**List of Practical’s**

1. Draw Various geomatric shapes using in built functions.

* Draw Line
* Draw Rectangle
* Draw Ellipse
* Draw Polygon

1. Draw complex object using basic shapes.
2. Change Font size, Font face etc using Font Dialog box
3. Change Forground And Background color using color Dialog box
4. Write a program to draw aline using Digital Differential Analyzer (DDA) algorithm
5. Write a program to draw aline using Bresenham’s algorithm
6. Write a program to draw circle using midpoint algorithm.
7. Write a program to draw animated circles
8. Write a program to implement Flood fill
9. Write a program to implement Boundary fill
10. **Draw Various geomatric shapes using in built functions.**

* ***Draw Line***

**Code:**

private void button1\_Click(object sender, EventArgs e)

{

// Creating object of Graphics Class

Graphics g = this.CreateGraphics ();

// Creating object of Pen Class Class

Pen p=new Pen (Brushes .Color,Width of brush);

g.DrawLine(p,Start point, End point);

}

***Example***

private void button1\_Click(object sender, EventArgs e)

{

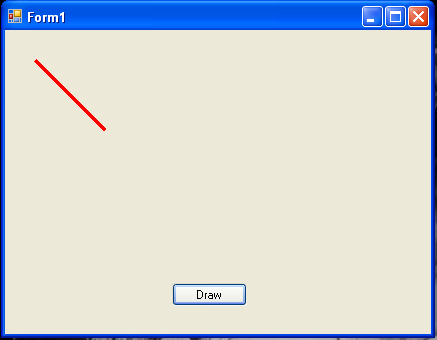
Graphics g = this.CreateGraphics ();

Pen p=new Pen (Brushes .Red,3);

g.DrawLine(p, 30,30, 100,100);

}

**Output:**



* **Draw Rectangle**

**Code:**

private void button1\_Click(object sender, EventArgs e)

{

// Creating object of Graphics Class

Graphics g = this.CreateGraphics ();

// Creating object of Pen Class Class

Pen p=new Pen (Brushes .Color,Width of brush);

Rectangle r = new Rectangle(Point X,Point,Y,Width, Height);

g.DrawRectangle(p, r);

}

**Example:**

private void button1\_Click(object sender, EventArgs e)

{

Graphics g = this.CreateGraphics ();

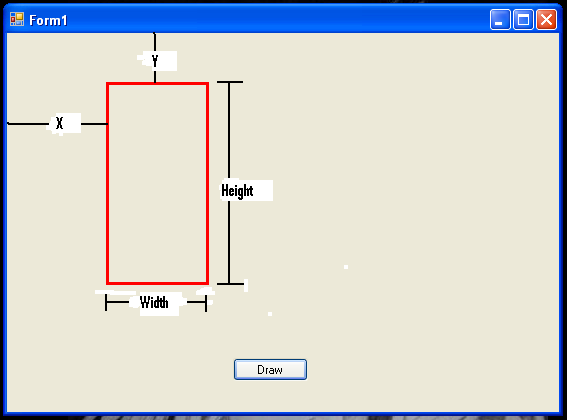
Pen p=new Pen (Brushes .Red,3);

Rectangle r = new Rectangle(20, 20, 100, 200);

g.DrawRectangle(p, r);

}

**Output:**



* **Draw Ellipse**

**Code:**

private void button1\_Click(object sender, EventArgs e)

{

// Creating object of Graphics Class

Graphics g = this.CreateGraphics ();

// Creating object of Pen Class Class

Pen p=new Pen (Brushes .Color,Width of brush);

g.DrawEllipse(pen,Point X, Point Y, Width, Height);

}

**Example:**

private void button1\_Click(object sender, EventArgs e)

