1. Draw the following pattern using standard graphics library:

a. Block Diagram of Computer

b. Display Flag of India

c. Flow Chart Symbols, DFD Symbols, ER-Diagram Symbols

a) Block Diagram of Computer

```
#include<iostream.h>
#include<graphics.h>
#include<conio.h>
main()
{
int gd=DETECT,gm,x,y,i=2,j;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
setcolor(i);
rectangle(100,50,500,400);
setcolor(i+1);
rectangle(120,170,210,300);
line(210,235,260,235);
line(260,235,255,230);
line(260,235,255,240);
setcolor(i=2);
outtextxy(145,220,"INPUT");
outtextxy(140,235,"DEVICE");
setcolor(i+4);
rectangle(260,120,350,330);
line(260,190,350,190);
line(260,260,350,260);
line(350,235,400,235);
line(400,235,395,230);
```

```
line(400,235,395,240);
setcolor(YELLOW);
outtextxy(285,150,"memory");
outtextxy(290,225,"ALU");
outtextxy(275,280,"controll");
outtextxy(290,295,"unit");
setcolor(i+8);
setcolor(i+6);
rectangle(400,170,480,300);
setcolor(i+9);
outtextxy(420,220,"output");
outtextxy(420,235,"Device");
setcolor(BLUE);
settextstyle(TRIPLEX_FONT, HORIZ_DIR,2);
outtextxy(150,70,"block diagram of computer*");
setcolor(WHITE);
outtextxy(285,350,"CPU");
getch();
return(0);
}
```

b. Flag of India

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

int main(){
        int gd = DETECT;
        int gm;
```

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

        rectangle(100,100,300,175);
        line(100,125,300,125);
        line(100,150,300,150);
        line(100,100,100,300);
        rectangle(50,300,150,325);
        rectangle(25,325,175,350);
        rectangle(0,350,200,375);

        circle(200,138,12);    //Ashok Chakra Circle
        line(188,138,212,138); //Horizontal Line in Ashok Chakra
        line(200,125,200,150); //Vertical Line in Ashok Chakra
        line(200,138,190,147);
        line(200,138,210,146);
        line(200,138,210,130);
        line(200,138,193,130);


        getch();
        closegraph();

}
```

C. Flow Chart Symbols

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
#include<dos.h>
void main()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\Turboc3\\BGI");
setcolor(WHITE);
line(40,10,120,10);
line(40,50,120,50);
arc(40,30,90,270,20);
arc(120,30,270,90,20);
line(50,70,160,70);
line(40,120,150,120);
line(50,70,40,120);
line(160,70,150,120);
rectangle(10,170,130,230);
line(70,250,20,300);
line(70,250,120,300);
line(20,300,70,350);
line(120,300,70,350);
circle(70,420,20);
line(70,400,70,370);
line(70,370,80,380);
circle(120,420,20);
line(120,440,120,470);
line(120,470,110,460);
```

```
line(120,470,130,460);
line(250,20,310,20);
line(310,20,300,10);
line(310,20,300,30);
line(310,30,310,80);
line(310,80,300,70);
line(310,80,320,70);
line(300,80,250,80);
line(250,80,260,70);
line(250,80,260,90);
line(250,25,250,75);
line(250,25,240,35);
line(250,25,260,35);
line(250,170,350,170);
line(250,200,350,200);
line(250,170,225,185);
line(350,170,375,185);
line(225,185,250,200);
line(375,185,350,200);
setcolor(YELLOW);
outtextxy(45,25,"START/END");
outtextxy(90,380,"CONNECTOR");
outtextxy(40,300,"DECISION");
outtextxy(20,190,"PROCESSING");
outtextxy(55,90,"INPUT/OUTPUT");
outtextxy(250,100,"FLOW LOOP");
outtextxy(270,185,"FOR LOOP");
getch();
closegraph();
}
```

2. Implement Bresenham's Line Drawing Algorithm

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void main()
{
int dx,dy,x,y,e,x1,y1,x2,y2;
int gd=DETECT,gm=DETECT;
initgraph(&gd,&gm,"c:\\turboc3\\bgi ");
clrscr();
printf("\n Enter the co-ordinates of first line :");
scanf("%d%d",&x1,&y1);
printf("\n Enter the co-ordinates of second line:=");
scanf("%d%d",&x2,&y2);
dx=abs(x2-x1);
dy=abs(y2-y1);
x=x1;
y=y1;
e=2*dy-dx;
putpixel(x,y,RED);
while(x<=x2)
{
if(e<0)
{
x=x+1;
y=y+1;
e=e+2*(dy);
```

```
}
else
{
x=x+1;
y=y+1;
e=e+2* (dy-dx);
}
putpixel(x,y,RED);
}
getch();
closegraph();
}


void pixel (float a,float b)
{
putpixel (x+a,y+b,1);
putpixel (x-a,y-b,2);
putpixel (x+a,y-b,3);
putpixel (x-a,y+b,4);
putpixel(x+b,y+a,5);
putpixel(x-b,y-a,6);
putpixel(x+b,y-a,7);
putpixel(x-b,y+a,8);
}
```

