slip 1
a) write a program to implement for smiling face animations using graphics function

```
#include<graphics.h>
#include<std.io>
#include<conio.h>
#include<dos.h>
void main()
{
int gd=DETECT,gm,i;
initgraph(&gd,&gm,"c:\\TurboC3\\BGI");

for(i=1;i<=10;i++)
{
cleardevice();

cirlce(200,200,30);//head
circle(190,190,5); //left eye
arc(190,192,50,130,10);
circle(210,190,5);// right eye
arc(210,192,50,130,10);
//arc(190,192,50,130,10);
//for smiling lips

if(i%2==0)
{
arc(200,210,180,360,10);
line(187,210,193,212);
line(207,210,213,212);
}
// not smiling

else
{
line (193,205,193,215);
line(193,210,207,210);
line(207,205,207,215);
}
delay(500);
}
getch();
closegraph();
}
```

b)write a program to draw co-ordinates axis at the center of the screen

slip 2

a) develop the program for DDA line drawing algorithm for pixel positions(1,1)(20,20) //C

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

```
#include<math.h>
#include<dos.h>
#include<conio.h>

void main()
{

float x,y,x1,y1,x2,y2,dx,dy,step;
int i,gd=DETECT,gm;
printf("Enter the value of x1:");
scanf("%f",&x1);

printf("Enter the value of y1:");
scanf("%f",&y1);

printf("Enter the value of x2:");
scanf("%f",&x2);

printf("Enter the value of y2:");
scanf("%f",&y2);

initgraph(&gd,&gm,"C:\\TurboC3\\BGI");

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,1);
x=x+dx;
y=y+dy;
i=i+1;
// sleep(1);
}
getch();
closegraph();
}
```

b)write a program to rotate a circle about an axis

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

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

int xc=50,yc=200,r=35;
int x[15],y[15];
void drawcircles()
{
setcolor(YELLOW);
circle(xc,yc,r);
circle(xc,yc,r+5);
}
void main()
{
double angle=0,theta;
int i,a;
int gd=DETECT,gm;
initgraph(&gd,&gm,"..\\bgi");
a=xc+r;
while(!kbhit())
{
while(a<=630)
{
theta=M_PI*angle/180;
cleardevice();
drawcircles();
for(i=0;i<18;i++)
{
theta=M_PI*angle/180;
x[i]=xc+r*cos(theta);
y[i]=yc+r*sin(theta);
angle+=20;
line(xc,yc,x[i],y[i]);
}
angle+=2; xc+=2; a=xc+r;
delay(50);
}
xc=50; r=35; a=xc+r;
}
getch();
closegraph();
}
```

SLIP 3

a) write c++ program to implements the concept of boundary fill algorithm

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

void bfill(int x,int y,int f_col,int b_col)
```

```
{
int current = getpixel(x,y);
if(current!=f_col&&current!=b_col) //f_col is fillcolor //b_col is bordercolor
{
delay(1);
putpixel(x,y,f_col);
bfill(x+1,y,f_col,b_col);
bfill(x-1,y,f_col,b_col);
bfill(x,y+1,f_col,b_col);
bfill(x,y-1,f_col,b_col);
}
}
void main(){
int xc,yc,r;
int gdriver = DETECT,gm;
initgraph(&gd,&gm,"C:\TC\BGI");
cout<<"Enter co-ordinates of the centre: ";
cin>>xc>>yc;
cout<<"Enter radius of circle: ";
cin>>r;
circle(xc,yc,r);
cout<<"Press any key to fill circle…";
getch();
bfill(xc,yc,RED,WHITE); //bfill is boundaryfill
getch();
closegraph();
}
```

b) write c++ program to show translation of an object

slip 4

a) divide your screen in four region draw circle, rectangle,ellipse,and half ellipse in each region
with appropriate message

b) write a program to fill a circle using flood fill algorithm

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

void floodfill(int x, int y,int oldcolor,int newcolor)
{
if(getpixel(x,y)==oldcolor);
{
delay(10);
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);
}
}

void main()
{
int gd=DETECT,gm,r;
int x,y
printf("Enter the x and y co-ordinates of the center of the circle:");
scanf("%d%d",&x&y);
printf("Enter the radius of circle:");
scanf("%d",&r);
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
circle(x,y,r);
floddfill(x,y,0,9);
getch();
closegraph();
}
```

slip 5

a) write c++ program to perform 2D rotation

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
#include<math.h>>
void main()
float x1,y1,x2,y2,x,y,x3,y3,x4,y4,a;
int ch;
int main(void)
{
 int gd= DETECT, gm;
 clrscr();
 initgraph(&gd,&gm,"c:\\tc\\bgi");
 cout<<"enter coordinates of line1:\n";
 cin>>x1>>y1>>x2>>y2;
 cout<<"enter coordinates for relative line:\n";
 cin>>x3>>y3;
 cout<<"enter the angle of rotation:\n";
 cin>>a;
 cleardevice();
 line(x1,y1,x2,y2);
 line(x2,y2,x3,y3);
 line(x1,y1,x3,y3);
 a=a*(3.14/180);
 x1=(x1*cos(a))-(y1*sin(a));
 y1=(x1*sin(a))+(y1*cos(a));
 x2=(x2*cos(a))-(y2*sin(a));
 y2=(x2*sin(a))+(y2*cos(a));
 x3=(x3*cos(a))-(y3*sin(a));
 y3=(x3*sin(a))+(y3*cos(a));
 cout<<"now hit a key to see rotation:";
 getch();
 line(x1,y1,x2,y2);
 line(x2,y2,x3,y3);
 line(x1,y1,x3,y3);
 getch();
 closegraph();
}
```

b)draw the following basic shape for in the center of screen
1.circle 2.Rectangle 3.Square 4.Ellipse 5.Line

```
//Circle

#include<stdio.h>
#include<graphics.h>
#include<conio.h>
void main()
{
int main(){
   int gd = DETECT,gm;
   int x ,y ,radius=80;
   initgraph(&gd, &gm, "C:\\TC\\BGI");
   x = getmaxx()/2;
   y = getmaxy()/2;
   outtextxy(x-100, 50, "CIRCLE Using Graphics in C");
   circle(x, y, radius);
   getch();
   closegraph();
   return 0;
}

//Rectangle

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

int main(){
   int gd = DETECT,gm;
   initgraph(&gd, &gm, "C:\\TC\\BGI");
   rectangle(150, 50, 400, 150);
   getch();
   closegraph();
   return 0;
}
//Square
```

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

{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
rectangle(400,350,250,200);
getch();
closegraph();
return 0;
}

