//code

Title : Write C++ program to implement Cohen Southerland line clipping algorithm.

Sample Code:

```
#include<iostream>
#include<stdlib.h>
#include<math.h>
#include<graphics.h>
#include<dos.h> using
namespace std;
class Coordinate
{
public: int
x,y; char
code[4];
};
class Lineclip
{
public:
Coordinate PT; void
drawwindow();
void drawline(Coordinate p1,Coordinate p2);
Coordinate setcode(Coordinate p); int
visibility(Coordinate p1,Coordinate p2);
Coordinate resetendpt(Coordinate p1,Coordinate p2);
};
int main()
{
Lineclip lc;
int gd = DETECT,v,gm;
Coordinate p1,p2,p3,p4,ptemp;
cout<<"\n Enter x1 and y1\n";
cin>>p1.x>>p1.y; cout<<"\n
Enter x2 and y2\n";
cin>>p2.x>>p2.y;
initgraph(&gd,&gm,"");
lc.drawwindow(); delay(2000);
lc.drawline (p1,p2);
```

```
delay(2000); cleardevice();
delay(2000);
p1=lc.setcode(p1);
p2=lc.setcode(p2);
v=lc.visibility(p1,p2);
delay(2000);
switch(v)
{
case 0: lc.drawwindow();
delay(2000);
lc.drawline(p1,p2);
break; case
1:lc.drawwindow();
delay(2000); break;
case 2:p3=lc.resetendpt(p1,p2);
p4=lc.resetendpt(p2,p1);
lc.drawwindow(); delay(2000);
lc.drawline(p3,p4);
break;
}
delay(2000);
closegraph();
}
void Lineclip::drawwindow()
{
line(150,100,450,100); line(450,100,450,350);
line(450,350,150,350); line(150,350,150,100);
}
void Lineclip::drawline(Coordinate p1,Coordinate p2)
{
line(p1.x,p1.y,p2.x,p2.y);
}
Coordinate Lineclip::setcode(Coordinate p)
{
Coordinate 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 Lineclip:: visibility(Coordinate p1,Coordinate p2)
{ int
i,flag=0;
for(i=0;i<4;i++)
{
if(p1.code[i]!='0' || (p2.code[i]=='1')) flag='0';
}
if(flag==0) return(0);
for(i=0;i<4;i++)
{
if(p1.code[i]==p2.code[i] && (p2.code[i]=='1')) flag='0';
}
if(flag==0)
return(1); return(2);
}
Coordinate Lineclip::resetendpt(Coordinate p1,Coordinate p2)
{
Coordinate 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[1]=='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);
}
```

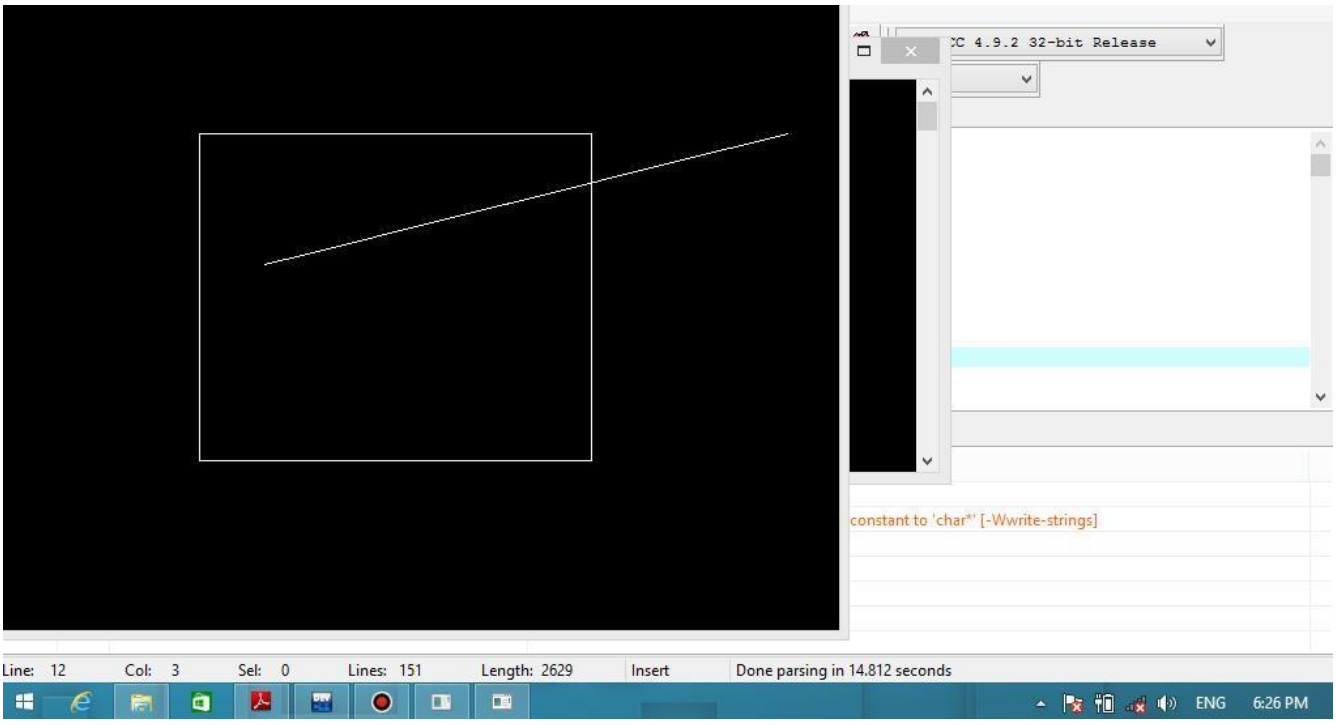
Input :

