CGA

1]

B) Draw a co-ordinate axis at the center of the screen.

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
void main()
{
int gdriver=DETECT,gmode;
int xcen,ycen;
initgraph(&gdriver,&gmode,"C:\\TC\\BGI");
xcen=getmaxx()/2;
ycen=getmaxy()/2;
line(xcen,0,xcen,getmaxy());
line(0,ycen,getmaxx(),ycen);
getch();
closegraph();
}
```

2]

A) Divide your screen into four region, draw circle, rectangle, ellipse and half ellipse in each region with appropriate message.

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
void main()
{
int gdriver=DETECT,gmode;
int xmax,ymax;

initgraph(&gdriver,&gmode,"C:\\TURBOC3\\BGI");
xmax=getmaxx()/2;
ymax=getmaxy()/2;
line(xmax,0,xmax,getmaxy());
line(0,ymax,getmaxx(),ymax);
circle(170,125,100);
outtextxy(150,125,"circle");
rectangle(58,251,304,392);
outtextxy(130,300,"rectangle");
arc(500,150,45,135,100);
outtextxy(470,150,"half ellipse");
ellipse(500,300,0,360,75,25);
```

```
outtextxy(470,300,"ellipse");
getch();
closegraph();
}
```

B) Draw a simple hut on the screen.

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
void main()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TC\\BGI");
printf("\t\t*************HUT***************");
line(150,100,50,200);
line(150,100,350,100);
line(150,100,300,200);
line(300,200,500,200);
line(350,100,500,200);
line(50,200,300,200);
rectangle(50,400,300,200);
rectangle(300,200,500,400);
rectangle(130,250,230,400);
getch();
closegraph();
}
```

3]

A) i. Circle ii. Rectangle iii. Square iv. Concentric Circles v. Ellipse vi. Line

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
void main()
{
int gdriver=DETECT,gmode;
int xcen,ycen,top,left,right,bottom;
initgraph(&gdriver,&gmode,"C:\\TC\\BGI");
xcen=getmaxx()/2;
ycen=getmaxy()/2;
circle(xcen,ycen,100);
circle(xcen,ycen,150);
left=getmaxx()/2-80;
top=getmaxy()/2-50;
right= getmaxx()/2+80;
```

```
bottom=getmaxy()/2+50;
rectangle(left,top,right,bottom);
rectangle(left-50,top-50,right+50,bottom+50);
line(xcen,0,xcen,getmaxy());
ellipse(xcen,ycen,0,360,75,25);
getch();
closegraph();
}
```

4]

A) Develop the program for DDA Line drawing algorithm.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main()
{
int xa,ya,xb,yb,gd,gm;
gd=DETECT;
initgraph(&gd,&gm,"c:\\TC\\bgi");
printf("\nEnter the Value of xa:");
scanf("%d",&xa);
printf("\nEnter the Value of ya:");
scanf("%d",&ya);
printf("\nEnter the Value of xb:");
scanf("%d",&xb);
printf("\nEnter the Value of yb:");
scanf("%d",&yb);
DDA(xa,ya,xb,xb);
getch();
}
DDA(xa,ya,xb,yb)
{
int k,dx,dy,steps;
float x,y,xinc,yinc;
dx=abs(xb-xa);
dy=abs(yb-ya);
if(dx>=dy)
steps=dx;
else
 steps=dy;
 xinc=dx/steps;
 yinc=dy/steps;
 x=xa;
 y=ya;
 putpixel(ceil(x),ceil(y),15);
 for(k=1;k<=steps;k++)
 {
 x=x+xinc;
```

```
 y=y+yinc;
 putpixel(x,y,15);
 }

 return 0;
 }
```

B) Develop the program for Bresenham's Line drawing algorithm.

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main()
{
     int gd=DETECT,gm;
     int x1,x2,y1,y2;
     int dx,dy,i,e;
     float x,y;
     clrscr();
     initgraph(&gd,&gm,"C:\\tc\\bgi");
     printf("Enter the starting point x1:");
     scanf("%d",&x1);
     printf("Enter the starting point y1:");
     scanf("%d",&y1);
     printf("Enter the starting point x2:");
     scanf("%d",&x2);
     printf("Enter the starting point y2:");
     scanf("%d",&y2);

     dx=x2-x1;
     dy=y2-y1;
     x=x1;
     y=y1;
     e=2*dy-dx;
     i=1;
     cleardevice();
     a:putpixel(ceil(x),ceil(y),WHITE);
     while(e>=0)
     {
        y=y++;
        e=e-2*dx;
     }
     x++;
     e=e+2*dy;
     i++;
     if(x<=x2)
     goto a;
     getch();
```

```
}
```
5]

A) Develop the program for the mid-point circle drawing algorithm.

