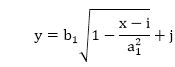
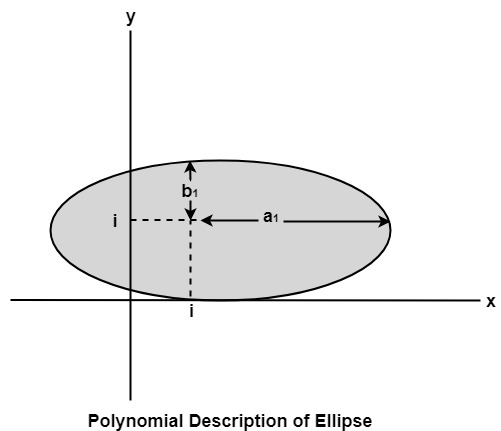
Polynomial Method:

The ellipse has a major and minor axis. If a1 and b1are major and minor axis respectively. The centre of ellipse is (i, j). The value of x will be incremented from i to a1and value of y will be calculated using the following formula



Drawback of Polynomial Method:

1. It requires squaring of values. So floating point calculation is required.
2. Routines developed for such calculations are very complex and slow.



Algorithm:

1. Set the initial variables: a = length of major axis; b = length of minor axis; (h, k) = coordinates of ellipse center; x = 0; i = step; xend = a.

2. Test to determine whether the entire ellipse has been scan-converted. If x>xend, stop.

3. Compute the value of the y coordinate:

Polynomial Method

4. Plot the four points, found by symmetry, at the current (x, y) coordinates:

          Plot (x + h, y + k)           Plot (-x + h, -y + k)           Plot (-y - h, x + k)           Plot (y + h, -x + k)

5. Increment x; x = x + i.

6. Go to step 2.

Program to draw an Ellipse using Polynomial Method:

1. #include <graphics.h>
2. #include <stdlib.h>
3. #include <math.h>
4. #include <stdio.h>
5. #include <conio.h>
6. #include <iostream.h>
8. **class** bresen
9. {
10. **float** x, y, a, b, r, t, te, xend, h, k, step;
11. **public**:
12. **void** get ();
13. **void** cal ();
14. };
15. **void** main ()
16. {
17. bresen b;
18. b.get ();
19. b.cal ();
20. getch ();
21. }
22. **void** bresen :: get ()
23. {
24. cout<<"\n ENTER CENTER OF ELLIPSE";
25. cout<<"\n enter (h, k) ";
26. cin>>h>>k;
27. cout<<"\n ENTER LENGTH OF MAJOR AND MINOR AXIS";
28. cin>>a>>b;
29. cout<<"\n ENTER Step Size";
30. cin>> step;
31. }
32. **void** bresen ::cal ()
33. {
34. /\* request auto detection \*/
35. **int** gdriver = DETECT,gmode, errorcode;
36. **int** midx, midy, i;
37. /\* initialize graphics and local variables \*/
38. initgraph (&gdriver, &gmode, " ");
39. /\* read result of initialization \*/
40. errorcode = graphresult ();
41. **if** (errorcode ! = grOK)    /\*an error occurred \*/
42. {
43. printf("Graphics error: %s \n", grapherrormsg (errorcode);
44. printf ("Press any key to halt:");
45. getch ();
46. exit (1); /\* terminate with an error code \*/
47. }
48. x = 0;
49. xend=a;
50. whilex (x<xend)
51. {
52. t= (1-((x \* x)/ (a \* a)));
53. **if** (t<0)
54. te=-t;
55. **else**
56. te=t;
57. y=b \* sqrt (te);
58. putpixel (h+x, k+y, RED);
59. putpixel (h-x, k+y, RED);
60. putpixel (h+x, y-y, RED);
61. putpixel (h-x, k-y, RED);
62. x+=step;
63. }
64. getch();
65. }

**Output:**

