**Ques 1: WAP to implement DDA Algorithm.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#define ROUND(a) ((int)(a+0.5))

void ddaline(int x1, int y1, int x2, int y2){

float xinc, yinc, x=x1, y=y1;

int dx = x2-x1;

int dy = y2-y1;

int length,k=1;

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

{length=abs(dx); }

else

{length=abs(dy);}

xinc= dx/(float)length;

yinc= dy/(float)length;

clrscr();

putpixel(ROUND(x),ROUND(y),WHITE);

while(k<=length)

{

x+=xinc;

y+=yinc;

putpixel(ROUND(x), ROUND(y),WHITE);

delay(50);

k++;

}

}

int main()

{

int x1, x2, y1, y2;

int gd = DETECT, gm;

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

printf("Enter intial point\n");

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

printf("Enter end point\n");

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

ddaline(x1, y1, x2, y2);

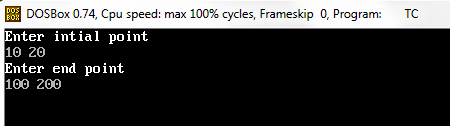
getch();

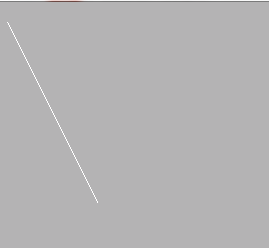
closegraph();

return 0;

}

**OUTPUT**

****

****

**Ques 2: WAP to implement Bresenham Line drawing Algorithm.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#include<conio.h>

void bresenhamline(int x1, int y1, int x2, int y2){

int dx=abs(x2-x1);

int dy=abs(y2-y1);

int twody=2\*dy;

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

int p=twody-dx;

int x,y;

int end;

if(x1<x2) {

x=x1;

y=y1;

end=x2;}

else if(x2>x1) {

x=x2;

y=y2;

end=x1;}

putpixel(x,y,WHITE);

while(x<end){

if(p<0){

x++;

p+=twody;}

else{

x++;y++;

p+=twodydx;}

putpixel(x,y,WHITE);

delay(50);}}

int main( ){

int x1, x2, y1, y2;

int gd = DETECT, gm;

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

printf("Enter intial point\n");

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

printf("Enter end point\n");

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

clrscr();

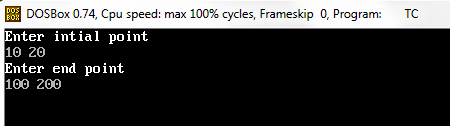
bresenhamline(x1, y1, x2, y2);

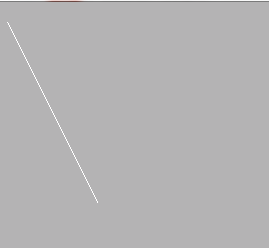
getch();

closegraph();

return 0;}

**OUTPUT**

****

****

**Ques 2: WAP to implement Bresenham Circle Drawing Algorithm (Midpoint circle algorithm).**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#include<conio.h>

void plotcircle(int xc,int yc,int x,int y){

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

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

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

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

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

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

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

putpixel(xc-y,yc-x,WHITE);}

void circledraw(int xc, int yc, int r){

int x=0;

int y=r;

int p=1-r;

plotcircle(xc,yc,x,y);

while(x<y){

if(p<0) {

x++;

p+=2\*x+1; }

else{

x++;

y--;

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

plotcircle(xc,yc,x,y);}}

int main( ){

int x,y,r;

int gd = DETECT, gm;

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

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

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

printf("Enter radius \n");

scanf("%d", &r);

clrscr();

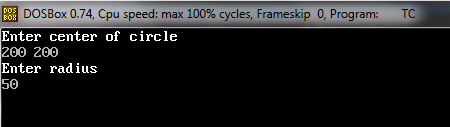
circledraw(x,y,r);

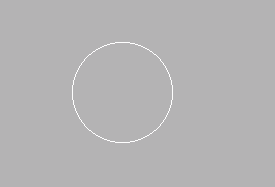
getch();

closegraph();

return 0;}

**OUTPUT**





**Ques 4: WAP to implement Ellipse Drawing Algorithm.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#include<conio.h>