3. Implement Bresenham's Circle Drawing Algorithm

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
int x,y,r;
float a,b,d;
void bresan();
void pixel(float a,float b);
void main()
{
int gm,gr;
clrscr();
detectgraph(&gm,&gr);
initgraph(&gm,&gr,"c:\\turboc3\\bgi"); printf("\Enter the center point of the circle:\n");
scanf("%d%d",&x,&y);
printf("\Enter the radius of circle:\n");
scanf("%d",&r);
bresan();
getch();
closegraph();
}
void bresan()
{
a=0;
b=r;
d=3-2*r;
while(a<=b)
```

```
{
a++;
if(d>=0)
{
b--;
d=d+10+4*(a-b);
}
else
d=d+(4*a)+6;
pixel(a,b);
}
}
void pixel (float a,float b)
{
putpixel (x+a,y+b,1);
putpixel (x-a,y-b,2);
putpixel (x+a,y-b,3);
putpixel (x-a,y+b,4);
putpixel(x+b,y+a,5);
putpixel(x-b,y-a,6);
putpixel(x+b,y-a,7);
putpixel(x-b,y+a,8);
}
```

4. Implement DDA line Drawing Algorithm

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
#include<dos.h>
void main()
{
float x,y,x1,y1,x2,y2,dx,dy,step;
int i=1,gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\turboc3\\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=x2-x1;
dy=y2-y1;
if(dx>=dy)
step=dx;
else
step=dy;
dx=dx/step;
dy=dy/step;
x=x1;
y=y1;
while(i<=step)
{
}
```

```
putpixel(x,y,5);
x=x+dx;
y=y+dy;
i=i+1;
delay(100);
closegraph();
getch();
}
```

5. Implementing Translation transformation on polygon

```
#include<graphics.h>
#include<stdlib.h>
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
int gm,gd=DETECT;
int x1,x2,x3,y1,y2,y3,nx1,nx2,nx3,ny1,ny2,ny3;
int tx,ty,c;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
printf("\t*** Programe for basic transformations ***\n");
printf("\n\tEnter the points of triangle:");
setcolor(3);
scanf("%d%d%d%d%d%d",&x1,&x2,&x3,&y1,&y2,&y3);
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x3,y3,x1,y1);
getch();
printf("\n1.Transaction,\n2.exit"); printf("\nEnter your choice:");
scanf("%d",&c);
switch(c)
{
case 1: printf("\nEnter the transaction factor:");
scanf("%d%d",&tx,&ty);
nx1=x1+tx;
ny1=y1+ty;
```

```
nx2=x2+tx;
ny2=y2+ty;
nx3=x3+tx;
ny3=y3+ty;
line(nx1,ny1,nx2,ny2);
line(nx2,ny2,nx3,ny3);
line(nx3,ny3,nx1,ny1);
getch();
case 2: exit :
break;
default:printf("Your enter choice is not valid");
break;
}
closegraph();
}
```

6. Implementing Scaling transformation on polygons.

```
#include<graphics.h>
#include<stdlib.h>
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
int gm,gd=DETECT;
int x1,x2,x3,y1,y2,y3,nx1,nx2,nx3,ny1,ny2,ny3,c;
int sx,sy;
float t;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
printf("\t*** Programe for basic teansformations***\n");
printf("\n\tEnter the points of triangle");
setcolor (3);
scanf("%d%d%d%d%d%d",&x1,&x2,&x3,&y1,&y2,&y3);
line(x1,y1,x2,y2);
line (x2,y2,x3,y3);
line(x3,y3,x1,y1);
getch();
printf("\n1.Scalling,\n2.exit");
printf("\nEnter Your Choice :");
scanf("%d",&c);
switch(c)
{
case 1: printf("\nEnter the scalling factor:");
printf("sx,sy");
```

```
scanf("%d%d",&sx,&sy);
nx1=x1*sx;
ny1=y2*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: exit:
break;
default:printf("Your enter choice is not valid");
break;
}
closegraph();
}
```