//Ellipse

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

int main(){
  int gd = DETECT,gm;
  int x ,y;
  initgraph(&gd, &gm, "C:\\TC\\BGI");
  x = getmaxx()/2;
  y = getmaxy()/2;

  outtextxy(x-100, 50, "ELLIPSE Using Graphics in C");
  /* Draw ellipse on screen */
  ellipse(x, y, 0, 360, 120, 60);

  getch();
  closegraph();
  return 0;
}

//Line

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

int main()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
line(100,100,200,200);
getch();
closegraph();
return 0;
}
```

slip 6

a) write c++program to perform 2D translations

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

void main()
{
int gd=DETECT,gm;
int x1,y1,x2,y2,tx,ty;
initgraph(&gd,&gm,"C:\\TUrboC3\\BGI");
printf("Enter Endpoint x1:");
scanf("%d",&x1);

printf("Enter Endpoint x2:");
scanf("%d",&x2);

printf("Enter Endpoint y1:");
scanf("%d",&y1);

printf("Enter Endpoint y2:");
scanf("%d",&y2);

line(x1,y1,x2,y2);
sleep(1);
printf("Enter Translation coordinates tx:");
scanf("%d",&tx)

printf("Enter Translation coordinates ty:");
scanf("%d",&ty)

x1=x1+tx;
y1=y1+ty;
x2=x2+tx;
y2=y2+ty;

printf("The new Line After Translation:")

line(x1,y1,x2,y2);
getch();
closegraph()'
}
```

b)Develop the program for bresenham line drawing algorithm

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

```
void main()
{
int dx,dy,x,y,p,x1,x2,y1,y2;
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TUrboC3\\BGI");
printf("Enter the x-coordinates of first point:x1:");
scanf("%d",&x1);

printf("Enter the x-coordinates of first point:x2:");
scanf("%d",&x2);

printf("Enter the x-coordinates of second point:y1:");
scanf("%d",&y1);

printf("Enter the x-coordinates of second point:y2:");
scanf("%d",&y2);

dx=abs(x2-x1);
dy=abs(y2-y1);
p=2*(dy-dx);
x=x1;
y=y1;
initgraph(&gd,gm,"C:\\TUrboC3\\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();

}
```

slip 7

a)write a c++ program for 2D rotation on given object //Triangle

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

```
#include<stdio.h>
#include<math.h>>
void main()
float x1,y1,x2,y2,x,y,x3,y3,x4,y4,a;
int ch;
int main(void)
{
 int gd= DETECT, gm;
 clrscr();
 initgraph(&gd,&gm,"c:\\tc\\bgi");
 cout<<"enter coordinates of line1:\n";
 cin>>x1>>y1>>x2>>y2;
 cout<<"enter coordinates for relative line:\n";
 cin>>x3>>y3;
 cout<<"enter the angle of rotation:\n";
 cin>>a;
 cleardevice();
 line(x1,y1,x2,y2);
 line(x2,y2,x3,y3);
 line(x1,y1,x3,y3);
 a=a*(3.14/180);
 x1=(x1*cos(a))-(y1*sin(a));
 y1=(x1*sin(a))+(y1*cos(a));
 x2=(x2*cos(a))-(y2*sin(a));
 y2=(x2*sin(a))+(y2*cos(a));
 x3=(x3*cos(a))-(y3*sin(a));
 y3=(x3*sin(a))+(y3*cos(a));
 cout<<"now hit a key to see rotation:";
 getch();
 line(x1,y1,x2,y2);
 line(x2,y2,x3,y3);
 line(x1,y1,x3,y3);
 getch();
 closegraph();
}
```

b)write c++ program to implement boundary fill algorithm

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

void bfill(int x,int y,int f_col,int b_col)
{
int current = getpixel(x,y);
if(current!=f_col&&current!=b_col) //f_col is fillcolor //b_col is bordercolor
{
delay(1);
putpixel(x,y,f_col);
bfill(x+1,y,f_col,b_col);
bfill(x-1,y,f_col,b_col);
bfill(x,y+1,f_col,b_col);
bfill(x,y-1,f_col,b_col);
}
}
void main(){
int xc,yc,r;
int gdriver = DETECT,gm;
initgraph(&gd,&gm,"C:\TC\BGI");
cout<<"Enter co-ordinates of the centre: ";
cin>>xc>>yc;
cout<<"Enter radius of circle: ";
cin>>r;
circle(xc,yc,r);
cout<<"Press any key to fill circle…";
getch();
bfill(xc,yc,RED,WHITE); //bfill is boundaryfill
getch();
closegraph();
}
```

slip 8

a) write a program to implement cohen sutherland line clillping

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

struct point
{
int x,y;
char code[4];
};
void drawwindow();
void drawline(point p1,point p2);
point setcode(point p);
int visibility(point p1,point p2);
point resetendpt(point p1,point p2);
void main()
{
int gd=DETECT,v,gm;
point p1,p2,p3,p4,ptemp;

printf("\n Enter x1 and y1\n");
scanf("%d%d",&p1.x,&p1.y);
printf("\n Enter 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);
getch()'
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: p2=resetendpt(p1,p2);
 p4=resetendpt(p2,p1);
 drawwindow();
 delay(500);
 drawline(p3,p4);
 break;
}

//delay(500);
getch();
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(point p1,point p2)
{
line(p1.x,p1.y,p2.x,p2.y);
}

point setcode(point p) //for setting 4 bit code

{
```

```
point 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(point p1,point p2)

{
int i,flag=0;

for(i=o;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);
}

point resetendpt(point p1,point p2)
{
point temp;
intx,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=o;i<4;i++)
temp.code[1]=p1.code[i];

return(temp);
}
else
return(p1);
}
}
```

b) write c++ program to implement polynomial polygon

slip 9

a)develop the program for DDA line drawing algorithm for pixel positions(0,0)(20,20) //C

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

void main()
{

float x,y,x1,y1,x2,y2,dx,dy,step;
int i,gd=DETECT,gm;
printf("Enter the value of x1:");
scanf("%f",&x1);

printf("Enter the value of y1:");
scanf("%f",&y1);

printf("Enter the value of x2:");
scanf("%f",&x2);

printf("Enter the value of y2:");
scanf("%f",&y2);

initgraph(&gd,&gm,"C:\\TurboC3\\BGI");