#define ROUND(a)((int)(a+0.5))

void plotellipse(int xc,int yc,int x,int y){

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

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

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

putpixel(xc-x,yc-y,WHITE);}

void ellipsedraw(int xc, int yc, int rx,int ry){

int rx2 = rx\*rx, ry2 = ry\*ry;

int tworx2 = 2\*rx2, twory2=2\*ry2;

int x=0, y=ry;

int p= ROUND(ry2 + 0.25\*rx2 - rx2\*ry);

int px= 0;

int py = tworx2\*ry;

plotellipse(xc,yc,x,y);

//REGION 1

while(px < py) {

x++;

px += twory2;

if(p<0)

p+= px + ry2;

else{

y--;

py -= tworx2;

p+= px+ry2-py;

}

plotellipse(xc,yc,x,y);

}

//REGION 2

p = ROUND(ry2\*(x+0.5)\*(x+0.5) + rx2\*(y-1)\*(y-1) - rx2\*ry2);

while (y>0) {

y--;

py -= tworx2;

if(p>0){

p+= rx2-py;

}

else{

x++;

px+=twory2;

p+=px-py+rx2;

}

plotellipse(xc,yc,x,y);

}

}

int main( )

{

int x,y,rx,ry;

int gd = DETECT, gm;

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

printf("Enter center of ellipse\n");

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

printf("Enter radius rx\n");

scanf("%d", &rx);

printf("Enter radius ry\n");

scanf("%d", &ry);

clrscr();

ellipsedraw(x,y,rx,ry);

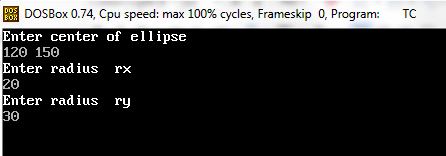
getch();

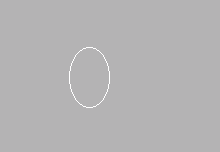
closegraph();

return 0;

}

**OUTPUT**





**Ques 5: WAP to implement Boundary Fill Algorithm.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#include<conio.h>

void boufill(int x,int y,int f,int b)

{

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

{

putpixel(x,y,f);

boufill(x+1,y,f,b);

boufill(x-1,y,f,b);

boufill(x,y+1,f,b);

boufill(x,y-1,f,b);

}

}

int main( )

{

int x,y,r;

int gd = DETECT, gm;

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

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

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

printf("Enter radius \n");

scanf("%d", &r);

circle(x,y,r);

printf("Press key to fill colour\n");

boufill(x,y,15,WHITE);

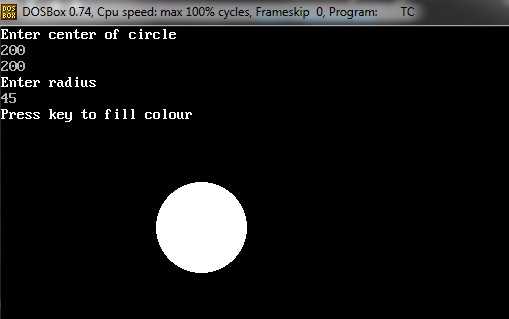
getch();

closegraph();

return 0;

}

**OUTPUT**



**Ques 6: WAP to implement Fill Algorithm.**

#include<stdio.h>

#include<stdlib.h>

#include<graphics.h>

#include<conio.h>

void flood(int x,int y,int f,int old)

{

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

{

putpixel(x,y,f);

flood(x+1,y,f,old);

flood(x-1,y,f,old);

flood(x,y+1,f,old);

flood(x,y-1,f,old);

}

}

int main( )

{

int x,y,r;

int gd = DETECT, gm;

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

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

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

printf("Enter radius \n");

scanf("%d", &r);

circle(x,y,r);

flood(x,y,15,BLACK);

getch();

closegraph();

return 0;

}

**OUTPUT**

