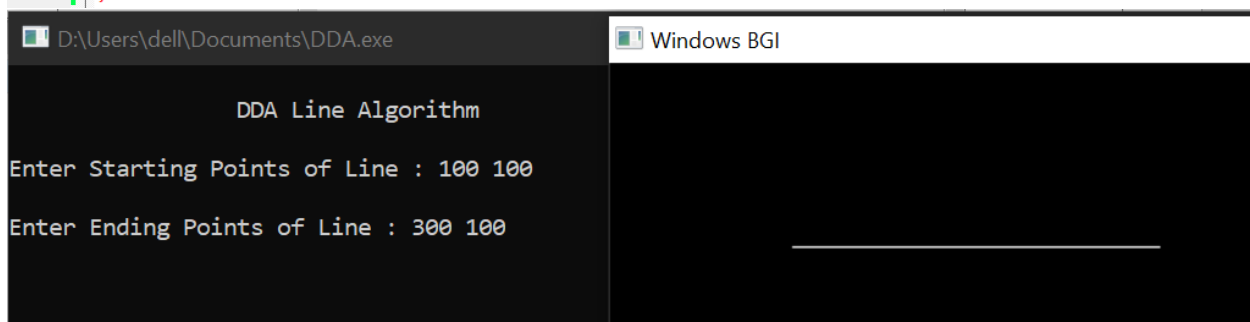


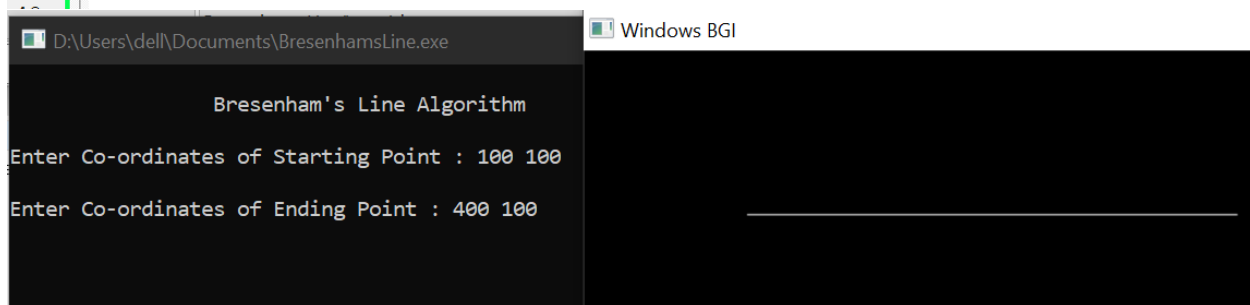
```
Start here x DDA.cpp x
1  #include<bits/stdc++.h>
2  #include<graphics.h>
3  #include<math.h>
4  #include<conio.h>
5  using namespace std;
6
7  void LineDDA()
8  {
9      int x1,x2,y1,y2;
10     cout<<"\n \t\t DDA Line Algorithm\n";
11     cout<<"\nEnter Starting Points of Line : ";
12     cin>>x1>>y1;
13     cout<<"\nEnter Ending Points of Line : ";
14     cin>>x2>>y2;
15
16     int Dx = x2-x1, Dy = y2-y1, steps,k;
17     float xin, yin, X = x1, Y = y1;
18     if(abs(Dx) > abs(Dy)){
19         steps = abs(Dx);
20     }
21     else{
22         steps = abs(Dy);
23     }
24
25     xin = Dx/(float)steps;
26     yin = Dy/(float)steps;
27
28     int gd = DETECT, gm;
29     initgraph(&gd, &gm, (char *) "");
30
31     putpixel(round(X), round(Y), WHITE);
32
33     for(k=0; k<steps; k++){
34         X = X + xin;
35         Y = Y + yin;
36         putpixel(round(X), round(Y), WHITE);
37     }
38
39     getch();
40     closegraph();
41 }
42
43
44 int main()
45 {
46     LineDDA();
47     return 0;
48 }
```



```

Start here x Bresenham'sLine.cpp x
1  #include<bits/stdc++.h>
2  #include<graphics.h>
3  #include<conio.h>
4  using namespace std;
5
6  void Bresenham'sLine()
7  {
8      int x0, y0, x1, y1;
9      cout<<"\n \t\t Bresenham's Line Algorithm\n";
10     cout<<"\nEnter Co-ordinates of Starting Point : ";
11     cin>>x0>>y0;
12     cout<<"\nEnter Co-ordinates of Ending Point : ";
13     cin>>x1>>y1;
14
15     int dx, dy, p, x, y;
16
17     dx = x1 - x0;
18     dy = y1 - y0;
19
20     x = x0;
21     y = y0;
22
23     p = 2 * (dy-dx);
24
25     int gd = DETECT, gm;
26     initgraph(&gd, &gm, (char *) "");
27
28     while(x < x1){
29         if(p >= 0){
30             putpixel(x, y, 7);
31             y = y+1;
32             p = p+(2*dy)-(2*dx);
33         }
34         else{
35             putpixel(x, y, 7);
36             p = p+(2*dy);
37         }
38         x = x+1;
39     }
40     getch();
41     closegraph();
42 }
43 int main()
44 {
45     Bresenham'sLine();
46     return 0;
47 }