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,1);
x=x+dx;
y=y+dy;
i=i+1;
// sleep(1);
}
getch();
closegraph();
}
```

b)divide your screen in four region draw circle, rectangle,ellipse,and half ellipse in each region
with appropriate message

slip 10

a)write a program to draw a simple hut on the screen

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

int main(){
 int gd = DETECT,gm;
    initgraph(&gd, &gm, "X:\\TC\\BGI");
    /* Draw Hut */
    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);

    /* Fill colours */
    setfillstyle(SOLID_FILL, BROWN);
    floodfill(152, 182, WHITE);
    floodfill(252, 182, WHITE);
    setfillstyle(SLASH_FILL, BLUE);
    floodfill(182, 252, WHITE);
    setfillstyle(HATCH_FILL, GREEN);
    floodfill(200, 105, WHITE);
    floodfill(210, 105, WHITE);

    getch();
    closegraph();
    return 0;
}
```

b)Develop the program for bresenham line drawing algorithm

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
#include<math.h>
void main()
{
int dx,dy,x,y,p,x1,x2,y1,y2;
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TUrboC3\\BGI");
printf("Enter the x-coordinates of first point:x1:");
scanf("%d",&x1);

printf("Enter the x-coordinates of first point:x2:");
scanf("%d",&x2);

printf("Enter the x-coordinates of second point:y1:");
scanf("%d",&y1);

printf("Enter the x-coordinates of second point:y2:");
scanf("%d",&y2);

dx=abs(x2-x1);
dy=abs(y2-y1);
p=2*(dy-dx);
x=x1;
y=y1;
initgraph(&gd,gm,"C:\\TUrboC3\\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();

}
```

slip 11

a) write a program to implement cohen sutherland line clillping

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

struct point
{
int x,y;
char code[4];
};
void drawwindow();
void drawline(point p1,point p2);
point setcode(point p);
int visibility(point p1,point p2);
point resetendpt(point p1,point p2);
void main()
{
int gd=DETECT,v,gm;
point p1,p2,p3,p4,ptemp;

printf("\n Enter x1 and y1\n");
scanf("%d%d",&p1.x,&p1.y);
printf("\n Enter 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);
getch()'
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: p2=resetendpt(p1,p2);
 p4=resetendpt(p2,p1);
 drawwindow();
 delay(500);
 drawline(p3,p4);
 break;
}

//delay(500);
getch();
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(point p1,point p2)
{
line(p1.x,p1.y,p2.x,p2.y);
}

point setcode(point p) //for setting 4 bit code

{
point 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(point p1,point p2)

{
int i,flag=0;

for(i=o;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);
}
```

```
point resetendpt(point p1,point p2)
{
point temp;
intx,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=o;i<4;i++)
temp.code[1]=p1.code[i];

return(temp);
}
else
return(p1);
}
}
```

b)write a c++ program to draw concetric circle & fill it with different color

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

```
int gd=DETECT,gm;
int x,y,r=100,r1=80,r2=60,r3=40;
initgraph(&gd,&gm,"C:TurboC3\\BGI");
x=getmaxx()12;
y=getmaxy()12;
setcolor(4)
cirlce(x,y,r);

setcolor(8)
cirlce(x,y,r1);

setcolor(3)
cirlce(x,y,r2);

setcolor(7)
cirlce(x,y,r3);

getch();
closegraph();
return 0;
}
```

slip 12

a)write a program to implement liang barsky line clipping

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

void main()
{

int i,gd=DETECT,gm;
int x1,y1,x2,y2,xmin,xmax,ymin,ymax,xx1,xx2,yy1,yy2,dx,dy;
float t1,t2,p[4],q[4],temp;

x1=120;
y1=120;
x2=300;
y2=300;

xmin=100;
ymin=100;
xmax=250;
ymax=250;

initgraph(&gd,&gm,"C:\\turboC3\\BGI");
rectangle(xmin.ymin,xmax,ymax);
dx=x2-x1;
dy=y2-y1;
```

```
p[0]=-dx;
p[1]=dx;
p[2]=-dy;
p[3]=dy;

q[0]=x1-xmin;
q[1]=xmax-x1;
q[2]=y1-ymin;
q[3]=ymax-y1;

for(i=0;i<4;i++)
{
if(p[i]==0)
{
printf("Line is parallel to one of the clipping boundary");
if(q[i]>=0)
{
if(i<2)
{
if(y1<ymin)
{
y1=ymin;
}
if(y2>ymax)
{
y2=ymax;
}
line(x1,y1,x2,y2);
}
if(i>1)
{
if(x1<xmin)
{
x1=min;
}
if(x2>xmax)
{
x2=xmax;
}
line(x1,y1,x2,y2);
}
}
}
}
t1=0;
t2=1;
for(i=o;i<4;i++)
{
temp=q[i]/p[i];
if(p[i]<0)
{
```

```
if(t1<=temp)
t1=temp;
}
else
{
if(t2>temp)
t2=temp;
}
}
if(t1<t2)
{
xx1=x1+t1*p[1];
xx2=x1+t2*p[1];
yy1=y1+t1*p[3];
yy2=y1+t2*p[3];
line(xx1,yy1,xx2,yy2);
}
getch();
closegraph();
}
```

b)write a program to draw a circle and line on a screen

slip 13

a)write c++ program to implement the boundary fill algorithm

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

void bfill(int x,int y,int f_col,int b_col)
{
int current = getpixel(x,y);
if(current!=f_col&&current!=b_col) //f_col is fillcolor //b_col is bordercolor
{
delay(1);
putpixel(x,y,f_col);
bfill(x+1,y,f_col,b_col);
bfill(x-1,y,f_col,b_col);
bfill(x,y+1,f_col,b_col);
bfill(x,y-1,f_col,b_col);
}
}
void main(){
int xc,yc,r;
int gdriver = DETECT,gm;
initgraph(&gd,&gm,"C:\TC\BGI");
cout<<"Enter co-ordinates of the centre: ";
cin>>xc>>yc;
cout<<"Enter radius of circle: ";
cin>>r;
circle(xc,yc,r);
cout<<"Press any key to fill circle…";
getch();
bfill(xc,yc,RED,WHITE); //bfill is boundaryfill
getch();
closegraph();
}
```

b)write c++ program to implement midpoint circle drawing algorithm

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

void main()
{
int xc,yc,r,pk,x,y;
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TUrboC3\\BGI");
printf("Enter the x-coordinates of the center:xc");
scanf("%d",&xc);
printf("Enter the y-coordinates of the center:yc:");
scanf("%d",&yc);
printf("Enter the radius:");
scanf("%d",&r);

x=0;
y=r;
pk=1-r;