7. Implementing Rotation transformation on polygons.

```
#include<graphics.h>
#include<stdlib.h>
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
int gm,gd=DETECT;
int x1,x2,x3,y1,y2,y3,nx1,nx2,nx3,ny1,ny2,ny3,c;
int sx,sy,tx,ty,r;
float t;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
printf("\t*** Programe for basic teansformations ***\n");
printf("\n\tEnter the points of triangle");
setcolor(3);
scanf("%d%d%d%d%d%d",&x1,&x2,&x3,&y1,&y2,&y3);
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x3,y3,x1,y1);
getch();
printf("\n1.rotation,\n2.exit");
printf("\nEnter your choice:");
scanf("%d",&c);
switch(c)
{
case 1: printf("\nEnter 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();
case 2: exit:
break;
default:printf("Your enter choice is not valid");
break;
}
closegraph();
}
```

8. Implement Cohen-Sutherland line clipping algorithm.

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<graphics.h>
#include<dos.h>
typedef struct coordinate
{
int x,y;
char code[4];
}PT;
void drawwindow();
void drawline(PT p1,PT p2);
PT setcode(PT p);
int visibility(PT p1,PT p2);
PT resetendpt(PT p1,PT p2);
void main()
{
int gd=DETECT,v,gm;
PT p1,p2,p3,p4,ptemp;
printf("\nEnter x1 and y1\n");
scanf("%d %d",&p1.x,&p1.y);
printf("\nEnter x2 and y2\n");
scanf("%d %d",&p2.x,&p2.y);
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
drawwindow();
delay(500);
drawline(p1,p2);
```

```
delay(500);
cleardevice();
delay(500);
p1=setcode(p1);
p2=setcode(p2);
v=visibility(p1,p2);
delay(500);
switch(v)
{
case 0: drawwindow();
delay(500);
drawline(p1,p2);
break;
case 1: drawwindow();
delay(500);
break;
case 2: p3=resetendpt(p1,p2);
p4=resetendpt(p2,p1);
drawwindow();
delay(500);
drawline(p3,p4);
break;
}
delay(5000);
closegraph();
}

void drawwindow()
{
line(150,100,450,100);
```

```
line(450,100,450,350);
line(450,350,150,350);
line(150,350,150,100);
}
void drawline(PT p1,PT p2)
{
line(p1.x,p1.y,p2.x,p2.y);
}
PT setcode(PT p) //for setting the 4 bit code
{
PT ptemp;
if(p.y<100)
ptemp.code[0]='1'; //Top
else
ptemp.code[0]='0';
if(p.y>350)
ptemp.code[1]='1'; //Bottom
else
ptemp.code[1]='0';
if(p.x>450)
ptemp.code[2]='1'; //Right
else
ptemp.code[2]='0';
if(p.x<150)
ptemp.code[3]='1'; //Left
else
ptemp.code[3]='0';
ptemp.x=p.x;
ptemp.y=p.y;
return(ptemp);
```

```
}
int visibility(PT p1,PT p2)
{
int i,flag=0;
for(i=0;i<4;i++)
{
if((p1.code[i]!='0') || (p2.code[i]!='0'))
flag=1;
}
if(flag==0)
return(0);
for(i=0;i<4;i++)
{
if((p1.code[i]==p2.code[i]) && (p1.code[i]=='1'))
flag='0';
}
if(flag==0)
return(1);
return(2);
}
PT resetendpt(PT p1,PT p2)
{
PT temp;
int x,y,i;
float m,k;
if(p1.code[3]=='1')
x=150;
if(p1.code[2]=='1')
x=450;
if((p1.code[3]=='1') || (p1.code[2]=='1'))
```

```
{
m=(float)(p2.y-p1.y)/(p2.x-p1.x);
k=(p1.y+(m*(x-p1.x)));
temp.y=k;
temp.x=x;
for(i=0;i<4;i++)
temp.code[i]=p1.code[i];
if(temp.y<=350 && temp.y>=100)
return (temp);
}
if(p1.code[0]=='1')
y=100;
if(p1.code[1]=='1')
y=350;
if((p1.code[0]=='1') || (p1.code[1]=='1'))
{
m=(float)(p2.y-p1.y)/(p2.x-p1.x);
k=(float)p1.x+(float)(y-p1.y)/m;
temp.x=k;
temp.y=y;
for(i=0;i<4;i++)
temp.code[i]=p1.code[i];
return(temp);
}
else
return(p1);
}
```