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
#include<math.h>
#include<dos.h>
void main()
{
int x,y,x_mid,y_mid,radius,dp;
int gm,gd=DETECT;
clrscr();
initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
printf("******* mid point circle drawing algorithm****\n\n");
printf("\n ENTER THE COORDINATES:");
scanf("%d %d",&x_mid,&y_mid);
printf("\n enter the radius=");
scanf("%d",&radius);
x=0;
y=radius;
dp=1-radius;
do
{
putpixel(x_mid+x,y_mid+y,YELLOW);
putpixel(x_mid+y,y_mid+x,YELLOW);
putpixel(x_mid-y,y_mid+x,YELLOW);
putpixel(x_mid-x,y_mid+y,YELLOW);
putpixel(x_mid-x,y_mid-y,YELLOW);
putpixel(x_mid-y,y_mid-x,YELLOW);
putpixel(x_mid+y,y_mid-x,YELLOW);
putpixel(x_mid+x,y_mid-y,YELLOW);
delay(100);
if(dp<0)
{
dp=dp+(2*x)+1;
}
else
{
y=y-1;
dp=dp+(2*x)-(2*y)+1;
}
x=x+1;
}
while(y>x);
getch();
}
```

B) Develop the program for the mid-point ellipse drawing algorithm

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void main()
{
	long x,y,x_center,y_center;
	long a_sqr,b_sqr,fx,fy,d,a,b,tmp1,tmp2;
	int gd,gm;
	gd=DETECT;
	initgraph(&gd,&gm,"C:\\TURBOC3\\BGI ");
	printf("****** MID POINT ELLIPSE ALOGORITHM ******");
	printf("\n\n Enter coordinate x and y =");
	scanf("%ld%ld",&x_center,&y_center);
	printf("\n\n Enter values x-radius and y_radius =");
	scanf("%ld%ld",&a,&b);
	x=0;
	y=b;
	a_sqr=a*a;
	b_sqr=b*b;
	fx=2*b_sqr*x;
	fy=2*a_sqr*y;
	d=b_sqr-(a_sqr*b)+(a_sqr*0.25);
	do{
	putpixel(x_center+x,y_center+y,YELLOW);
	putpixel(x_center-x,y_center-y,YELLOW);
	putpixel(x_center+x,y_center-y,YELLOW);
	putpixel(x_center-x,y_center+y,YELLOW);
	if(d<0)
	{
	d=d+fx+b_sqr;
	}
	else
	{
	y=y-1;
	d=d+fx+-fy+b_sqr;
	fy=fy-(2*a_sqr);
	}
	x=x+1;
	fx=fx+(2*b_sqr);
	delay(100);
	}
	while(fx<fy);
	tmp1=(x+0.5)*(x+0.5);
	tmp2=(y-1)*(y-1);
	d=b_sqr*tmp1+a_sqr*tmp2-(a_sqr*b_sqr);
```

```
        do
        {
        putpixel(x_center+x,y_center+y,YELLOW);
        putpixel(x_center-x,y_center-y,YELLOW);
        putpixel(x_center+x,y_center-y,YELLOW);
        putpixel(x_center-x,y_center+y,YELLOW);
        if(d>=0)
        d=d-fy+a_sqr;
        else
        {
        x=x+1;
        d=d+fx-fy+a_sqr;
        fx=fx+(2*b_sqr);
        }
        y=y-1;
        fy=fy-(2*a_sqr);
        }
        while(y>0);
        getch();
        closegraph();
}
```

6]

A) Write a program to implement 2D scaling.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
int gd=DETECT,gm;
int n,x[100],y[100],i;
float sx,sy;
void draw()
{
for(i=0;i<n-1;i++)
{
line(x[i],y[i],x[i+1],y[i+1]);
}
line(x[0],y[0],x[n-1],y[n-1]);
}
void scale()
{
for(i=0;i<n;i++)
{
x[i]=x[0]+(int)((float)(x[i]-x[0])*sx);
y[i]=y[0]+(int)((float)(y[i]-y[0])*sy);
}
}
```

```
void main(){
 printf("Enter the number of sides: ");
 scanf("%d", &n);
 printf("Enter coordinates x and y: ");
 for(i = 0; i<n; i++)
  scanf("%d%d", &x[i], &y[i]);
 printf("Enter scale factors (sx and sy): ");
 scanf("%f%f", &sx, &sy);
 initgraph(&gd, &gm, "C:\\TC\\BGI");
 setcolor(RED);
 draw();
 getch();

