Name - Siddharth Thapa

Univ. Roll no. - 1121144

Subject - Computer Graphics

Subject Code - PBC-602

P1.

Bresenham Line Drawing Algorithm.

Step 1: Start.

Step 2: Declare variables $x_1, x_2, y_1, y_2, d, i_1, i_2, dx, dy$

Step 3: Enter values of x_1, y_1, x_2, y_2

Step 4: Calculate $dx = x_2 - x_1$

$dy = y_2 - y_1$

$i_1 = 2 * dy$

$i_2 = 2 * (dy - dx)$

$d = i_1 - dx$

Step 5: Consider (x, y) as starting point x_{end} as max. possible value of x.

Then $x = x_2$

$y = y_2$

$x_{end} = x_1$

If $dx > 0$

Then $x = x_1$

$y = y_1$

$x_{end} = x_2$

Step 6 : Generate point at (x, y) coordinates

Step 7: Check if whole line is generated

If $x >= x_{end}$

Stop

Step 8: Calculate co-ordinates of next pixel

If $d < 0$

Then $d = d + i_1$

If $d \geq 0$

Then $d = d + i_2$

Increment $y = y + 1$

Step 9: Increment $x = x + 1$

Step 10: Draw point of latest coordinates

Step 11: Go to step 7.

Step 12: End.

Program to implement Bresenham's Line Drawing.

```
#include<stdio.h>
#include<graphics.h>
void drawline (int x0, int y0, int x1, int y1)
{
int dx, dy, p, x, y;
dx = x1 - x0;
dy = y1 - y0;
x = x0;
y = y0;
p = 2 * dy - dx;
while (x<x1)
{
if (p>=0)
{
putpixel(x, y, 7);
y = y + 1;
p = p + 2 * dy - 2 * dx;
}
```

```
else
{
putpixel (x, y, 7);
p = p + 2 * dy;
}
x = x + 1;
}
}
int main()
{
int gdriver = DETECT, gmode, error, x0, y0,
x1, y1;
initgraph (&gdriver, &gmode, "C:\\turboc3\\
bgi");
printf ("Enter coordinates: ");
scanf ("%d %d", &x0, &y0);
scanf ("%d %d", &x1, &y1);
drawline (x0, y0, x1, y1);
return 0;
}
```

P2. Midpoint Circle Drawing Algorithm

Step 1: Put $x = 0$, $y = 1$ in equation 2.

we have $p = 1 - r$

Step 2: Repeat steps while $x \leq y$

Plot (x, y)

If $(p < 0)$

Then set $p = p + 2x + 3$.

Else

$p = p + 2x + 3$

$y = y - 1$

$x = x + 1$

Step 3: End.

Program to implement mid point circle drawing algorithm.

```
#include<graphics.h>
#include<stdio.h>
void midpoint (int midx, int midy, int r)
{
int x=0, y=r, gd=0, gm, di, dnext;
  initgraph(&gd, &gm, "");
di = 1.25-r;
while(x<=y)
{
  if (di>=0)
  {
    dnext = di+2*(x-y)+1;
    x++;
    y--;
  }
else
  {dnext = di+2*x+1;
   x++;
}
```

```
putpixel (x+midx, y+midy, 5);
putpixel (y+midx, x+midy, 5);
putpixel (-y+midx, -y+midy, 5);
putpixel (-y+midx, -x+midy, 5);
putpixel (-y+midx, x+midy, 5);
putpixel (y+midx, -x+midy, 5);
putpixel (x+midx, -y+midy, 5);
putpixel (-x+midx, y+midy, 5);
    di = dnext;
  }
  getch();
  closegraph();
}
int main()
{
  int gd = 0, gm;
  int midx = 0, midy = 0, r = 0.
  printf("Enter radius of circle:");
  scanf("%d", &r);
  printf("Enter co-ordinates of center
          (x and y):");
  scanf("%d %d", &midx, &midy);
```

```
midpoint(midx, midy, r);
return 0;
```