**PRACTICAL NO : 1(A)**

**AIM: Study and enlist the basic function used for graphics in C / C++ /Python**

**language .Give an example for each of them.**

1. **Arc Function in C:**

**SOURCE CODE:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

arc(100,100,0,135,50);

getch();

return 0;

}

1. **Drawpoly function:**

**SOURCE CODE:**

#include<graphics.h>

#include<conio.h>

main()

{

init gd=DETECT,gm,points[]={320,150,420,300,250,300,320,150};

initgraph(&gd,&gm,"C:\\TC\\BGI");

drawpoly(4,points);

getch();

closegraph();

return 0;

}

**3. outtext/outtextxy Function:**

**Source code:**

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

outtext("press any key to close");

outtextxy(100,100,"graphics");

getch();

closegraph();

}

**4. Circle function in C:**

**SOURCE CODE:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

circle(100,100,50);

getch();

closegraph();

return 0;

}

**5. Cleardevice function in C:**

**SOURCE CODE:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

outtext("Press any key to clear the screen");

getch();

cleardevice();

outtext("Press any key to exit");

getch();

closegraph();

return 0;

}

**6. Floodfill function in C:**

**SOURCE CODE:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

setcolor(RED);

circle(100,100,50);

floodfill(100,100,RED);

getch();

closegraph();

return 0;

}

**7. Set color function**

**Source code:**

#include<graphics.h>

#include<conio.h>

void main()

{

clrscr();

int gd=DETECT,gm;

initgraph(&gd,&gm,"C://TC//BGI");

setcolor(RED);

circle(100,100,80);

getch();

closegraph();

}

**8.setfillstyle function:**

**Source code:**

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm;

initgraph(&gd,&gm,"C://TC//BGI");

setcolor(RED);

circle(200,200,80);

setfillstyle(SOLID\_FILL,GREEN);

floodfill(200,200,RED);

getch();

closegraph();

}

**PRACTICAL NO: 2**

**AIM: DIVIDE YOUR SCREEN INTO FOUR REGION,DRAW CIRCLE,RECTANGLE,ELLIPSE AND HALF ELLIPSE IN EACH REGION WITH APPROPRIATE MESSAGES.**

**SOURCE CODE:**

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm;

int points[]={400,350,450,400,300,400,400,350};

initgraph(&gd,&gm,"C://TC//BGI");

circle(100,100,50);

outtextxy(75,200,"Circle");

rectangle(200,50,350,150);

outtextxy(250,200,"Rectangle");

ellipse(525,100,0,360,100,50);

outtextxy(500,200,"Ellipse");

line(75,300,400,300);

outtextxy(225,325,"Line");

sector(100,400,30,360,50,60);

outtextxy(150,450,"Sector");

drawpoly(4,points);

getch();

closegraph();

}

**PRACTICAL NO: 2(B)**

**AIM: DRAW A SIMPLE HUT ON THE SCREEN.**

**SOURCE CODE:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

setcolor(WHITE);

rectangle(150,180,250,300);

rectangle(250,180,420,300);

rectangle(180,250,220,300);

line(200,100,150,180);

line(200,100,250,180);

line(200,100,370,100);

line(370,100,420,180);

setfillstyle(SOLID\_FILL,BROWN);

floodfill(152,182,WHITE);

floodfill(252,182,WHITE);

setfillstyle(SLASH\_FILL,BLUE);

floodfillstyle(182,252,WHITE);

setfillstyle(HATCH\_FILL,GREEN);

floodfill(200,105,WHITE);

floodfill(210,105,WHITE);

getch();

closegraph();

return 0;

}

**PRACTICAL NO: 2(C)**

**AIM: DRAW SMILEY.**

**SOURCE CODE:**

# include<graphics.h>

#include<conio.h>

void main()

{

clrscr();