while(x<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);
if(pk<0)
{
x=x+1;
pk=pk+(2*x)+3;
}
else
{
x=x+1;
y=y-1;
pk=pk+(2*x)-(2*y)+5;
}
}
getch();
closegraph();
}
```

slip 14

a)write c++ program to implement 2D scaling

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

void main()
{
int gd=DETECT,gm;
int x1,y1,x2,y2,sx,sy;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
printf("Enter Endpoint x1:");
scanf("%d"&x1);
printf("Enter Endpoint x2:");
scanf("%d"&x2);
printf("Enter Endpoint y1:");
scanf("%d"&y1);
printf("Enter Endpoint y2:");
scanf("%d"&y2);

//setcolor(WHITE);
line(x1,y1,x2,y2);
sleep(1);
printf("Enter the scaling coordinates sx:");
scanf("%d",&sx);

printf("Enter the scaling coordinates sy:");
scanf("%d",&sy);

x1=x1*sx;
y1=y1*sy;
x2=x2*sx;
y2=y2*sy;

printf("The new line after scaling:::");
setcolor(WHITE);

line(x1,y1,x2,y2);
getch();
closegraph();
}
```

b) write c++ program to implement flood fill algorithm

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

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

void floodfill(int x, int y,int oldcolor,int newcolor)
{
if(getpixel(x,y)==oldcolor);
{
delay(10);
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);
}
}

void main()
{
int gd=DETECT,gm,r;
int x,y
printf("Enter the x and y co-ordinates of the center of the circle:");
scanf("%d%d",&x&y);
printf("Enter the radius of circle:");
scanf("%d",&r);
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
circle(x,y,r);
floddfill(x,y,0,9);
getch();
closegraph();
}
```

slip 15

a)write c++ program to implement the 2d rotation of an object

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
#include<math.h>>
void main()
float x1,y1,x2,y2,x,y,x3,y3,x4,y4,a;
int ch;
int main(void)
{
 int gd= DETECT, gm;
 clrscr();
 initgraph(&gd,&gm,"c:\\tc\\bgi");
 cout<<"enter coordinates of line1:\n";
 cin>>x1>>y1>>x2>>y2;
 cout<<"enter coordinates for relative line:\n";
 cin>>x3>>y3;
 cout<<"enter the angle of rotation:\n";
 cin>>a;
 cleardevice();
 line(x1,y1,x2,y2);
 line(x2,y2,x3,y3);
 line(x1,y1,x3,y3);
 a=a*(3.14/180);
 x1=(x1*cos(a))-(y1*sin(a));
 y1=(x1*sin(a))+(y1*cos(a));
 x2=(x2*cos(a))-(y2*sin(a));
 y2=(x2*sin(a))+(y2*cos(a));
 x3=(x3*cos(a))-(y3*sin(a));
 y3=(x3*sin(a))+(y3*cos(a));
 cout<<"now hit a key to see rotation:";
 getch();
 line(x1,y1,x2,y2);
 line(x2,y2,x3,y3);
 line(x1,y1,x3,y3);
 getch();
 closegraph();
}
```

b) write a c++ program for bouncing ball

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

int main() {
 int gd = DETECT, gm;
 int i, x, y, flag=0;
 initgraph(&gd, &gm, "C:\\TC\\BGI");

 /* get mid positions in x and y-axis */
 x = getmaxx()/2;
 y = 30;


 while (!kbhit()) {
  if(y >= getmaxy()-30 || y <= 30)
    flag = !flag;
    /* draws the gray board */
    setcolor(RED);
    setfillstyle(SOLID_FILL, RED);
    circle(x, y, 30);
    floodfill(x, y, RED);

 /* delay for 50 milli seconds */
 delay(50);

 /* clears screen */
 cleardevice();
 if(flag){
     y = y + 5;
 } else {
     y = y - 5;
 }
    }
 getch();
    closegraph();
    return 0;
}
```

slip 16

a)draw the following basic shape for in the center of screen
1.circle 2.Rectangle 3.Square 4.Ellipse 5.Line

```
//Circle

#include<stdio.h>
#include<graphics.h>
#include<conio.h>
void main()
{
int main(){
   int gd = DETECT,gm;
   int x ,y ,radius=80;
   initgraph(&gd, &gm, "C:\\TC\\BGI");
   x = getmaxx()/2;
   y = getmaxy()/2;
   outtextxy(x-100, 50, "CIRCLE Using Graphics in C");
   circle(x, y, radius);
   getch();
   closegraph();
   return 0;
}

//Rectangle

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

int main(){
   int gd = DETECT,gm;
   initgraph(&gd, &gm, "C:\\TC\\BGI");
   rectangle(150, 50, 400, 150);
   getch();
   closegraph();
   return 0;
}
//Square
```

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

{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
rectangle(400,350,250,200);
getch();
closegraph();
return 0;
}

//Ellipse

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

int main(){
  int gd = DETECT,gm;
  int x ,y;
  initgraph(&gd, &gm, "C:\\TC\\BGI");
  x = getmaxx()/2;
  y = getmaxy()/2;

  outtextxy(x-100, 50, "ELLIPSE Using Graphics in C");
  /* Draw ellipse on screen */
  ellipse(x, y, 0, 360, 120, 60);

  getch();
  closegraph();
  return 0;
}

//Line

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

int main()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
line(100,100,200,200);
getch();
closegraph();
return 0;
}
```

b) Write c++ program for drawing line using DDA line drawing alogrithm

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

void main()
{

float x,y,x1,y1,x2,y2,dx,dy,step;
int i,gd=DETECT,gm;
printf("Enter the value of x1:");
scanf("%f",&x1);

printf("Enter the value of y1:");
scanf("%f",&y1);

printf("Enter the value of x2:");
scanf("%f",&x2);

printf("Enter the value of y2:");
scanf("%f",&y2);

initgraph(&gd,&gm,"C:\\TurboC3\\BGI");

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,1);
x=x+dx;
y=y+dy;
i=i+1;
// sleep(1);
}
getch();
closegraph();
}
```

slip 17

a) develop the program for the mid-point circle drawing algorithm

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

void main()
{
int xc,yc,r,pk,x,y;
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TUrboC3\\BGI");
printf("Enter the x-coordinates of the center:xc");
scanf("%d",&xc);
printf("Enter the y-coordinates of the center:yc:");
scanf("%d",&yc);
printf("Enter the radius:");
scanf("%d",&r);

x=0;
y=r;
pk=1-r;

while(x<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);
if(pk<0)
{
x=x+1;
pk=pk+(2*x)+3;
}
else
{
x=x+1;
y=y-1;
pk=pk+(2*x)-(2*y)+5;
}
}
getch();
closegraph();
}
```

b) write c/c++/python program for moving car on the screen

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

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

int main() {
   int gd = DETECT, gm;
   int i, maxx, midy;