```



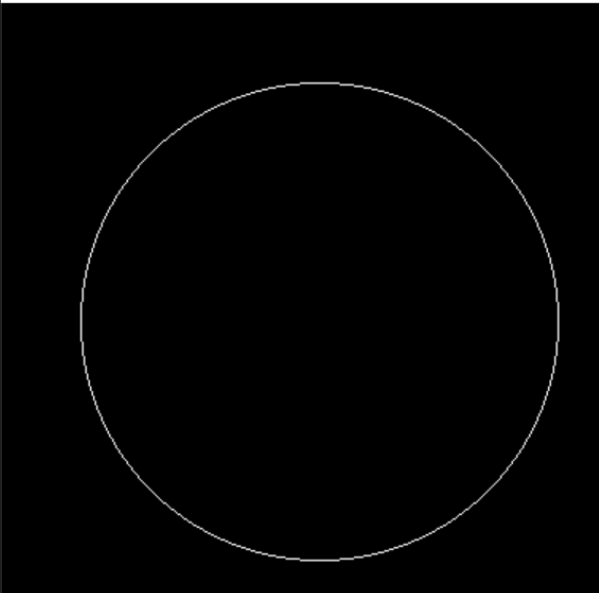
```
Start here x Bresenham'sCircle.cpp x
1  #include<bits/stdc++.h>
2  #include<graphics.h>
3  #include<conio.h>
4  using namespace std;
5
6  void Bresenham'sCircle()
7  {
8      int h, k, r;
9      void plotPixel(int, int, int, int);
10     cout<<"\n \t\t Bresenham's Circle Algorithm\n";
11     cout<<"\nEnter Center Co-ordinates of Circle : ";
12     cin>>h>>k;
13     cout<<"\nEnter Radius of Circle : ";
14     cin>>r;
15
16     int x=0,y=r,d = 3-2*r ;
17
18     int gd = DETECT,gm;
19     initgraph(&gd,&gm,(char *) "");
20
21     while(x <= y)
22     {
23         plotPixel(x,y,h,k);
24         if(d < 0)
25         {
26             d = d+(4*x)+6;
27         }
28         else
29         {
30             d = d+(4*x)-(4*y)+10;
31             y--;
32         }
33         x++;
34     }
35     getch();
36     closegraph();
37 }
38 void plotPixel(int x, int y, int h, int k)
39 {
40
41     putpixel(x+h, y+k, WHITE);
42     putpixel(x+h, -y+k, WHITE);
43     putpixel(-x+h, -y+k, WHITE);
44     putpixel(-x+h, y+k, WHITE);
45     putpixel(y+h, x+k, WHITE);
46     putpixel(y+h, -x+k, WHITE);
47     putpixel(-y+h, -x+k, WHITE);
48     putpixel(-y+h, x+k, WHITE);
49
50 }
51 int main()
52 {
53     Bresenham'sCircle();
54     return 0;
55 }
```

D:\Users\deli\Documents\Bresenham'sCircle.exe

Bresenham's Circle Algorithm

Enter Center Co-ordinates of Circle : 200 200

Enter Radius of Circle : 150



```

Start here x MidpointCircle.cpp x
1  #include<bits/stdc++.h>
2  #include<graphics.h>
3  #include<conio.h>
4  #include<math.h>
5  using namespace std;
6
7  int main()
8  {
9      int gd = DETECT, gm = DETECT;
10     int x=0, y, r, d, h, k;
11     initgraph(&gd,&gm,(char *)"");
12     cout<<"\n \t\t MidPoint Circle Algorithm\n";
13     cout<<"\nEnter Center Co-ordinates of Circle : ";
14     cin>>h>>k;
15     cout<<"\nEnter Radius of Circle : ";
16     cin>>r;
17
18     y = r;
19     putpixel(h,k,WHITE);
20     d = 1-r;
21
22     while(x <= y)
23     {
24         putpixel(x+h, y+k, WHITE);
25         putpixel(x+h, -y+k, WHITE);
26         putpixel(-x+h, -y+k, WHITE);
27         putpixel(-x+h, y+k, WHITE);
28         putpixel(y+h, x+k, WHITE);
29         putpixel(y+h, -x+k, WHITE);
30         putpixel(-y+h, -x+k, WHITE);
31         putpixel(-y+h, x+k, WHITE);
32
33         if(d < 0){
34             d = d+(2*x)+3;
35         }
36         else{
37             d = d+2*(x-y)+5;
38             y--;
39         }
40         x++;
41     }
42     getch();
43     closegraph();
44 }

```

