*\*bmatrix = new int\*[rows];  for (int i = 0; i < rows; i++) {  matrix[i] = new int[cols];  bmatrix[i] = new int[2];  }  for(int i = 0;i<rows;i++){  cout<<"Enter point 1(x1,y1) :";  cin >> matrix[i][0] >> matrix[i][1];  cout<<"Enter point 2(x2,y2) :";  cin >> matrix[i][2] >> matrix[i][3];  }  cout<<"Enter window xmin, xmax,ymin,ymax :";  cin >> xmin >> xmax >> ymin >> ymax;  glColor3f(0.0,0.0,0.0);  glBegin(GL\_LINE\_LOOP);  glVertex2f(xmin,ymin);  glVertex2f(xmin,ymax);  glVertex2f(xmax,ymax);  glVertex2f(xmax,ymin);  glEnd();  glColor3f(0.0,0.0,1.0);  for(int i=0;i<rows;i++){  glBegin(GL\_LINE\_STRIP);  glVertex2f(matrix[i][0],matrix[i][1]);  glVertex2f(matrix[i][2],matrix[i][3]);  glEnd();  for (int j=0;j<2;j++){  if(matrix[i][0+2\*j]<xmin){  if(matrix[i][1+2\*j]<ymin){  bmatrix[i][0+j] = 5;  }  else if (matrix[i][1+2\*j]>ymax){  bmatrix[i][0+j] = 9;  }  else{  bmatrix[i][0+j] = 1;  }  }  else if(matrix[i][0+2\*j]>xmax){  if(matrix[i][1+2\*j]<ymin){  bmatrix[i][0+j] = 6;  }  else if (matrix[i][1+2\*j]>ymax){  bmatrix[i][0+j] = 10;  }  else{  bmatrix[i][0+j] = 2;  }  }  else{  if(matrix[i][1+2\*j]<ymin){  bmatrix[i][0+j] = 4;  }  else if (matrix[i][1+2\*j]>ymax){  bmatrix[i][0+j] = 8;  }  else{  bmatrix[i][0+j] = 0;  }  }  }  }  glColor3f(1.0,0.0,0.0);  for(int i=0;i<rows;i++){  if(bmatrix[i][0]==0 && bmatrix[i][1]==0){  glBegin(GL\_LINE\_STRIP);  glVertex2f(matrix[i][0],matrix[i][1]);  glVertex2f(matrix[i][2],matrix[i][3]);  glEnd();  }  else if((bmatrix[i][0]&bmatrix[i][1])!=0){  continue;  }  else{  float m = static\_cast<float>(matrix[i][3]-matrix[i][1])/static\_cast<float>(matrix[i][2]-matrix[i][0]);  if(bmatrix[i][0]==8){  y=ymax;  x=matrix[i][0]+(y-matrix[i][1])/m;  }  else if(bmatrix[i][0]==9){  y=ymax;  x=matrix[i][0]+(y-matrix[i][1])/m;    if(x>=xmin && x<=xmax){  }  else{  x= xmin;  y = matrix[i][1]+m\*(x-matrix[i][0]);  }  }  else if(bmatrix[i][0]==10){  y=ymax;  x=matrix[i][0]+(y-matrix[i][1])/m;  if(x>=xmin && x<=xmax){  }  else{  x= xmax;  y = matrix[i][1]+m\*(x-matrix[i][0]);  }  }  else if(bmatrix[i][0]==4){  y=ymin;  x=matrix[i][0]+(y-matrix[i][1])/m;  }  else if(bmatrix[i][0]==5){  y=ymin;  x=matrix[i][0]+(y-matrix[i][1])/m;  if(x>=xmin && x<=xmax){  }  else{  x= xmin;  y = matrix[i][1]+m\*(x-matrix[i][0]);  }  }  else if(bmatrix[i][0]==6){  y=ymin;  x=matrix[i][0]+(y-matrix[i][1])/m;  if(x>=xmin && x<=xmax){  }  else{  x= xmax;  y = matrix[i][1]+m\*(x-matrix[i][0]);  }  }  else if(bmatrix[i][0]==1){  x= xmin;  y = matrix[i][1]+m\*(x-matrix[i][0]);  }  else if(bmatrix[i][0]==2){  x= xmax;  y = matrix[i][1]+m\*(x-matrix[i][0]);  }  else{  x = matrix[i][0];  y = matrix[i][1];  }  if(bmatrix[i][1]==8){  y1=ymax;  x1=matrix[i][2]+(y1-matrix[i][3])/m;  }  else if(bmatrix[i][1]==9){  y1=ymax;  x1=matrix[i][2]+(y1-matrix[i][3])/m;  if(x1>=xmin && x1<=xmax){  }  else{  x1= xmin;  y1 = matrix[i][3]+m\*(x1-matrix[i][2]);  }  }  else if(bmatrix[i][1]==10){  y1=ymax;  x1=matrix[i][2]+(y1-matrix[i][3])/m;  if(x1>=xmin && x1<=xmax){  }  else{  x1= xmax;  y1 = matrix[i][3]+m\*(x1-matrix[i][2]);  }  }  else if(bmatrix[i][1]==4){  y1=ymin;  x1=matrix[i][2]+(y1-matrix[i][3])/m;  }  else if(bmatrix[i][1]==5){  y1=ymin;  x1=matrix[i][2]+(y1-matrix[i][3])/m;  if(x1>=xmin && x1<=xmax){  }  else{  x1= xmin;  y1 = matrix[i][3]+m\*(x1-matrix[i][2]);  }  }  else if(bmatrix[i][1]==6){  y1=ymin;  x1=matrix[i][2]+(y1-matrix[i][3])/m;  if(x1>=xmin && x1<=xmax){  }  else{  x1= xmax;  y1 = matrix[i][3]+m\*(x1-matrix[i][2]);  }  }  else if(bmatrix[i][1]==1){  x1= xmin;  y1 = matrix[i][3]+m\*(x1-matrix[i][2]);  }  else if(bmatrix[i][1]==2){  x1= xmax;  y1 = matrix[i][3]+m\*(x1-matrix[i][2]);  }  else{  x1 = matrix[i][2];  y1 = matrix[i][3];  }  glBegin(GL\_LINE\_STRIP);  glVertex2f(x,y);  glVertex2f(x1,y1);  glEnd();  }  }  glFlush();  } int main(int argc,char\* argv[]) {  glutInit(&argc,argv);  glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB);  glutInitWindowSize(480,480);  glutCreateWindow("Cohen Sutherland Line clipping");  glutDisplayFunc(myDisplay);  myInit();  glutMainLoop();  return 1; } |
| --- |