{

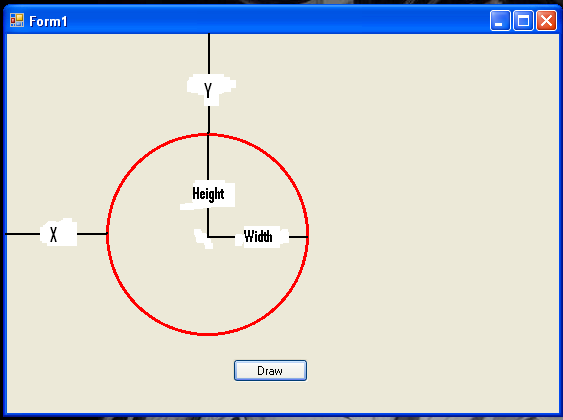
Graphics g = this.CreateGraphics ();

Pen p=new Pen (Brushes .Red,3);

g.DrawEllipse(p,100,100,200,200 );

}

**Output:**



* **Draw Polygon**

**Code:**

private void button1\_Click(object sender, EventArgs e)

{

// Creating object of Graphics Class

Graphics g = this.CreateGraphics ();

// Creating object of Pen Class Class

Pen p=new Pen (Brushes .Color,Width of brush);

//Declaring Array

Point[] polygonPoints = new Point[5];

polygonPoints[0] = new Point(x,y);

polygonPoints[1] = new Point(x,y);

polygonPoints[2] = new Point(x,y);

polygonPoints[3] = new Point(x,y);

polygonPoints[4] = new Point(x,y);

g.DrawPolygon(p, polygonPoints);

}

**Example:**

private void button1\_Click(object sender, EventArgs e)

{

Point[] polygonPoints = new Point[5];

polygonPoints[0] = new Point(113, 283);

polygonPoints[1] = new Point(70, 156);

polygonPoints[2] = new Point(180, 70);

polygonPoints[3] = new Point(290, 156);

polygonPoints[4] = new Point(250, 283);

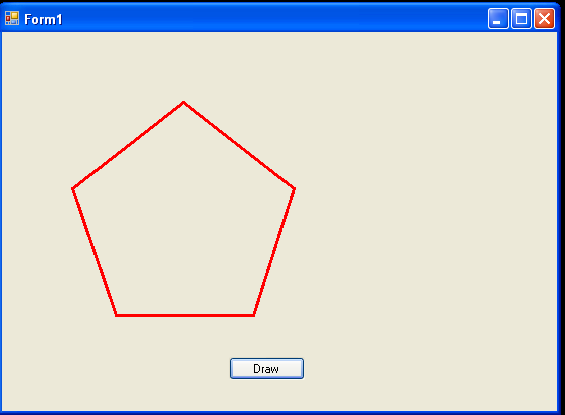
Graphics g = this.CreateGraphics ();

Pen p=new Pen (Brushes .Red,3);

g.DrawPolygon(p, polygonPoints);

}

**Output:**



1. **Draw complex object using basic shapes.**

**Code:**

private void button1\_Click(object sender, EventArgs e)

{

Graphics g = this.CreateGraphics();

Pen p = new Pen(Brushes.Yellow, 3);

g.DrawEllipse(p, 100, 100, 200, 200);

g.DrawEllipse(p, 160, 150, 20, 40);

g.FillEllipse (Brushes .Black , 160, 170, 20, 20);

g.DrawEllipse(p, 220, 150, 20, 40);

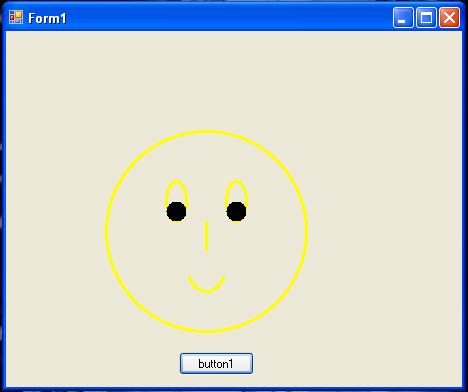
g.FillEllipse(Brushes .Black , 220, 170, 20, 20);

g.DrawLine(p, 200, 190, 200, 220);

g.DrawArc(p, 180, 200, 40, 60, 40, 100);

