**Practical Question 4**

**Cohen and Sutherland line clipping**

#include<iostream>

#include<graphics.h>

#include<conio.h>

using namespace std;

typedef struct coordinate

{

int x,y;

char code[4];

}point;

void line(point p1,point p2)

{

line(p1.x,p1.y,p2.x,p2.y);

}

point setcode(point p)

{

point ptemp;

if(p.y<100)

ptemp.code[0]='1';

else

ptemp.code[0]='0';

if(p.y>350)

ptemp.code[1]='1';

else

ptemp.code[1]='0';

if(p.x>450)

ptemp.code[2]='1';

else

ptemp.code[2]='0';

if(p.x<150)

ptemp.code[3]='1';

else

ptemp.code[3]='0';

ptemp.x=p.x;

ptemp.y=p.y;

return(ptemp);

}

int line\_area(point p1,point p2)

{

int i,flag=0;

for(i=0;i<4;i++)

{

if((p1.code[i]!='0') || (p2.code[i]!='0'))

flag=1;

}

if(flag==0)

return(0);

for(i=0;i<4;i++)

{

if((p1.code[i]==p2.code[i]) && (p1.code[i]=='1'))

flag='0';

}

if(flag==0)

return(1);

return(2);

}

point resetlinept(point p1,point p2)

{

point temp;

int x,y,i;

float m,k;

if(p1.code[3]=='1')

x=150;

if(p1.code[2]=='1')

x=450;

if((p1.code[3]=='1') || (p1.code[2]=='1'))

{

m=(float)(p2.y-p1.y)/(p2.x-p1.x);

k=(p1.y+(m\*(x-p1.x)));

temp.y=k;

temp.x=x;

for(i=0;i<4;i++)

temp.code[i]=p1.code[i];

if(temp.y<=350 && temp.y>=100)

return (temp);

}

if(p1.code[0]=='1')

y=100;

if(p1.code[1]=='1')

y=350;

if((p1.code[0]=='1') || (p1.code[1]=='1'))

{

m=(float)(p2.y-p1.y)/(p2.x-p1.x);

k=(float)p1.x+(float)(y-p1.y)/m;

temp.x=k;

temp.y=y;

for(i=0;i<4;i++)

temp.code[i]=p1.code[i];

return(temp);

}

else

return(p1);

}

int main()

{

initwindow(800,800);

point p1,p2,p3,p4,ptemp;

cout<<"Enter starting co-ordinate"<<endl;

cin>>p1.x>>p1.y;

cout<<"Enter ending co-ordinate"<<endl;

cin>>p2.x>>p2.y;

delay(1000);

rectangle(150,350,450,100);

delay(1000);

line(p1,p2);

delay(1000);

cleardevice();

delay(1000);

p1=setcode(p1);

p2=setcode(p2);

int a=line\_area(p1,p2);

delay(1000);

switch(a)

{

case 0:

rectangle(150,350,450,100);;

delay(1000);

line(p1,p2);

break;

case 1:

rectangle(150,350,450,100);

delay(1000);

break;

case 2:

p3=resetlinept(p1,p2);

p4=resetlinept(p2,p1);

rectangle(150,350,450,100);

delay(1000);

line(p3,p4);

break;

}

delay(7000);

closegraph();

getch();

return 0;

}

**OUTPUT:**