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

rectangle(100,50,150,75);

rectangle(200,50,250,75);

rectangle(50,100,100,125);

rectangle(100,125,150,150);

rectangle(150,150,200,175);

rectangle(200,125,250,150);

rectangle(250,100,300,125);

setfillstyle(SOLID\_FILL,YELLOW);

floodfill(102,52,WHITE);

floodfill(202,52,WHITE);

setfillstyle(SOLID\_FILL,MAGENTA);

floodfill(52,102,WHITE);

floodfill(102,127,WHITE);

floodfill(152,152,WHITE);

floodfill(202,127,WHITE);

floodfill(252,102,WHITE);

getch();

closegraph();

}

**PRACTICAL NO: 3**

**AIM: DRAW THE FOLLOWING BASIC SHAPES IN THE CENTER OF THE SCREEN (CIRCLE,RECTANGLE,SECTOR,ELLIPSE,SQUARE,CONCENTRIC CIRCLE,LINE)**

**SOURCE CODE:**

**1.Circle**

**Source code:**

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm,midx,midy;

initgraph(&gd,&gm,"C:\\TC\\BGI");

midx=getmaxx()/2;

midy=getmaxy()/2;

circle(midx,midy,50);

getch();

closegraph();

}

**2. Ellipse**

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm,midx,midy;

initgraph(&gd,&gm,"C:\\TC\\BGI");

midx=getmaxx()/2;

midy=getmaxy()/2;

ellipse(midx,midy,0,360,100,50);

getch();

closegraph();

}

1. **Square**

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm,midx,midy;

initgraph(&gd,&gm,"C:\\TC\\BGI");

midx=getmaxx()/2;

midy=getmaxy()/2;

rectangle (midx-100,midx-100,midx+100,midy+100);

getch();

closegraph();

}

1. **Rectangle**

**Source code:**

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm,midx,midy;

initgraph(&gd,&gm,"C:\\TC\\BGI");

midx=getmaxx()/2;

midy=getmaxy()/2;

ellipse(midx,midy,0,360,100,50);

rectangle(midx-150,midx-100,midx+150,midy+100);

getch();

closegraph();

}

1. **Sector**

**Source code:**

#include<graphics.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm,midx,midy;

initgraph(&gd,&gm,"C:\\TC\\BGI");

midx=getmaxx()/2;

midy=getmaxy()/2;

ellipse(midx,midy,0,360,100,50);

rectangle(midx-150,midx-100,midx+150,midy+100);

sector(midx,midy,50,360,60,80);

getch();

closegraph();

}

**:**

**6. Concentric circle:**

**Source code:**

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm,midx,midy;

initgraph(&gd,&gm,"C://TC//BGI");

midx=getmaxx()/2;

midy=getmaxy()/2;

setcolor(MAGENTA);

circle(midx,midy,50);

setcolor(GREEN);

circle(midx,midy,75);

setcolor(YELLOW);

circle(midx,midy,100);

getch();

closegraph();

}

**7.Line**

**Source code:**

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm,midx,midy;

initgraph(&gd,&gm,"C://TC//BGI");

midx=getmaxx()/2;

midy=getmaxy()/2;

line(midx-100,midy,midx+100,midy);

getch();

closegraph();

}

**PRACTICAL NO: 4(A)**

**AIM: DEVELOP THE PROGRAM FOR DDA LINE DRAWING ALGORITHM**

**SOURCE CODE:**

#include<graphics.h>

#include<stdio.h>

#include<math.h>

#include<dos.h>

void main()

{

float x,y,x1,y1,x2,y2,dx,dy,step;

int i,gd=DETECT,gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

printf("Enter the value of x1 and y1:");

scanf("%f%f",&x1,&y1);

printf("Enter the value of x2 and y2:");

scanf("%f%f",&x2,&y2);

dx=abs(x2-x1);

dy=abs(y2-y1);

if(dx>=dy)

step=dx;

else

step=dy;

dx=dx/step;

dy=dy/step;

x=x1;

y=y1;

i=1;

while(i<step)

{

putpixel(x,y,5);

x=x+dx;

y=y+dy;

i=i+1;

delay(100);

}

closegraph();