}

**Output:**



1. **Change Font size, Font face etc using Font Dialog box**

**Code:**

*--Steps to include font Dialog Box on Windows Form*

*File🡪New🡪Project🡪Toolbox🡪Dialog box🡪Font Dialog Box(double click)*

private void button1\_Click(object sender, EventArgs e)

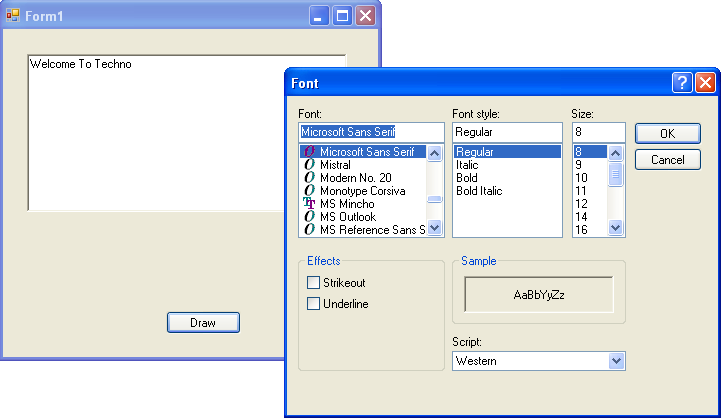
{

if (fontDialog1.ShowDialog() == DialogResult.OK)

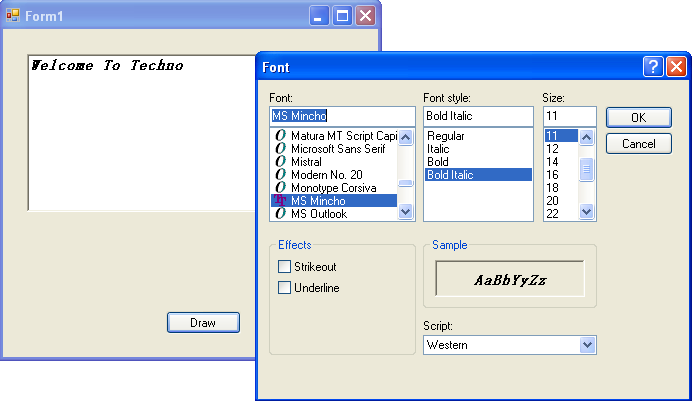
richTextBox1.Font = fontDialog1.Font;

}

**Before Implimentation:**



**After Implimentation:**



1. **Change Forground And Background color using color Dialog box**

**Code:**

***Fore color***

private void button1\_Click(object sender, EventArgs e)

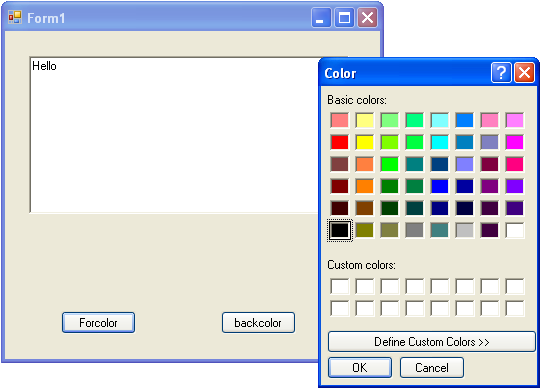
{

if (colorDialog1.ShowDialog() == DialogResult.OK)

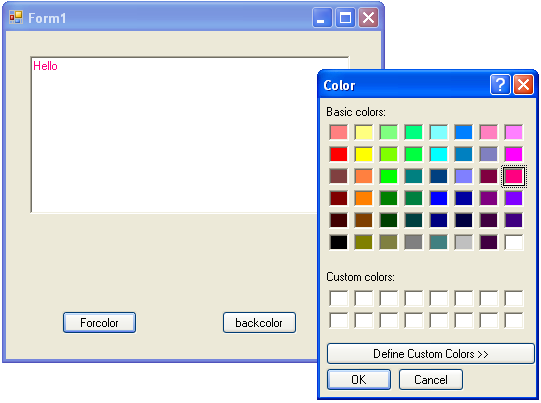
richTextBox1.ForeColor = colorDialog1 .Color ;