   /* initialize graphic mode */
   initgraph(&gd, &gm, "X:\\TC\\BGI");
   /* maximum pixel in horizontal axis */
   maxx = getmaxx();
   /* mid pixel in vertical axis */
   midy = getmaxy()/2;

   for (i=0; i < maxx-150; i=i+5) {
      /* clears screen */
      cleardevice();

      /* draw a white road */
      setcolor(WHITE);
      line(0, midy + 37, maxx, midy + 37);

      /* Draw Car */
      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);
      /*  Draw Wheels */
      circle(30 + i, midy + 25, 9);
      circle(90 + i, midy + 25, 9);
```

```
        floodfill(30 + i, midy + 25, BLUE);
        floodfill(90 + i, midy + 25, BLUE);
        /* Add delay of 0.1 milli seconds */
        delay(100);
    }

    getch();
    closegraph();
    return 0;
}
```

slip 18

a) implement basic function used for graphics in c/c++/python language give an example for each of them

b) write a simple program to develop text screen saver using graphics functions.

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
void main ()
{
int gd=DETECT,gm,x,i;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
for(x=0;x<500;x++)
{
cleardevice();
settextstyle(1,0,5);
setcolor(RED);
outtextxy(50,415-x,"Welcome");
setcolor(GREEN);
outtextxy(250,415,-x,"to");
setcolor(YELLOW);
settextstyle(3,0,5);
outtextxy(350,415-x,"Graphics");
}
getch();
closegraph();
}
```

slip 19

a)develop the program for the mid-point ellipse alogrithm

```
#include<graphics.h>
//#include<stdlib.h0>
#include<iostream.h>
#include<conio.h>
void main()
{
 clrscr();
int gd=DETECT,gm;
int xc,yc,x,y;float p;
long rx,ry;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
cout<<"Enter the coordinates of center:";
cin>>xc>>yc;
cout<<"Enter x,y radius of ellipse:";
cin>>rx>>ry;

//Region 1
p=ry*ry-rx*rx*ry+rx*rx/4;
x=0;y=ry;
while(2.0*ry*ry*x<=2.0*rx*rx*y)
{
if(P<0)
{
x++;
p=p+2*ry*ry*x+ry*ry;
}
else
x++;y--;
p=p+2*ry*ry*x-2*rx*rx*y+ry*ry;
}
putpixel(xc+x,yc+y,RED);
putpixel(xc+x,yc-y,RED);
putpixel(xc-x,yc+y,RED);
putpixel(xc-x,yc-y,RED);
}

//Region 2
p=ry*ry*(x+0.5)*(x+0.5)+rx*rx*(y-1)*(y-1)-rx*rx*ry*ry;
while(y>0)
{
if(p<=0)
{
x++;y--;
p=p+2*ry*ry*x-2*rx*rx*y+rx*rx;
}
else
{
y--;
p=p-2*rx*rx*y+rx*rx;
}
putpixel(xc+x,yc+y,RED);
putpixel(xc+x,yc-y,RED);
putpixel(xc-x,yc+y,RED);
putpixel(xc-x,yc-y,RED);
}
getch();
closegraph();
}
```

b) write a program to implement 2D scaling

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

void main()
{
int gd=DETECT,gm;
int x1,y1,x2,y2,sx,sy;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
printf("Enter Endpoint x1:");
scanf("%d"&x1);
printf("Enter Endpoint x2:");
scanf("%d"&x2);
printf("Enter Endpoint y1:");
scanf("%d"&y1);
printf("Enter Endpoint y2:");
scanf("%d"&y2);

//setcolor(WHITE);
line(x1,y1,x2,y2);
sleep(1);
printf("Enter the scaling coordinates sx:");
scanf("%d",&sx);

printf("Enter the scaling coordinates sy:");
scanf("%d",&sy);

x1=x1*sx;
y1=y1*sy;
x2=x2*sx;
y2=y2*sy;

printf("The new line after scaling:::");
setcolor(WHITE);

line(x1,y1,x2,y2);
getch();
closegraph();
}
```

slip 20

a) program to create a house like figure and perform the followung operation
1.Scaling about the orgin followed by transaltion
2.scaling with reference to an arbitrary point
3.reflect about the line y=mx+c

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

```
void reset (int h[][2])
{
   int val[9][2] = {
  { 50, 50 },{ 75, 50 },{ 75, 75 },{ 100, 75 },
  { 100, 50 },{ 125, 50 },{ 125, 100 },{ 87, 125 },{ 50, 100 }
     };
   int i;
   for (i=0; i<9; i++)
   {
 h[i][0] = val[i][0]-50;
 h[i][1] = val[i][1]-50;
   }
}
void draw (int h[][2])
{
   int i;
   setlinestyle (DOTTED_LINE, 0, 1);
   line (320, 0, 320, 480);
   line (0, 240, 640, 240);
   setlinestyle (SOLID_LINE, 0, 1);
   for (i=0; i<8; i++)
 line (320+h[i][0], 240-h[i][1], 320+h[i+1][0], 240-h[i+1][1]);
   line (320+h[0][0], 240-h[0][1], 320+h[8][0], 240-h[8][1]);
}
void rotate (int h[][2], float angle)
{
   int i;
   for (i=0; i<9; i++)
   {
 int xnew, ynew;
 xnew = h[i][0] * cos (angle) - h[i][1] * sin (angle);
 ynew = h[i][0] * sin (angle) + h[i][1] * cos (angle);
 h[i][0] = xnew; h[i][1] = ynew;
   }
}
void scale (int h[][2], int sx, int sy)
{
   int i;
   for (i=0; i<9; i++)
   {
 h[i][0] *= sx;
 h[i][1] *= sy;
   }
}
void translate (int h[][2], int dx, int dy)
{
   int i;
   for (i=0; i<9; i++)
   {
 h[i][0] += dx;
 h[i][1] += dy;
```