 scale();
 setcolor(BLUE);
 draw();
 getch();
}
```

B) Write a program to perform 2D translation

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
int gd=DETECT,gm;
int n,x[100],y[100],i;
float tx,ty;
void draw()
{
  for(i=0;i<n-1;i++)
  {
   line(x[i],y[i],x[i+1],y[i+1]);
  }
  line(x[0],y[0],x[n-1],y[n-1]);
}
void translate()
{
  for(i=0;i<n;i++)
  {
   x[i]=x[i]+tx;
   y[i]=y[i]+ty;
  }
}
void main()
{
 printf("Enter the number of sides:");
 scanf("%d",&n);
 printf("Enter co-ordinates x,y for each point:");
```

```
 for(i=0;i<n;i++)
  scanf("%d%d",&x[i],&y[i]);
 printf("Enter translation factors:(tx,ty)");
 scanf("%f%f",&tx,&ty);
 initgraph(&gd,&gm,"c:\\TC\\BGI");
 setcolor(RED);
 draw();
 getch();
 translate();
 setcolor(BLUE);
 draw();
 getch();
}
```

7]

A) Perform 2D Rotation on a given object.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
int gd=DETECT,gm;
int n,x[100],y[100],i,xp,yp,degree;
float radian;
void rotation();
void draw();
void degToRad()
{
radian=(float)degree*3.14f/180;
}
void main()
{
printf("Enter number of side:");
scanf("%d",&n);
printf("Enter the coordinates: x,y for each point");
for(i=0;i<n;i++)
scanf("%d%d",&x[i],&y[i]);
printf("\nEnter pivot point");
scanf("%d%d",&xp,&yp);
printf("\nEnter rotation angle:");
scanf("%d",&degree);
degToRad();
initgraph(&gd,&gm,"C:\\TURBOC3C\\BGI");
cleardevice();
setcolor(RED);
draw();
getch();

rotation();
```

```
setcolor(BLUE);
draw();
getch();
}

void draw()
{
for(i=0;i<n-1;i++)
{
line(x[i],y[i],x[i+1],y[i+1]);
}
line(x[0],y[0],x[n-1],y[n-1]);
}
void rotation()
{
float t,v;
for(i=0;i<n;i++)
{
t=x[i]-xp;
v=y[i]-yp;
x[i]=xp+floor(t*cos(radian)-v*sin(radian));
y[i]=yp+floor(v*cos(radian)+t*sin(radian));
}
}
```

8]

A) Write a program to implement Cohen-Sutherland clipping.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
#include<process.h>
int code1[4],code2[4],r1,r2,r3,r4,x1,x2,y1,y2,i,n;
void draw(int,int,int,int);
void clip(int,int,int,int);

void draw(int a1,int b1,int a2,int b2)
{

x1=a1,y1=b1,x2=a2,y2=b2;
line(x1,y1,x2,y2);
}
void clip(int r1,int r2,int r3,int r4)
{
int cnt=0,ch=0,i,count=0;
float x,y,dx,dy;
float m;
```

```
int temp[4];
while(ch<2)
{
dx=x2-x1;
dy=y2-y1;
if(dy==0)
dy++;
if(dx==0)
dx++;
m=dy/dx;
if(x1<r1)
code1[0]=1;
if(x1>r3)
code1[1]=1;
if(y1<r2)
code1[2]=1;
if(y1>r4)
code1[3]=1;
if(x2<r1)
code2[0]=1;
if(x2>r3)
code2[1]=1;
if(y2>r2)
code2[2]=1;
if(y2>r4)
code2[3]=1;
if(cnt<=0)
{
cnt++;
clearviewport();
outtextxy(50,50,"Line Before Clipping");
rectangle(r1,r2,r3,r4);
draw(x1,y1,x2,y2);
getch();
}
for(i=0;i<3;i++)
temp[i]=code1[i]&&code2[i];
for(i=0;i<3;i++)
if(temp[i]!=0)
count++;
if(count!=0)
{
outtextxy(100,440,"Line Cannot Be Clipped");
getch();
exit(0);
}
if(code1[0]==1)
```

```
{
y1=y1+(r1-x1)*m;
x1=r1;
}
if(code1[1]==1)
{
y1=y1+((r3-x1)*m);
y1=r3;
}
if(code1[2]==1)
{
x1=x1+((r2-y1)/m);
y1=r2;
}
if(code1[3]==1)
{
x1=x1+(r4-y1)/m;
y1=r4;
}
if(code2[0]==1)
{
y2=y2+((r2-x2)*m);
x2=r1;
}
if(code2[1]==1)
{
y2=y2+((r3-x2)*m);
x2=r3;
}
if(code2[2]==1)
{
x2=x2+((r2-y2)/m);
y2=r2;
}
if(code2[3]==1)
{
x2=x2+((r4-y2)/m);
y2=r4;
}
clearviewport();
for(i=0;i<=3;i++)
{
code1[1]=0;
code2[1]=0;
}
ch++;
}
```

```
outtextxy(50,50,"Line After Clipping");
rectangle(r1,r2,r3,r4);
draw(x1,y1,x2,y2);
}

void main()
{
int gd=DETECT,gm;
int x1,y1,x2,y2,r1,r2,r3,r4;
clrscr();
initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
printf("\nEnter the rectangle top left x:\t");
scanf("\n%d",&r1);
printf("\nEnter the rectangle top left y:\t");
scanf("\n%d",&r2);
printf("\nEnter the rectangle bottom right x:\t");
scanf("\n%d",&r3);
printf("\nEnter the rectangle bottom right y:\t");
scanf("\n%d",&r4);
printf("\nEnter the line co-ordinate x1: \t");
scanf("\n%d",&x1);
printf("\nEnter the line co-ordinate y1: \t");
scanf("\n%d",&y1);
printf("\nEnter the line co-ordinate x2: \t");
scanf("\n%d",&x2);
printf("\nEnter the line co-ordinate y2: \t");
scanf("\n%d",&y2);
cleardevice();
rectangle(r1,r2,r3,r4);
draw(x1,y1,x2,y2);
clip(r1,r2,r3,r4);
getch();
}
```