}

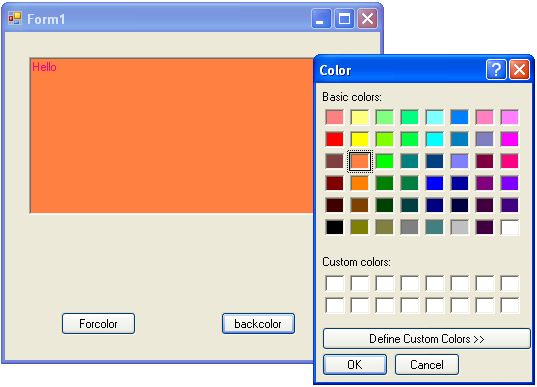
**Before Background and foreground color:**



**After foreground color:**



**After Background and foreground color:**



1. **WAP to draw aline using Digital Differential Analyzer (DDA) algorithm**

**Code:**

# include <iostream.h>

# include <graphics.h>

# include <conio.h>

# include <math.h>

void dda\_line(const int,const int,const int,const int);

int main( )

{

int driver=VGA;

int mode=VGAHI;

int x\_1=0;

int y\_1=0;

int x\_2=0;

int y\_2=0;

do

{

gotoxy(8,10);

cout<<"Coordinates of Point-I (x1,y1) :";

gotoxy(8,11);

cout<<"=================================";

gotoxy(12,13);

cout<<"Enter the value of x1 = ";

cin>>x\_1;

gotoxy(12,14);

cout<<"Enter the value of y1 = ";

cin>>y\_1;

gotoxy(8,18);

cout<<"Coordinates of Point-II (x2,y2) :";

gotoxy(8,19);

cout<<"==================================";

gotoxy(12,21);

cout<<"Enter the value of x2 = ";

cin>>x\_2;

gotoxy(12,22);

cout<<"Enter the value of y2 = ";

cin>>y\_2;

initgraph(&driver,&mode,"c:\\tc\\Bgi");

setcolor(15);

dda\_line(x\_1,y\_1,x\_2,y\_2);

setcolor(15);

outtextxy(110,460,"Press <Enter> to continue or any other key to exit.");

int key=int(getch( ));

if(key!=13)

break;

}while(1);

return 0;

}

void dda\_line(const int x\_1,const int y\_1,const int x\_2,const int y\_2)

{

int color=getcolor( );

int x1=x\_1;

int y1=y\_1;

int x2=x\_2;

int y2=y\_2;

if(x\_1>x\_2)

{

x1=x\_2;

y1=y\_2;

x2=x\_1;

y2=y\_1;

}

float dx=(x2-x1);

float dy=(y2-y1);

int steps=abs(dy);

if(abs(dx)>abs(dy))

steps=abs(dx);

float x\_inc=(dx/(float)steps);

float y\_inc=(dy/(float)steps);

float x=x1;

float y=y1;

putpixel(x,y,color);

for(int count=1;count<=steps;count++)

{

x+=x\_inc;

y+=y\_inc;

putpixel((int)(x+0.5),(int)(y+0.5),color);

}

}

**Output:**

Coordinates of Point-I (x1,y1)

=================================

Enter the value of x1 =40

Enter the value of y1=40

Coordinates of Point-II (x2,y2)

=================================

Enter the value of x2 =90

Enter the value of y2=90

1. **WAP to draw aline using Bresenham’s algorithm**

**Code:**

# include <iostream.h>

# include <graphics.h>

# include <conio.h>

# include <math.h>

void bresenham\_line(const int,const int,const int,const int);

int main( )

