Precatical 2:

Q4: Write a program to draw a circle using BRESENHAM algorithm.

```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
void BRECIRCLE(int xc,int yc,int x,int y)
{
putpixel(xc+x,yc+y,15);
putpixel(xc+x,yc-y,14);
putpixel(xc-x,yc+y,13);
putpixel(xc-x,yc-y,12);
putpixel(xc+y,yc+x,11);
putpixel(xc+y,yc-x,10);
putpixel(xc-y,yc+x,9);
putpixel(xc-y,yc-x,8);
delay(10);
}
void CIRCLE(int xc,int yc,int r)
{
int x=0,y=r,d=3-(2*r);
BRECIRCLE(xc,yc,x,y);
while(x<=y)
{
if(d<=0)
{
d=d+(4*x)+6;
}
else
{
d=d+(4*x)-(4*y)+10;
y=y-1;
}
x=x+1;
BRECIRCLE(xc,yc,x,y);
}
1
}
int main(void)
{
int xc,yc,r;
printf("Enter the values of xc and yc :");
scanf("%d%d",&xc,&yc);
printf("Enter the value of radius :");
scanf("%d",&r);
initwindow(1000,700);
for(int i=0;i<10;i++){
CIRCLE(xc,yc,r);
```

```
r=r-5;
}
getch();
closegraph();
return 0;
}
```

Q5: Write a menu driven program to input user choice as follows:
Display line using DDA algorithm
Display line using BRESENHAM algorithm
Display circle using BRESENHAM algorithm

```
#include<iostream>
#include<graphics.h>
using namespace std;
void BRESENHAM(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;
}}
void EIGHTPOINTS(int xc,int yc,int x,int y)
{
putpixel(xc+x,yc+y,15);
putpixel(xc+x,yc-y,14);
putpixel(xc-x,yc+y,13);
putpixel(xc-x,yc-y,12);
putpixel(xc+y,yc+x,11);
putpixel(xc+y,yc-x,10);
putpixel(xc-y,yc+x,9);
putpixel(xc-y,yc-x,8);
delay(10);
}
void BHC(int xc,int yc,int r)
```

```
{
int x=0,y=r,d=3-(2*r);
EIGHTPOINTS(xc,yc,x,y);
while(x<=y)
{
if(d<=0)
{
d=d+(4*x)+6;
}
3
else
{
d=d+(4*x)-(4*y)+10;
y=y-1;
}
x=x+1;
EIGHTPOINTS(xc,yc,x,y);
}
}i
nt main(){
int C;
cout<<"1=Line using DDA \n";
cout<<"2=Line using BRESENHAM \n";
cout<<"3=Circle using BRESENHAM \n";
cout<<"Enter your choise = ";
cin >> C;
switch (C){
case 1:
int i,x0,x1,y0,y1;
float x,y,dx,dy,steps;
cout<<"Enter co-ordinates of first point: ";
cin >> x0>>y0;
cout<<"Enter co-ordinates of second point: ";
cin>>x1>>y1;
dx=(float)(x1-x0);
dy=(float)(y1-y0);
if(dx>=dy){
steps=dx;
}
else{
steps=dy;
}
dx = dx/steps;
dy = dy/steps;
x = x0;
y = y0;
i = 1;
initwindow(1700,800);
while(i<= steps)
```

```
{
4
putpixel(x, y, 15);
x += dx;
y += dy;
i=i+1;
}
getch();
break;
case 2:
int x00, y00, x01, y01;
cout<<"Enter co-ordinates of first point: ";
cin>>x00>>y00;
cout<<"Enter co-ordinates of second point: ";
cin>>x01>>y01;
initwindow(1700,800);
BRESENHAM(x00, y00, x01, y01);
getch();
break;
case 3:
int xc,yc,r;
printf("Enter the values of xc and yc :");
scanf("%d%d",&xc,&yc);
printf("Enter the value of radius :");
scanf("%d",&r);
initwindow(1700,800);
for(int i=0;i<10;i++){
BHC(xc,yc,r);}
break;
default:
cout<<"Wrong choise";
break;
}r
eturn 0;
getch();
}
```

