Ans 2 →

Step 1: Calculate positions of both endpoints of the line.

Step 2: Perform OR operation on both of these endpoints

Step 3: If the OR operation gives 0000
Then line is considered to be visible
else
perform AND operation on both the endpoints
if And ≠0000
then the line is invisible
else
And =0000
Line is considered the clipped case.

Step 4: If line is clipped case, find an intersection with boundaries of the window

$$m = (y_2 - y_1)/(x_2 - x_1)$$

(a)- if bit 1 is "1" line intersects with left boundary of rectangle window

$$y_3 = y_1 + m(x - x_1)$$

where $x = x_{wmin}$

where xwmin is the minimum value of
x co-ordinate of window

(b)- If bit-2 is "1" line intersect with right boundry

$y_3 = y_1 + m(x - x_1)$

where $x = x_{wmax}$

where x would is maximum value of x co-ordinate of the window

(c)- If bit 3 is "1" line intersects with bottom boundry

$x_3 = x_1 + (y - y_1)/m$

where $y = y_{wmin}$

y_{wmin} is the minimum value of y co-ordinate of the window

(d)- If bit 4 is "1" line intersects with the top boundary

$x_3 = x_1 + (y - y_1)/m$

where $y = y_{wmax}$

y_{wmax} is the maximum value of y coordinate of the window.

Program ⇒

```
# include <iostream.h>
# include < conio.h>
# include< graphics.h>
# include < dos.h>

class data
{
 int gd, gmode, x, y, xmin, ymin, ymax, xmax;
  int a1, a2
  float x1, y1, x2, y2, x3, y3;
  int xs, ys, xe, ye;
 float maxx, maxy;
 Public;
 void getdata();
 void find();
  void clip();
 void display (Float, float, float, float);
 void clockond (int);
 void Showbit (int);
 };
  void data :: getdata()
```

```
{
cout<< "Enter the minimum and maximum
        coordinate of window(x,y)";
cin >>xmin >>ymin >>xmax >>ymax;
cout<< "Enter the end points of the link to
        be clipped";
cin >>xs >>ys >>xe >>ye;
display (xs, ys, xe, ye);
}
void data :: display (float xs, float, ys, float, xe
                      float ye)
{
int gd = DETECT
initgraph (& gd, & gmode, "");
maxx = getmaxx ();
maxy = getmaxy ();
line (maxx/2, 0, maxx/2, maxy);
line (0, maxy/2, maxx, maxy/2);
rectangle (maxx/2 + xmin, maxy/2 - ymax, maxx/2
+ xmax, maxy/2 - ymin);
```

```
line (maxx/2+xs, maxy/2-ys, maxx/2+xe, maxy/2-ye
getch();
}
void data :: find()
{
a1=0;
a2=0;
if((ys-max)>0)
   a1+8=8;
if((ymin-ys)>0)
   a1+=4;
if ((xs-xmax)>0)
   a1+=1
if ((ye-ymax)>0)
   a2+=8;
   if ((ymin-ye)>0)
   a2+=4;
   if ((xe-xmax)>0)
   a2+=2;
   if ((xmin-xe)>0)
   a2+=1;
```

```
cout<<"The area code of 1st point is";
 showbit(a1);
 getch();
cout<<"The area code of 2nd point is";
 showbit(a2);
  getch();
}
void data::showbit(int n)
{
 int i,K,and;
  for(i=3;i>=0;i--)
 {
 and=0x1<<i;
  K=n&and;
K==0?cout<<"0":cout<<"1";
  }
  }
void data::clip()
{
int j=a1&a2;
  if(j==0)
 {
cout<<"Line is perfect candidate for
                clipping";
```

```
if (a1==0)
 {
 else
 {
  checkonof(a1);
   x2=x1 : y2=y1;
 }
if (a2=0)
 {
   x3=xe; y3=ye;
 }
else
 {
   checkonof(a2);
   x3=x1; y3=y1;
 }
x8=x2; y8=y2; xe=x3; ye=y3;
cout<<end;
   display (x8, y8, xe, ye);
cout<<"Line after clipping";
   getch();
}
void data::checkonof(int i)
{
int j, k, l, m
```

```
i = i&1;
x1 = 0; y1 = 0;
if (i==1)
{
  x1 = xmin;
y1 = ys + ((x1-xs)/(xe-xs))*(ye-ys);
}
j = i&2;
if (j<<1)
{
  y1 = ymax;
  x1 = xs + ((y1-ys)/(ye-ys))*(xe-xs);
}
k = i&4;
if (k==4)
{
  y1 = ymin;
  x1 = xs + ((y1-ys)/(ye-ys))*(xe-xs);
}
m = i&8;
  if (m==1)
  {
  x1 = xmax;
```

```
y1=y+((x1-x3)/(xe-x5))*(ye-y8);
}
main()
{
data S;
clrscr();
S.getdata();
S.find();
closegraph();
return();
}
```