{

int driver=VGA;

int mode=VGAHI;

int x\_1=0;

int y\_1=0;

int x\_2=0;

int y\_2=0;

do

{

gotoxy(8,10);

cout<<"Coordinates of Point-I (x1,y1) :";

gotoxy(8,11);

cout<<"================================";

gotoxy(12,13);

cout<<"Enter the value of x1 = ";

cin>>x\_1;

gotoxy(12,14);

cout<<"Enter the value of y1 = ";

cin>>y\_1;

gotoxy(8,18);

cout<<"Coordinates of Point-II (x2,y2) :";

gotoxy(8,19);

cout<<"==================================";

gotoxy(12,21);

cout<<"Enter the value of x2 = ";

cin>>x\_2;

gotoxy(12,22);

cout<<"Enter the value of y2 = ";

cin>>y\_2;

initgraph(&driver,&mode,"c:\\tc\\Bgi");

setcolor(15);

bresenham\_line(x\_1,y\_1,x\_2,y\_2);

setcolor(15);

outtextxy(110,460,"Press <Enter> to continue or any other key to exit.");

int key=int(getch( ));

if(key!=13)

break;

} while(1);

return 0;

}

void bresenham\_line(const int x\_1,const int y\_1,const int x\_2,const int y\_2)

{

int color=getcolor( );

int x1=x\_1;

int y1=y\_1;

int x2=x\_2;

int y2=y\_2;

if(x\_1>x\_2)

{

x1=x\_2;

y1=y\_2;

x2=x\_1;

y2=y\_1;

}

int dx=abs(x2-x1);

int dy=abs(y2-y1);

int inc\_dec=((y2>=y1)?1:-1);

if(dx>dy)

{

int two\_dy=(2\*dy);

int two\_dy\_dx=(2\*(dy-dx));

int p=((2\*dy)-dx);

int x=x1;

int y=y1;

putpixel(x,y,color);

while(x<x2)

{

x++;

if(p<0)

p+=two\_dy;

else

{

y+=inc\_dec;

p+=two\_dy\_dx;

}

putpixel(x,y,color);

}

}

else

{

int two\_dx=(2\*dx);

int two\_dx\_dy=(2\*(dx-dy));

int p=((2\*dx)-dy);

int x=x1;

int y=y1;

putpixel(x,y,color);

while(y!=y2)

{

y+=inc\_dec;

if(p<0)

p+=two\_dx;

else

{

x++;

p+=two\_dx\_dy;

}

putpixel(x,y,color);

}

}

}

**Output:**

Coordinates of Point-I (x1,y1)

=================================

Enter the value of x1 =40

Enter the value of y1=40

Coordinates of Point-II (x2,y2)

=================================

Enter the value of x2 =90

Enter the value of y2=90

1. **Wap to draw circle using midpoint algorithm.**

**Code:**

# include <iostream.h>

# include <graphics.h>

# include <conio.h>

# include <math.h>

void midpoint\_circle(const int,const int,const int);

int main( )

{

int driver=VGA;

int mode=VGAHI;

int h=0;

int k=0;

int r=0;

do

{

gotoxy(8,10);

cout<<"Central Point of the Circle : (x,y) :";

gotoxy(8,11);

cout<<"======================================";

gotoxy(12,13);

cout<<"Enter the value of x\_center = ";

cin>>h;

gotoxy(12,14);

cout<<"Enter the value of y\_center = ";

cin>>k;

gotoxy(8,18);

cout<<"Radius of the Circle : r :";

gotoxy(8,19);

cout<<"========================================";

gotoxy(12,21);

cout<<"Enter the value of r = ";

cin>>r;

initgraph(&driver,&mode,"c:\\tc\\Bgi");

setcolor(15);

midpoint\_circle(h,k,r);

setcolor(15);

outtextxy(110,460,"Press <Enter> to continue or any other key to exit.");

int key=int(getch( ));

if(key!=13)

break;

}while(1);

return 0;