9]

A) Write a program to fill a circle using Flood Fill Algorithm.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<dos.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);
}
}

int 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:\\TURBOC3\\BGI");
circle(x,y,radius);
floodFill(x,y,0,15);
delay(10);
getch();
return 0;
}
```

B) Write a program to fill a circle using Boundary Fill Algorithm

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<dos.h>

void boundaryfill(int x,int y,int f_color,int b_color)
{
if(getpixel(x,y) !=b_color && getpixel(x,y)!=f_color)
{
delay(5);
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);
}
}

int 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:\\TURBOC3\\BGI");
circle(x,y,radius);
boundaryfill(x,y,4,15);
delay(10);
closegraph();
return 0;
}
```

10]

A) Develop a simple text screen saver using graphics functions

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<stdlib.h>
void main()
{
int gd=DETECT,gm,x=600;
initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
for(x=0;x<250;x++)
{
x %=250;
setcolor(random(16));
circle(random(635),random(70),50);
circle(random(635),random(70),50);
circle(random(635),random(70),50);
circle(random(635),random(70),50);
circle(random(635),random(70),50);
clearviewport();
settextstyle(1,0,5);
setcolor(RED);
outtextxy(50,415-2*x,"*WORLD*");
setcolor(GREEN);
outtextxy(200,415-2*x,"*of*");
setcolor(YELLOW);
settextstyle(3,0,5);
outtextxy(350,415-2*x,"*graphics*");
}
getch();
}
```

B) Perform smiling face animation using graphic functions.

```
#include<conio.h>
#include<graphics.h>
#include<stdlib.h>
#include<dos.h>
```

```
int main()
{
int gd=DETECT,gm,area,temp1,temp2,left=25,top=75;
void *p;
initgraph(&gd,&gm,"C:\\TURBOC3\\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,"Smilling 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);

delay(100);
left=temp1;
top=temp2;
}
getch();
closegraph();
return 0;
}
```

C）Draw the moving car on the screen

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<stdlib.h>
void main()
{
```

```
int gd=DETECT,gm;
int i,maxx,midy;

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

maxx=getmaxx();
midy=getmaxy()/2;
for(i=0;i<maxx-150;i=i+5)
{
cleardevice();
setcolor(WHITE);
line(0,midy+37,maxx,midy+37);
setcolor(YELLOW);
setfillstyle(SOLID_FILL,RED);
line(i,midy+23,i,midy);
line(i,midy,40+i,midy-20);
line(40+i,midy-20,80+i,midy-20);
line(80+i,midy-20,100+i,midy);
line(100+i,midy,120+i,midy);
line(120+i,midy,120+i,midy+23);
line(0+i,midy+23,18+i,midy+23);
arc(30+i,midy+23,0,180,12);
line(42+i,midy+23,78+i,midy+23);
arc(90+i,midy+23,0,180,12);
line(102+i,midy+23,120+i,midy+23);
line(28+i,midy,43+i,midy-15);
line(43+i,midy-15,57+i,midy-15);
line(57+i,midy-15,57+i,midy);
line(57+i,midy,28+i,midy);
line(62+i,midy-15,77+i,midy-15);
line(77+i,midy-15,92+i,midy);
line(92+i,midy,62+i,midy);
line(62+i,midy,62+i,midy-15);
floodfill(5+i,midy+22,YELLOW);
setcolor(BLUE);
setfillstyle(SOLID_FILL,DARKGRAY);
circle(30+i,midy+25,9);
circle(90+i,midy+25,9);
floodfill(30+i,midy+25,BLUE);
floodfill(90+i,midy+25,BLUE);
delay(100);
}
getch();
closegraph();
}
```