```
    }
}
void reflect (int h[][2], int m, int c)
{
 int i;
 float angle;
 for (i=0; i<9; i++)
  h[i][1] -= c;
 angle = M_PI/2 - atan (m);
 rotate (h, angle);
 for (i=0; i<9; i++)
  h[i][0] = -h[i][0];
 angle = -angle;
 rotate (h, angle);
 for (i=0; i<9; i++)
  h[i][1] += c;
}

void main()
{
 int gd=DETECT,gm;
 initgraph(&gd,&gm,"..\\bgi");
 int h[9][2],sx,sy,x,y,m,c,choice;
 do
 {
  clrscr();
  printf("1. Scaling about the origin.\n");
  printf("2. Scaling about an arbitrary point.\n");
  printf("3. Reflection about the line y = mx + c.\n");
  printf("4. Exit\n");
  printf("Enter the choice: ");
  scanf("%d",&choice);
  switch(choice)
  {
   case 1: printf ("Enter the x- and y-scaling factors: ");
    scanf ("%d%d", &sx, &sy);
    reset (h);
    draw (h);
    getch();
    scale (h, sx, sy);
    cleardevice();
    draw (h);
    getch();
    break;

   case 2: printf ("Enter the x- and y-scaling factors: ");
    scanf ("%d%d", &sx, &sy);
    printf ("Enter the x- and y-coordinates of the point: ");
    scanf ("%d%d", &x, &y);
    reset (h);
    translate (h, x, y);// Go to arbitrary point
    draw(h); /Show its arbitrary position
    getch();
    cleardevice();
    translate(h,-x,-y);//Take it back to origin
    draw(h);
    getch();
    cleardevice();
    scale (h, sx, sy);//Now Scale it
    draw(h);
    getch();
    translate (h, x, y);//Back to Arbitrary point
    cleardevice();
    draw (h);
    putpixel (320+x, 240-y, WHITE);
    break;

   case 3: printf ("Enter the values of m and c: ");
    scanf ("%d%d", &m, &c);
    reset (h);
    draw (h);
    getch();
    reflect (h, m, c);
    cleardevice();
    draw (h);
    break;

   case 4: exit(0);
  }
 }while(choice!=4);
}
```

slip 21

a)write a program to implement 2D translation

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

void main()
{
int gd=DETECT,gm;
int x1,y1,x2,y2,tx,ty;
initgraph(&gd,&gm,"C:\\TUrboC3\\BGI");
printf("Enter Endpoint x1:");
scanf("%d",&x1);

printf("Enter Endpoint x2:");
scanf("%d",&x2);

printf("Enter Endpoint y1:");
scanf("%d",&y1);
printf("Enter Endpoint y2:");
scanf("%d",&y2);

line(x1,y1,x2,y2);
sleep(1);
printf("Enter Translation coordinates tx:");
scanf("%d",&tx)

printf("Enter Translation coordinates ty:");
scanf("%d",&ty)

x1=x1+tx;
y1=y1+ty;
x2=x2+tx;
y2=y2+ty;

printf("The new Line After Translation:")

line(x1,y1,x2,y2);
getch();
closegraph()'
}
```

b)write a program to draw concentric circle & fill it with different color

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
int main()
{
int gd=DETECT,gm;
int x,y,r=100,r1=80,r2=60,r3=40;
initgraph(&gd,&gm,"C:TurboC3\\BGI");
x=getmaxx()12;
y=getmaxy()12;
setcolor(4)
cirlce(x,y,r);

setcolor(8)
cirlce(x,y,r1);

setcolor(3)
cirlce(x,y,r2);

setcolor(7)
cirlce(x,y,r3);

getch();
closegraph();
return 0;
}
```

slip 22

a) write a c++ program to demostrate 2D transaltion,rotation&scaling using swtich case

```
#include <conio.h>
#include <iostream.h>
#include <graphics.h>
#include <math.h>
void main(){
int x1=200,y1=200,x2=250,y2=250,x3=180,y3=270,option;
int gdriver = DETECT,gmode;
initgraph(&gdriver,&gmode,"C:\TC\BGI");
do{
cleardevice();
gotoxy(1,1);
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x3,y3,x1,y1);
cout<<"\n1.Translation 2.Scaling 3.Rotation 4.Exit\nEnter your choice: ";
cin>>option;
switch(option){
case 1:
float tx,ty;
cout<<"Enter tx & ty: ";
cin>>tx>>ty;
x1+=tx;x2+=tx;x3+=tx;
y1+=ty;y2+=ty;y3+=ty;
break;

case 2:
float sx,sy;
cout<<"Enter sx & sy: ";
cin>>sx>>sy;
x1*=sx;x2*=sx;x3*=sx;
y1*=sy;y2*=sy;y3*=sy;
break;

case 3:
float deg;
cout<<"Enter angle: ";
cin>>deg;
deg = deg*3.14/180;
int x,y;
x=x1;y=y1;
x1 = x*cos(deg)-y*sin(deg);
y1 = x*sin(deg)+y*cos(deg);
x=x2;y=y2;
x2 = x*cos(deg)-y*sin(deg);
y2 = x*sin(deg)+y*cos(deg);
x=x3;y=y3;
x3 = x*cos(deg)-y*sin(deg);
y3 = x*sin(deg)+y*cos(deg);
break;

case 4:
break;

default:
cout<<“Invalid choice”;
}
}
while(option!=4);
closegraph();
}
```

b) write c++ program for implementing polynomial polygon

slip 23

a) write a program to perform smiling face using graphics

```
#include<graphics.h>
#include<std.io>
#include<conio.h>
#include<dos.h>
void main()
{
int gd=DETECT,gm,i;
initgraph(&gd,&gm,"c:\\TurboC3\\BGI");

for(i=1;i<=10;i++)
{
cleardevice();

cirlce(200,200,30);//head
circle(190,190,5); //left eye
arc(190,192,50,130,10);
circle(210,190,5);// right eye
arc(210,192,50,130,10);
//arc(190,192,50,130,10);
//for smiling lips

if(i%2==0)
{
arc(200,210,180,360,10);
line(187,210,193,212);
line(207,210,213,212);
}
// not smiling

else
{
line (193,205,193,215);
line(193,210,207,210);
line(207,205,207,215);
}
delay(500);
}
getch();
closegraph();
}
```

b) write a c++ program to implementation 2d rotation of an object

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
#include<math.h>>
void main()
float x1,y1,x2,y2,x,y,x3,y3,x4,y4,a;
int ch;
int main(void)
{
 int gd= DETECT, gm;
 clrscr();
 initgraph(&gd,&gm,"c:\\tc\\bgi");
 cout<<"enter coordinates of line1:\n";
 cin>>x1>>y1>>x2>>y2;
 cout<<"enter coordinates for relative line:\n";
 cin>>x3>>y3;
 cout<<"enter the angle of rotation:\n";
 cin>>a;
 cleardevice();
 line(x1,y1,x2,y2);
 line(x2,y2,x3,y3);
 line(x1,y1,x3,y3);
 a=a*(3.14/180);
 x1=(x1*cos(a))-(y1*sin(a));
 y1=(x1*sin(a))+(y1*cos(a));
 x2=(x2*cos(a))-(y2*sin(a));
 y2=(x2*sin(a))+(y2*cos(a));
 x3=(x3*cos(a))-(y3*sin(a));
 y3=(x3*sin(a))+(y3*cos(a));
 cout<<"now hit a key to see rotation:";
 getch();
 line(x1,y1,x2,y2);
 line(x2,y2,x3,y3);
 line(x1,y1,x3,y3);
 getch();
 closegraph();
}
```