}

void midpoint\_circle(const int h,const int k,const int r)

{

int color=getcolor( );

int x=0;

int y=r;

int p=(1-r);

do

{

putpixel((h+x),(k+y),color);

putpixel((h+y),(k+x),color);

putpixel((h+y),(k-x),color);

putpixel((h+x),(k-y),color);

putpixel((h-x),(k-y),color);

putpixel((h-y),(k-x),color);

putpixel((h-y),(k+x),color);

putpixel((h-x),(k+y),color);

x++;

if(p<0)

p+=((2\*x)+1);

else

{

y--;

p+=((2\*(x-y))+1);

}

}while(x<=y);

}

**Output:-**

Central Point of the Circle : (x,y) :

======================================

Enter the value of x\_center =100

Enter the value of y\_center =100

Radius of the Circle : r :

=====================================

Enter the value of r =50

1. **Write a program to draw animated circles**

**Code:**

#include<stdlib.h>

#include<conio.h>

#include<graphics.h>

#include<dos.h>

void main()

{

int x,y,i;

int g=DETECT,d;

initgraph(&g,&d,"c:\\tc\\bgi");

cleardevice();

x=getmaxx()/2;

y=getmaxy()/2;

settextstyle(TRIPLEX\_FONT, HORIZ\_DIR, 3);

setbkcolor(rand());

setcolor(4);

outtextxy(30,100,"Press");

outtextxy(30,130,"any");

outtextxy(30,160,"key");

outtextxy(30,190, "to");

outtextxy(30,220,"Quit");

while (!kbhit())

{

setcolor(rand());

for (int i=0;i<50;i++)

circle(x,y,i );

setcolor(rand());

for (int j=70;j<120;j++)

circle(x,y,j);

setcolor(rand());

for (int k=140;k<190;k++)

circle(x,y,k);

setcolor(rand());

for (int l=210;l<230;l++)

circle(x,y,l);

delay(200);

}

getch();

closegraph();

}

**Output:**

1. **Write a program to implement Flood fill**

**Code:**

# include <iostream.h>

# include <graphics.h>

# include <conio.h>

# include <math.h>

void Flood\_fill(const int,const int,const int,const int);

int main( )

{

int driver=VGA;

int mode=VGAHI;

initgraph(&driver,&mode,"c:\\tc\\Bgi");

setcolor(15);

circle(175,175,40);

Flood\_fill(175,175,10,0);

getch( );

return 0;

}

void Flood\_fill(const int x,const int y, const int fill\_color,const int old\_color)

{

if(getpixel(x,y)==old\_color)

{

putpixel(x,y,fill\_color);

Flood\_fill((x+1),y,fill\_color,old\_color);

Flood\_fill((x-1),y,fill\_color,old\_color);

Flood\_fill(x,(y+1),fill\_color,old\_color);

Flood\_fill(x,(y-1),fill\_color,old\_color);

}

}

**Output:**

1. **Write a program to implement Boundary fill**

**Code:**

# include <iostream.h>

# include <graphics.h>

# include <conio.h>

# include <math.h>

void Boundary\_fill(const int,const int,const int,const int);

int main( )

{

int driver=VGA;

int mode=VGAHI;

initgraph(&driver,&mode,"c:\\tc\\Bgi");

setcolor(10);

circle(175,175,40);

Boundary\_fill(175,175,5,10);

getch( );

return 0;

}

void Boundary\_fill(const int x,const int y, const int fill\_color,const int boundary\_color)

{

if(getpixel(x,y)!=boundary\_color&& getpixel(x,y)!=fill\_color)

{

putpixel(x,y,fill\_color);

Boundary\_fill((x+1),y,fill\_color,boundary\_color);

Boundary\_fill((x-1),y,fill\_color,boundary\_color);

Boundary\_fill(x,(y+1),fill\_color,boundary\_color);

Boundary\_fill(x,(y-1),fill\_color,boundary\_color);

}

}

**Output:**