}

**PRACTICAL NO: 4(B)**

**AIM: DEVELOP THE PROGRAM FOR BRESENHAM’S LINE DRAWING ALGORITHM.**

**SOURCE CODE:**

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int dx,dy,x,y,p,x1,y1,x2,y2;

int gd,gm;

clrscr();

printf("Enter co-ordinates of first point:");

scanf("%d%d",&x1,&y1);

printf("Enter co-ordinates of second point:");

scanf("%d%d",&x2,&y2);

dx=(x2-x1);

dy=(y2-y1);

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

x=x1;

y=y1;

detectgraph(&gd,&gm);

initgraph(&gd,&gm,"C:\\TC\\BGI");

//putpixel(x,y,WHITE);

while(x<=x2)

{

if(p<0)

{

x=x+1;

y=y;

p=p+2\*(dy);

}

else

{

x=x+1;

y=y+1;

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

}

putpixel(x,y,WHITE);

}

getch();

closegraph();

}

**PRACTICAL NO: 5(A)**

**AIM: DEVELOP THE PROGRAM FOR THE MID-POINT CIRCLE DRAWING ALGORITHM.**

**SOURCE CODE:**

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void pixel(int xc,int yc,int x,int y);

void main()

{

int gd=DETECT,gm,xc,yc,r,x,y,Pk;

clrscr();

initgraph(&gd,&gm,"C:\\Tc\\BGI");

printf("\*\*\* Mid-Point Circle Drawing algorithm \*\*\*\n");

printf("Enter the value of Xc\t");

scanf("%d",&xc);

printf("Enter the value of Yc \t");

scanf("%d",&yc);

printf("Enter the Radius of circle\t");

scanf("%d",&r);

x=0;

y=r;

Pk=1-r;

pixel(xc,yc,x,y);

while(x<y)

{

if(Pk<0)

{

x=x+1;

Pk=Pk+(4\*x)+1;

}

else

{

x=x+1;

y=y-1;

Pk=Pk+(4\*x)-(4\*y)+1;

}

pixel(xc,yc,x,y);

}

getch();

closegraph();

}

void pixel(int xc,int yc,int x,int y)

{

putpixel(xc+x,yc+y,RED);

putpixel(xc+y,yc+x,RED);

putpixel(xc-y,yc+x,RED);

putpixel(xc-x,yc+y,RED);

putpixel(xc-x,yc-y,RED);

putpixel(xc-y,yc-x,RED);

putpixel(xc+y,yc-x,RED);

putpixel(xc+x,yc-y,RED);

}

**PRACTICAL NO:5(B)**

**AIM: Develop The Program For The Bresenham’s Mid-Point Circle Drawing Algorithm**

**SOURCE CODE:**

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void pixel(int xc,int yc,int x,int y);

void main()

{

int gd=DETECT,gm,xc,yc,r,x,y,Pk;

clrscr();

initgraph(&gd,&gm,"C:\\Tc\\BGI");

printf("\*\*\* Mid-Point Circle Drawing algorithm \*\*\*\n");

printf("Enter the value of Xc\t");

scanf("%d",&xc);

printf("Enter the value of Yc \t");

scanf("%d",&yc);

printf("Enter the Radius of circle\t");

scanf("%d",&r);

x=0;

y=r;

Pk=1-r;

pixel(xc,yc,x,y);

while(x<y)

{

if(Pk<0)

{

x=x+1;

Pk=Pk+(4\*x)+6;

}

else

{

x=x+1;

y=y-1;

Pk=Pk+(4\*x)-(4\*y)+10;

}

pixel(xc,yc,x,y);

}

getch();

closegraph();

}

void pixel(int xc,int yc,int x,int y)

{

putpixel(xc+x,yc+y,RED);

putpixel(xc+y,yc+x,RED);

putpixel(xc-y,yc+x,RED);

putpixel(xc-x,yc+y,RED);

putpixel(xc-x,yc-y,RED);

putpixel(xc-y,yc-x,RED);

putpixel(xc+y,yc-x,RED);

putpixel(xc+x,yc-y,RED);

}

**PRACTICAL NO: 6**

**AIM: A) WRITE A PROGRAM TO IMPLEMENT 2D SCALING.**

**B) WRITE A PROGRAM TO PERFORM 2D TRANSLATION.**

**SOURCE CODE:**

#include <graphics.h>

#include <stdio.h>

#include <conio.h>

#include<math.h>

void main()