Q6: Write a program to input coordinates of two line segment. Display both the line and highlight intersection point. Display appropriate message if line segments are parallel or not intersecting.

```
#include<iostream>
#include<graphics.h>
using namespace std;
int main(){
int x00,x01,y00,y01,x10,x11,y10,y11;
cout<<"Enter co-ordinates of first line x0,y0,x1,y1:";
cin >> x00>>y00>>x01>>y01;
cout<<"Enter co-ordinates of second line x0,y0,x1,y1: ";
cin >> x10>>y10>>x11>>y11;
int dx0=abs(x00-x01);
```

```
int dy0=abs(y00-y01);
int dx1=abs(x10-x11);
int dy1=abs(y10-y11);
if(dx0==dx1 && dy0==dy1){
cout <<"Line parellel";
}
initwindow(1700,800);
line(x00,y00,x01,y01);
line(x10,y10,x11,y11);
getch();
}
```

Q8: Display three characters of your name in Gujarati using bitmap method.

```
#include <iostream>
#include <conio.h>
#include <graphics.h>
main()
{
   int i,j,k,x,y;

   int ch1[][10]={ {1,1,1,1,1,1,1,1,1,1},
                   {1,1,1,1,1,1,1,1,1,1},
                   {1,1,0,0,0,0,0,0,1,1},
                   {1,1,0,0,0,0,0,0,1,1},
                   {1,1,1,1,1,1,1,1,1,1},
                   {1,1,1,1,1,1,1,1,1,1},
                   {1,1,0,0,0,0,0,0,0,0},
                   {1,1,0,0,0,0,0,0,0,0},
                   {1,1,0,0,0,0,0,0,0,0},
                   {1,1,0,0,0,0,0,0,0,0}};
   int ch2[][10]={ {0,0,0,0,1,1,0,0,0,0},
                   {0,0,1,1,0,0,1,1,0,0},
                   {0,1,1,0,0,0,0,1,1,0},
                   {0,1,1,0,0,0,0,1,1,0},
                   {0,1,1,0,0,0,0,1,1,0},
                   {0,1,1,1,1,1,1,1,1,0},
                   {0,1,1,0,0,0,0,1,1,0},
                   {0,1,1,0,0,0,0,1,1,0},
                   {0,1,1,0,0,0,0,1,1,0},
                   {0,1,1,0,0,0,0,1,1,0}};
   int ch3[][10]={ {1,1,0,0,0,0,0,0,1,1},
                   {1,1,1,1,0,0,0,0,1,1},
                   {1,1,1,1,0,0,0,0,1,1},
                   {1,1,0,1,1,0,0,0,1,1},
                   {1,1,0,1,1,0,0,0,1,1},
                   {1,1,0,0,1,1,0,0,1,1},
                   {1,1,0,0,1,1,0,0,1,1},
                   {1,1,0,0,0,1,1,0,1,1},
                   {1,1,0,0,0,0,1,1,1,1},
                   {1,1,0,0,0,0,1,1,1,1}};
```

```
   initwindow(900,500);
   for(k=0;k<3;k++)
   {
      for(i=0;i<10;i++)
      {
         for(j=0;j<10;j++)
         {
            if(k==0)
            {
               if(ch1[i][j]==1)
               putpixel(j+250,i+230,10);
            }
            if(k==1)
            {
               if(ch2[i][j]==1)
               putpixel(j+300,i+230,10);
            }
            if(k==2)
            {
               if(ch3[i][j]==1)
               putpixel(j+350,i+230,10);
            }
         }
        delay(100);
      }
   }
   getch();
   closegraph();
}
```

Q9: Modify DDA line algorithm to generate 5 – pixel thick line.

```
#include<iostream>
#include<graphics.h>
using namespace std;
int main(){
int i,x0,x1,y0,y1;
float x,y,dx,dy,steps;
cout<<"Enter co-ordinates of first point: ";
cin >> x0>>y0;
cout<<"Enter co-ordinates of second point: ";
cin>>x1>>y1;
dx=(float)(x1-x0);
dy=(float)(y1-y0);
if(dx>=dy){
steps=dx;
}
else{
steps=dy;
}
dx = dx/steps;
```

```
dy = dy/steps;
x = x0;
y = y0;
i = 1;
initwindow(1700,800);
while(i<= steps)
{
putpixel(x,y-1,15);
putpixel(x-1,y,15);
putpixel(x, y, 15);
putpixel(x+1,y,15);
putpixel(x,y+1,15);
x += dx;
y += dy;
i=i+1;
}
getch();
}
```