Sample Input:

// Enter number of lines :4

// Enter point 1(x1,y1) :45 125

// Enter point 2(x2,y2) :225 110

// Enter point 1(x1,y1) :80 75

// Enter point 2(x2,y2) :160 85

// Enter point 1(x1,y1) :45 110

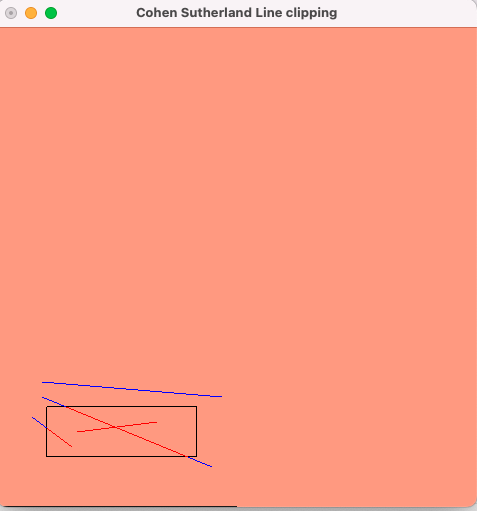
// Enter point 2(x2,y2) :215 40

// Enter point 1(x1,y1) :35 90

// Enter point 2(x2,y2) :75 60

// Enter window xmin, xmax,ymin,ymax :50 200 50 100

Sample Output:



**Learning Outcome:**

1. Learnt how to perform line clipping using cohen sutherland line clipping algorithm

2. Learnt how to construct the clipping window..

3. Was able to distinguish the clipped lines from actual lines.