{

int gm;

int gd=DETECT;

int x1,x2,x3,y1,y2,y3,nx1,nx2,nx3,ny1,ny2,ny3,c;

int sx,sy,xt,yt,r;

float t;

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

printf("\ Program for basic transformations");

printf("\n Enter the points of triangle");

setcolor(1);

scanf("%d%d%d%d%d%d",&x1,&y1,&x2,&y2,&x3,&y3);

line(x1,y1,x2,y2);

line(x2,y2,x3,y3);

line(x3,y3,x1,y1);

getch();

printf("\n 1.Scaling\n 2.Translation\n 3.exit");

printf("\nEnter your choice:");

scanf("%d",&c);

switch(c)

{

case 1:

setcolor(2);

printf("\n Enter the scaling factor");

scanf("%d%d",&sx,&sy);

nx1=x1\*sx;

ny1=y1\*sy;

nx2=x2\*sx;

ny2=y2\*sy;

nx3=x3\*sx;

ny3=y3\*sy;

line(nx1,ny1,nx2,ny2);

line(nx2,ny2,nx3,ny3);

line(nx3,ny3,nx1,ny1);

getch();

case 2:

setcolor(4);

printf("\n Enter the translation factor");

scanf("%d%d",&xt,&yt);

nx1=x1+xt;

ny1=y1+yt;

nx2=x2+xt;

ny2=y2+yt;

nx3=x3+xt;

ny3=y3+yt;

line(nx1,ny1,nx2,ny2);

line(nx2,ny2,nx3,ny3);

line(nx3,ny3,nx1,ny1);

getch();

case 3:

break;

default:

printf("Enter the correct choice");

}

closegraph();

}

**:**

**PRACTICAL NO:7(A)**

**AIM: PERFORM 2D ROTATION ON A GIVEN OBJECT.**

**SOURCE CODE:**

#include <graphics.h>

#include <stdio.h>

#include <conio.h>

#include<math.h>

void main()

{

int gm;

int gd=DETECT;

int x1,x2,x3,y1,y2,y3,nx1,nx2,nx3,ny1,ny2,ny3,c;

int sx,sy,xt,yt,r;

float t;

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

printf("\Program for basic transformations");

printf("\n\Enter the points of triangle");

setcolor(1);

scanf("%d%d%d%d%d%d",&x1,&y1,&x2,&y2,&x3,&y3);

line(x1,y1,x2,y2);

line(x2,y2,x3,y3);

line(x3,y3,x1,y1);

getch();

setcolor(3);

printf("\n Enter the angle of rotation");

scanf("%d",&r);

t=3.14\*r/180;

nx1=abs(x1\*cos(t)-y1\*sin(t));

ny1=abs(x1\*sin(t)+y1\*cos(t));

nx2=abs(x2\*cos(t)-y2\*sin(t));

ny2=abs(x2\*sin(t)+y2\*cos(t));

nx3=abs(x3\*cos(t)-y3\*sin(t));

ny3=abs(x3\*sin(t)+y3\*cos(t));

line(nx1,ny1,nx2,ny2);

line(nx2,ny2,nx3,ny3);

line(nx3,ny3,nx1,ny1);

getch();

closegraph();

}

**PRACTICAL NO:8(A)**

**AIM:WRITE A PROGRAM TO FILL A CIRCLE USING FLOODFILL ALGORITHM.**

**SOURCE CODE:**

#include<stdio.h>

#include<graphics.h>

# include<conio.h>

void floodFill(int x,int y,int oldcolor,int newcolor)

{

if(getpixel(x,y) == oldcolor)

{

putpixel(x,y,newcolor);

floodFill(x+1,y,oldcolor,newcolor);

floodFill(x,y+1,oldcolor,newcolor);

floodFill(x-1,y,oldcolor,newcolor);

floodFill(x,y-1,oldcolor,newcolor);

getch();

}

}

//getpixel(x,y) gives the color of specified pixel

void main()