Q10: Modify DDA line algorithm to generate line with the pattern(dash line, dotted line, dot-dash line).

```
#include<iostream>
#include<graphics.h>
using namespace std;
int main(){
int i,x0,x1,y0,y1;
float x,y,dx,dy,steps;
cout<<"Enter co-ordinates of first point: ";
cin >> x0>>y0;
cout<<"Enter co-ordinates of second point: ";
cin>>x1>>y1;
dx=(float)(x1-x0);
dy=(float)(y1-y0);
if(dx>=dy){
steps=dx;
}
else{
steps=dy;
}
dx = dx/steps;
dy = dy/steps;
x = x0;
y = y0;
i = 1;
int choise;
cout<<"1=dash line \n";
cout<<"2=dotted line\n";
cout<<"3=dot-dashed line \n";
cout<<"Enter your Choise = ";
cin>>choise;initwindow(1700,800);
```

```
switch(choise){
case 1:
while(i<= steps)
{
for(int j=0;j<=2;j++){
putpixel(x, y, 15);
x += dx;
y += dy;
i=i+1;
}
for(int k=0;k<2;k++){
x += dx;
y += dy;
i=i+1;
}
}
break;
case 2:
while(i<= steps)
{
for(int j=0;j<1;j++){
putpixel(x, y, 15);
x += dx;
y += dy;
i=i+1;
}
for(int k=0;k<2;k++){
x += dx;
y += dy;
i=i+1;
}
}
break;
case 3:
while(i<= steps)
{
for(int j=0;j<=2;j++){
putpixel(x, y, 15);
x += dx;
y += dy;
i=i+1;
}
for(int k=0;k<2;k++){
x += dx;
y += dy;
i=i+1;
} for(int j=0;j<1;j++){
putpixel(x, y, 15);
x += dx;
```

```
y += dy;
i=i+1;
}
for(int k=0;k<2;k++){
x += dx;
y += dy;
i=i+1;
}
}
break;
default:
cout<<"Wrong choise";
}
getch();
}
```

Q11: Write a program to implement display file concept. Insert more than two lines in display file. Display it using display file interpreter program.

```
#include<iostream>
#include<graphics.h>
using namespace std;
struct dfile{
int opcode;
int x,y;
};
void disp(struct dfile df[],int count){
for(int i=0;i<count;i++){
switch(df[i].opcode)
{
case 1:
moveto(df[i].x,df[i].y);
break;
case 2:
lineto(df[i].x,df[i].y);
break;
case 3:
circle(df[i].x,df[i].y,75);
default:
break;
}
delay(300);
}}
main(){
initwindow(1920,1080);
int y=getmaxy();
struct dfile df[]={
1,100,y-100,
2,200,y-200,
2,400,y-200,
2,200,y-500,
```

```
2,100,y-100};
disp(df,15);
getch();
}
```

Q12: Write a program to add N – edge polygon in display file. Display it using display file interpreter program.

```
#include<iostream>
#include<graphics.h>
using namespace std;
struct dfile{
int opcode;
int x,y;
};
void disp(struct dfile df[],int count){
for(int i=0;i<count;i++){
switch(df[i].opcode)
{
case 1:
moveto(df[i].x,df[i].y);
break;
case 2:
lineto(df[i].x,df[i].y);
case 3:
int P,X,Y;
cout<<"Enter N-Edge of ploygon = ";
cin>>P;
moveto(df[i].x,df[i].y);
for(int i=0;i<P;i++){
cout <<"enter cordinates \n";
cin>>X>>Y;
lineto(X,Y);
}l
ineto(df[i].x,df[i].y);
break;
default:
break;
}
delay(300);
}}i
nt main(){
int y=getmaxy(),p;
struct dfile df[]={
3,100,100
};
initwindow(1920,1080);
disp(df,15);
getch();
}
```