x1 , y1:
200
200

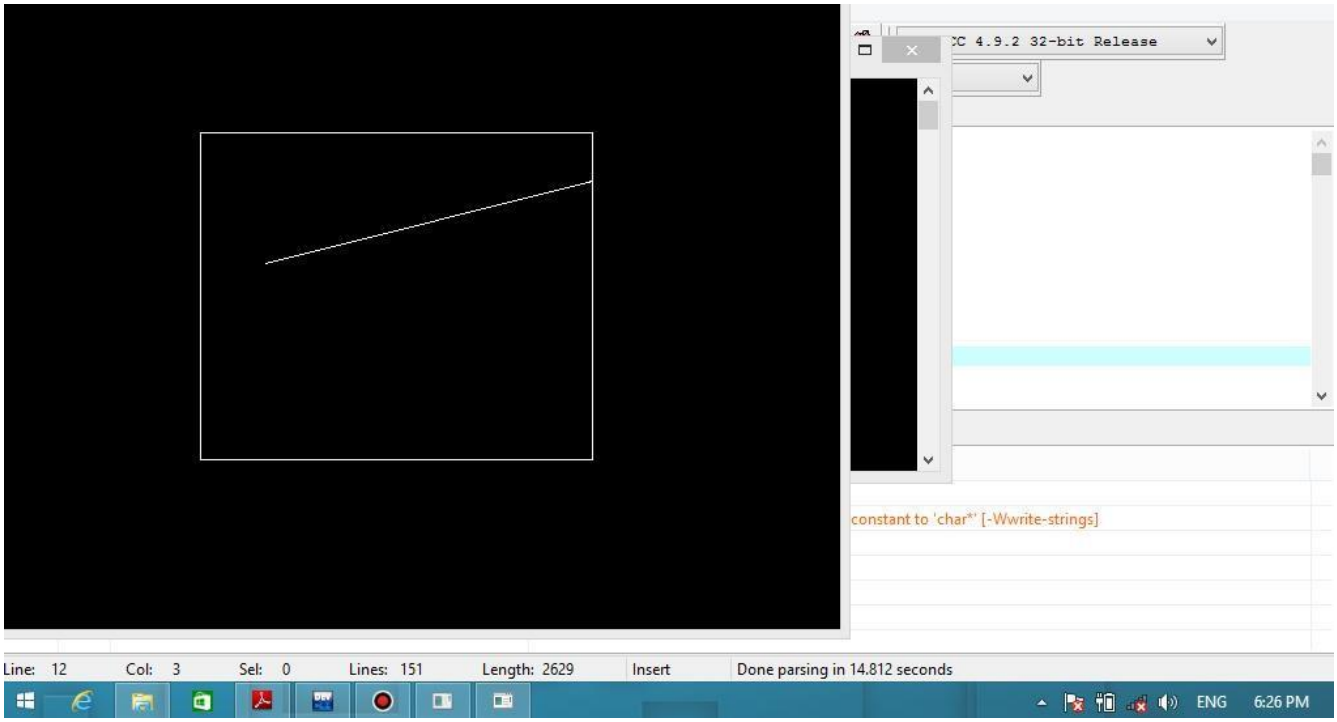
x2, y2: 600
100

Output :

Without clipping :



With clipping :


XC 4.9.2 32-bit Release
constant to 'char*' [-Wwrite-strings]
Line: 12 Col: 3 Sel: 0 Lines: 151 Length: 2629 Insert Done parsing in 14.812 seconds
ENG
6:26 PM