slip 24

a) write a program to perform smiling face animation using graphics functions

```
#include<graphics.h>
#include<std.io>
#include<conio.h>
#include<dos.h>
void main()
{
int gd=DETECT,gm,i;
initgraph(&gd,&gm,"c:\\TurboC3\\BGI");

for(i=1;i<=10;i++)
{
cleardevice();

cirlce(200,200,30);//head
circle(190,190,5); //left eye
arc(190,192,50,130,10);
circle(210,190,5);// right eye
arc(210,192,50,130,10);
//arc(190,192,50,130,10);
//for smiling lips

if(i%2==0)
{
arc(200,210,180,360,10);
line(187,210,193,212);
line(207,210,213,212);
}
// not smiling

else
{
line (193,205,193,215);
line(193,210,207,210);
line(207,205,207,215);
}
delay(500);
}
getch();
closegraph();
}
```

b) develop the program for bresenham line drawing algorithm

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
#include<math.h>
void main()
{
int dx,dy,x,y,p,x1,x2,y1,y2;
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TUrboC3\\BGI");
printf("Enter the x-coordinates of first point:x1:");
scanf("%d",&x1);

printf("Enter the x-coordinates of first point:x2:");
scanf("%d",&x2);

printf("Enter the x-coordinates of second point:y1:");
scanf("%d",&y1);

printf("Enter the x-coordinates of second point:y2:");
scanf("%d",&y2);

dx=abs(x2-x1);
dy=abs(y2-y1);
p=2*(dy-dx);
x=x1;
y=y1;
initgraph(&gd,gm,"C:\\TUrboC3\\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();

}
```

slip 25

a) write a program to demonstrate 2D translation rotation scaling using switch case

```
#include <conio.h>
#include <iostream.h>
#include <graphics.h>
#include <math.h>
void main(){
```

```
int x1=200,y1=200,x2=250,y2=250,x3=180,y3=270,option;
int gdriver = DETECT,gmode;
initgraph(&gdriver,&gmode,"C:\TC\BGI");
do{
cleardevice();
gotoxy(1,1);
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x3,y3,x1,y1);
cout<<"\n1.Translation 2.Scaling 3.Rotation 4.Exit\nEnter your choice: ";
cin>>option;
switch(option){
case 1:
float tx,ty;
cout<<"Enter tx & ty: ";
cin>>tx>>ty;
x1+=tx;x2+=tx;x3+=tx;
y1+=ty;y2+=ty;y3+=ty;
break;

case 2:
float sx,sy;
cout<<"Enter sx & sy: ";
cin>>sx>>sy;
x1*=sx;x2*=sx;x3*=sx;
y1*=sy;y2*=sy;y3*=sy;
break;

case 3:
float deg;
cout<<"Enter angle: ";
cin>>deg;
deg = deg*3.14/180;
int x,y;
x=x1;y=y1;
x1 = x*cos(deg)-y*sin(deg);
y1 = x*sin(deg)+y*cos(deg);
x=x2;y=y2;
x2 = x*cos(deg)-y*sin(deg);
y2 = x*sin(deg)+y*cos(deg);
x=x3;y=y3;
x3 = x*cos(deg)-y*sin(deg);
y3 = x*sin(deg)+y*cos(deg);
break;

case 4:
break;

default:
cout<<"Invalid choice";
}
}
```

```
while(option!=4);
closegraph();
}
```

b) write a program to draw a simple hut on the screen

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

int main(){
 int gd = DETECT,gm;
   initgraph(&gd, &gm, "X:\\TC\\BGI");
   /* Draw Hut */
   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);

   /* Fill colours */
   setfillstyle(SOLID_FILL, BROWN);
   floodfill(152, 182, WHITE);
   floodfill(252, 182, WHITE);
   setfillstyle(SLASH_FILL, BLUE);
   floodfill(182, 252, WHITE);
   setfillstyle(HATCH_FILL, GREEN);
   floodfill(200, 105, WHITE);
   floodfill(210, 105, WHITE);

   getch();
   closegraph();
   return 0;
}
```

slip 26

a)implement basic function used for graphics in c/c++/python give example for each of them

b)write c++ program to implement boundary fill algorithm

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

void bfill(int x,int y,int f_col,int b_col)
{
int current = getpixel(x,y);
if(current!=f_col&&current!=b_col) //f_col is fillcolor //b_col is bordercolor
{
delay(1);
putpixel(x,y,f_col);
bfill(x+1,y,f_col,b_col);
bfill(x-1,y,f_col,b_col);
bfill(x,y+1,f_col,b_col);
bfill(x,y-1,f_col,b_col);
}
}
void main(){
int xc,yc,r;
int gdriver = DETECT,gm;
initgraph(&gd,&gm,"C:\TC\BGI");
cout<<"Enter co-ordinates of the centre: ";
cin>>xc>>yc;
cout<<"Enter radius of circle: ";
cin>>r;
circle(xc,yc,r);
cout<<"Press any key to fill circle…";
getch();
bfill(xc,yc,RED,WHITE); //bfill is boundaryfill
getch();
closegraph();
}
```

slip 27

a)develop a program for mid point circle drawing alogrithm

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

void main()
{
int xc,yc,r,pk,x,y;
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TUrboC3\\BGI");
printf("Enter the x-coordinates of the center:xc");
scanf("%d",&xc);
printf("Enter the y-coordinates of the center:yc:");
scanf("%d",&yc);
printf("Enter the radius:");
scanf("%d",&r);

x=0;
y=r;
pk=1-r;

while(x<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);
if(pk<0)
{
x=x+1;
pk=pk+(2*x)+3;
}
else
{
x=x+1;
y=y-1;
pk=pk+(2*x)-(2*y)+5;
}
}
getch();
closegraph();
}
```

b)write a program to draw coordinates axis at the center of the screen

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

int main()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
line(300,200,300,390);
line(200,290,390,290);
getch();
closegraph();
return();
}
```

slip 28

a) write a program to perform smiling face animation using graphics functions