{

int gm,gd=DETECT,radius;

int x,y;

printf("Enter x and y positions for circle\n");

scanf("%d%d",&x,&y);

printf("Enter radius of circle\n");

scanf("%d",&radius);

initgraph(&gd,&gm,"c:\\tc\\bgi");

circle(x,y,radius);

floodFill(x,y,0,15);

closegraph();

}

**PRACTICAL NO:8(B)**

**AIM: WRITE A PROGRAM TO FILL A CIRCLE BOUNDARY FILL ALGORITHM.**

**SOURCE CODE:**

#include<stdio.h>

#include<graphics.h>

#include<conio.h>

void boundaryfill(int x,int y,int f\_color,int b\_color)

{

if(getpixel(x,y)!=b\_color && getpixel(x,y)!=f\_color)

{

putpixel(x,y,f\_color);

boundaryfill(x+1,y,f\_color,b\_color);

boundaryfill(x,y+1,f\_color,b\_color);

boundaryfill(x-1,y,f\_color,b\_color);

boundaryfill(x,y-1,f\_color,b\_color);

}

}

void main()

{

int gm,gd=DETECT,radius;

int x,y;

printf("Enter x and y positions for circle\n");

scanf("%d%d",&x,&y);

printf("Enter radius of circle\n");

scanf("%d",&radius);

initgraph(&gd,&gm,"C:\\TC\\BGI");

circle(x,y,radius);

boundaryfill(x,y,4,15);

getch();

closegraph();

}

**PRACTICAL NO: 9(A)**

**AIM: PERFORM SMILING FACE ANIMATION USING GRAPHIC FUNCTION.**

**SOURCE CODE:**

#include<graphics.h>

#include<conio.h>

#include<stdlib.h>

main()

{

   int gd = DETECT, gm, area, temp1, temp2, left = 25, top = 75;

   void \*p;

   initgraph(&gd, &gm, "C:**\\**TC**\\**BGI");

   setcolor(YELLOW);

   circle(50, 100, 25);

   setfillstyle(SOLID\_FILL, YELLOW);

   floodfill(50, 100, YELLOW);

   setcolor(BLACK);

   setfillstyle(SOLID\_FILL, BLACK);

   fillellipse(44, 85, 2, 6);

   fillellipse(56, 85, 2, 6);

   ellipse(50, 100, 205, 335, 20, 9);

   ellipse(50, 100, 205, 335, 20, 10);

   ellipse(50, 100, 205, 335, 20, 11);

   area = imagesize(left, top, left + 50, top + 50);

   p = malloc(area);

   setcolor(WHITE);

   settextstyle(SANS\_SERIF\_FONT, HORIZ\_DIR, 2);

   outtextxy(155, 451, "Smiling Face Animation");

   setcolor(BLUE);

   rectangle(0, 0, 639, 449);

   while(!kbhit())

   {

      temp1 = 1 + random (588);

      temp2 = 1 + random (380);

      getimage(left, top, left + 50, top + 50, p);

      putimage(left, top, p, XOR\_PUT);

      putimage(temp1 , temp2, p, XOR\_PUT);

      delay(100);

      left = temp1;

      top = temp2;

   }

   getch();

   closegraph();

   return 0;

}

**PRACTICAL NO: 9(B)**

**AIM: DRAW THE MOVING CAR ON THE SCREEN.**

**SOURCE CODE:**

#include <graphics.h>

#include <dos.h>

#include <conio.h>

main()

{

int i, j = 0, gd = DETECT, gm;

initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");

settextstyle(DEFAULT\_FONT,HORIZ\_DIR,2);

outtextxy(25,240,"Press any key to view the moving car");

getch();

setviewport(0,0,639,440,1);

for( i = 0 ; i <= 420 ; i = i + 10, j++ )

{

rectangle(50+i,275,150+i,400);

rectangle(150+i,350,200+i,400);

circle(75+i,410,10);

circle(175+i,410,10);

setcolor(j);

delay(100);

if( i == 420 )

break;

clearviewport();

}

getch();

closegraph();

return 0;

}