Q13: Write a program to display following using display file concept.

```
#include<iostream>
#include<graphics.h>
#include<conio.h>
#include<dos.h>
using namespace std;
struct dfile{
int opcode;
int x,y;
};
void disp(struct dfile df[], int count){
for(int i=0;i<=count;i++){
//delay(300);
switch(df[i].opcode){
case 1:
moveto(df[i].x,df[i].y);
break;
case 2:
lineto(df[i].x,df[i].y);
break;
default:
break;
}
}
}
main(){
initwindow(1920,1080);
struct dfile df[]{
//mountain
1,100,270,
2,300,40,
2,490,300,
1,450,230,
2,640,10,
2,940,330,
1,900,270,
2,1030,110,
2,1230,300,
1,1190,250,
2,1410,70,
2,1530,200,
//river
1,100,270,
2,470,400,
2,1000,380,
2,1318,420,
//second river
1,100,470,
2,450,600,
2,980,580,
```

```
2,1320,620,
2,1390,590,
1,1430,570,
2,1540,530,
//tree,
1,1390,480,
2,1390,650,
2,1430,650,
2,1430,480,
//bot
1,530,470,
2,600,520,
2,710,520,
2,770,470,
2,530,470,
2,530,320,
2,620,470,
1,550,480,
2,750,480,
//man
1,720,480,
2,720,450,
1,700,450,
2,740,450,
//bird
1,810,90,
2,820,100,
2,830,90,
1,1190,80,
2,1200,90,
2,1210,80,
1,1150,110,
2,1160,120,
2,1170,110,
1,1090,80,
2,1100,90,
2,1110,80
};
disp(df,60);
circle(1410,360,115);
circle(720,440,10);
//sun
circle(450,90,50);
getch();
closegraph();
}
```

Q14: Write a program to display any one from the following using display file.
A bus, A car, A windmill, A kite in the sky, A man, A cycle, A train

```
#include<iostream>
#include<graphics.h>
using namespace std;
struct dfile{
int opcode;
int x,y;
};
void disp(struct dfile df[],int count){
for(int i=0;i<count;i++){
switch(df[i].opcode)
{
case 1:
moveto(df[i].x,df[i].y);
break;
case 2:
lineto(df[i].x,df[i].y);
break;
case 3:
circle(df[i].x,df[i].y,75);
default:
break;
}
delay(100);
}
}
main(){
initwindow(1920,1080);
int y=getmaxy();
struct dfile df[]={
1,100,y-150,
2,100,y-300,
2,200,y-300,
2,300,y-450,
2,700,y-450,
2,900,y-300,
2,1100,y-300,
2,1100,y-150,
1,200,y-300,
2,900,y-300,
1,1100,y-150,
2,800,y-150,
3,725,y-150,
1,650,y-150,
2,400,y-150,
3,325,y-150,
1,250,y-150,
2,100,y-150
};
disp(df,150);
```

```
getch();
closegraph();
}
```
Q15: Write a program to input an N- edge polygon in display file (randomly) and a point. Using odd even method display message point is inside or outside. Display polygon and a point on screen.

```
#include<iostream>
#include<graphics.h>
#include <stdlib.h>
#include <bits/stdc++.h>
using namespace std;
struct dfrecord{
int opcode;
int x,y;
};
class Dfile{
public:
dfrecord *ptr;
int size;
Dfile(){
}
Dfile(int s){
size=s;
ptr=new dfrecord[s];
}
void init(){
moveto(100,100);
for(int i=0;i<size-1;i++)
{
ptr[i].opcode=rand()%1+2;
ptr[i].x=rand()%500;
ptr[i].y=rand()%500;
} for(int i=0;i<size-1;i++){
switch(ptr[i].opcode){
case 1:
moveto(ptr[i].x,ptr[i].y);
break;
case 2:
lineto(ptr[i].x,ptr[i].y);
break;
default:
break;
}
}
lineto(100,100);
}
};
main(){
int Size;
```

```
cout<<"Enter side = ";
cin>>Size;
initwindow(500,500);
Dfile df(Size);
df.init();
cout<<"OK";
getch();
closegraph();
}
```