```
#include<graphics.h>
#include<std.io>
#include<conio.h>
#include<dos.h>
void main()
{
int gd=DETECT,gm,i;
initgraph(&gd,&gm,"c:\\TurboC3\\BGI");

for(i=1;i<=10;i++)
{
cleardevice();

cirlce(200,200,30);//head
circle(190,190,5); //left eye
arc(190,192,50,130,10);
circle(210,190,5);// right eye
arc(210,192,50,130,10);
//arc(190,192,50,130,10);
//for smiling lips

if(i%2==0)
{
arc(200,210,180,360,10);
line(187,210,193,212);
line(207,210,213,212);
}
// not smiling

else
{
line (193,205,193,215);
line(193,210,207,210);
line(207,205,207,215);
}
delay(500);
}
getch();
closegraph();
}
```

b) write c++ program to implment flood fill algorithm

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

void floodfill(int x, int y,int oldcolor,int newcolor)
{
if(getpixel(x,y)==oldcolor);
{
delay(10);
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);
}
}

void main()
{
int gd=DETECT,gm,r;
int x,y
printf("Enter the x and y co-ordinates of the center of the circle:");
scanf("%d%d",&x&y);
printf("Enter the radius of circle:");
scanf("%d",&r);
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
circle(x,y,r);
floddfill(x,y,0,9);
getch();
closegraph();
}
```

slip 29

a) write a program to draw a simple hut on screen

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

int main(){
 int gd = DETECT,gm;
    initgraph(&gd, &gm, "X:\\TC\\BGI");
    /* Draw Hut */
    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);

    /* Fill colours */
    setfillstyle(SOLID_FILL, BROWN);
    floodfill(152, 182, WHITE);
    floodfill(252, 182, WHITE);
    setfillstyle(SLASH_FILL, BLUE);
    floodfill(182, 252, WHITE);
    setfillstyle(HATCH_FILL, GREEN);
    floodfill(200, 105, WHITE);
    floodfill(210, 105, WHITE);

    getch();
    closegraph();
    return 0;
}
```

b)write a c++ program for boundary fill algorithm

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

void bfill(int x,int y,int f_col,int b_col)
{
int current = getpixel(x,y);
if(current!=f_col&&current!=b_col) //f_col is fillcolor //b_col is bordercolor
{
delay(1);
putpixel(x,y,f_col);
bfill(x+1,y,f_col,b_col);
bfill(x-1,y,f_col,b_col);
bfill(x,y+1,f_col,b_col);
bfill(x,y-1,f_col,b_col);
}
}
void main(){
int xc,yc,r;
int gdriver = DETECT,gm;
initgraph(&gd,&gm,"C:\TC\BGI");
cout<<"Enter co-ordinates of the centre: ";
cin>>xc>>yc;
cout<<"Enter radius of circle: ";
cin>>r;
circle(xc,yc,r);
cout<<"Press any key to fill circle…";
getch();
bfill(xc,yc,RED,WHITE); //bfill is boundaryfill
getch();
closegraph();
}
```

slip 30

a) write a program to implement liang barsky line clipping

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

void main()
{

int i,gd=DETECT,gm;
int x1,y1,x2,y2,xmin,xmax,ymin,ymax,xx1,xx2,yy1,yy2,dx,dy;
float t1,t2,p[4],q[4],temp;
```

```
x1=120;
y1=120;
x2=300;
y2=300;

xmin=100;
ymin=100;
xmax=250;
ymax=250;

initgraph(&gd,&gm,"C:\\turboC3\\BGI");
rectangle(xmin.ymin,xmax,ymax);
dx=x2-x1;
dy=y2-y1;

p[0]=-dx;
p[1]=dx;
p[2]=-dy;
p[3]=dy;

q[0]=x1-xmin;
q[1]=xmax-x1;
q[2]=y1-ymin;
q[3]=ymax-y1;

for(i=0;i<4;i++)
{
if(p[i]==0)
{
printf("Line is parallel to one of the clipping boundary");
if(q[i]>=0)
{
if(i<2)
{
if(y1<ymin)
{
y1=ymin;
}
if(y2>ymax)
{
y2=ymax;
}
line(x1,y1,x2,y2);
}
if(i>1)
{
if(x1<xmin)
{
x1=min;
}
if(x2>xmax)
{
x2=xmax;
}
line(x1,y1,x2,y2);
}
}
}
}
t1=0;
t2=1;
for(i=o;i<4;i++)
{
temp=q[i]/p[i];
if(p[i]<0)
{
if(t1<=temp)
t1=temp;
}
else
{
if(t2>temp)
t2=temp;
}
}
if(t1<t2)
{
xx1=x1+t1*p[1];
xx2=x1+t2*p[1];
yy1=y1+t1*p[3];
yy2=y1+t2*p[3];
line(xx1,yy1,xx2,yy2);
}
getch();
closegraph();
}
```

b) write a small program for square and rectangle on a screen

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

int main(){
  int gd = DETECT,gm;
  initgraph(&gd, &gm, "C:\\TC\\BGI");

  /* Draw rectangle on screen */
  rectangle(150, 50, 400, 150);

  /* Draw square on screen */
  square(150, 200, 400, 350);
```

```
   getch();
   closegraph();
   return 0;
}
```

slip 31

a)write a program for the mid point ellipse drawing algorithm

```
#include<graphics.h>
//#include<stdlib.h0>
#include<iostream.h>
#include<conio.h>
void main()
{
 clrscr();
int gd=DETECT,gm;
int xc,yc,x,y;float p;
long rx,ry;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
cout<<"Enter the coordinates of center:";
cin>>xc>>yc;
cout<<"Enter x,y radius of ellipse:";
cin>>rx>>ry;

//Region 1
p=ry*ry-rx*rx*ry+rx*rx/4;
x=0;y=ry;
while(2.0*ry*ry*x<=2.0*rx*rx*y)
{
if(P<0)
{
x++;
p=p+2*ry*ry*x+ry*ry;
}
else
x++;y--;
p=p+2*ry*ry*x-2*rx*rx*y+ry*ry;
}
putpixel(xc+x,yc+y,RED);
putpixel(xc+x,yc-y,RED);
putpixel(xc-x,yc+y,RED);
putpixel(xc-x,yc-y,RED);
}

//Region 2
p=ry*ry*(x+0.5)*(x+0.5)+rx*rx*(y-1)*(y-1)-rx*rx*ry*ry;
while(y>0)
{
if(p<=0)
{
x++;y--;
p=p+2*ry*ry*x-2*rx*rx*y+rx*rx;
}
else
{
y--;
p=p-2*rx*rx*y+rx*rx;
}
putpixel(xc+x,yc+y,RED);
putpixel(xc+x,yc-y,RED);
putpixel(xc-x,yc+y,RED);
putpixel(xc-x,yc-y,RED);
}
getch();
closegraph();
}
```

b) write program to draw coordinates axis at the center of the screen

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

int main()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
line(300,200,300,390);
line(200,290,390,290);
getch();
closegraph();
return();
}
```