INDEX

1. Study libgraph and installation of libgraph and write a small program based on predefined funtions of libgraph.	
2. Write a program to implement DDA algorithm and use same to draw a STAR on screen.	
3. Write a program to implement a Bresenham line drawing algorithm and using same algorithm to draw a hut on a screen.	
4. Write a program to implement Circle using Mid-Point Algorithm and use same algorithm to draw an olympics ring on a screen.	
5. Write a program to implement Circle using Bresenham circle drawing algorithm and using same draw solar system on the screen.	
6. Write a program to implement the boundary fill algorithm and using the same algorithm to fill color in hut.	
7. Write a program to draw a star a on the screen and move that polygon to other part of the screen using the translation transformation.	
8. Write a program to draw a boat on the screen and rotate that boat by 90 degree using rotation transformation.	
9. Write a program a draw a photo frame on the screen and enlarge that frame by implementing scaling transformation.	
10. User has drawn a star and he is interested to see only a half part of it. Write a program to implement that condition using any line clipping algorithm.	
11. Project:	

Experiment-1

OBJECTIVE: Study libgraph and installation of libgraph and write a small program based on predefined funtions of libgraph.

What is Libgraph?

Libgraph is an implementation of the TurboC graphics API (graphics.h) on GNU/Linux using SDL. In Ubuntu, we need to install libgraph library that contains **graphics.h** header file and other important files to be able to run any graphics program . **libgraph** is an implementation of the TurboC graphics API (graphics.h) on GNU/Linux using SDL. It is simple, easy-to-use 2D graphics interface - could be used for simple prototyping, visualization or studying graphics algorithms.

Steps to install libgraph on ubuntu:

1. After that just download this libgraph installation files from the following link.(Download Libgraph files).

2. Then copy the package to the home directory. And extract it there.

3. Once it extracted open the folder. And right click on empty space and click on 'Open in terminal'.

4. In terminal execute following command one by one.

```
. /configure

 sudo make install

 sudo cp /usr/local/lib/libgraph.* /usr/lib
```

5. And done. You have successfully installed libgraph. To check its installation type man libgraph and it will show up a manual page of libgraph.

And one more thing. You can compile a graphics program by typing following.

For C++ program

```
g++ .cpp - lgraph
```

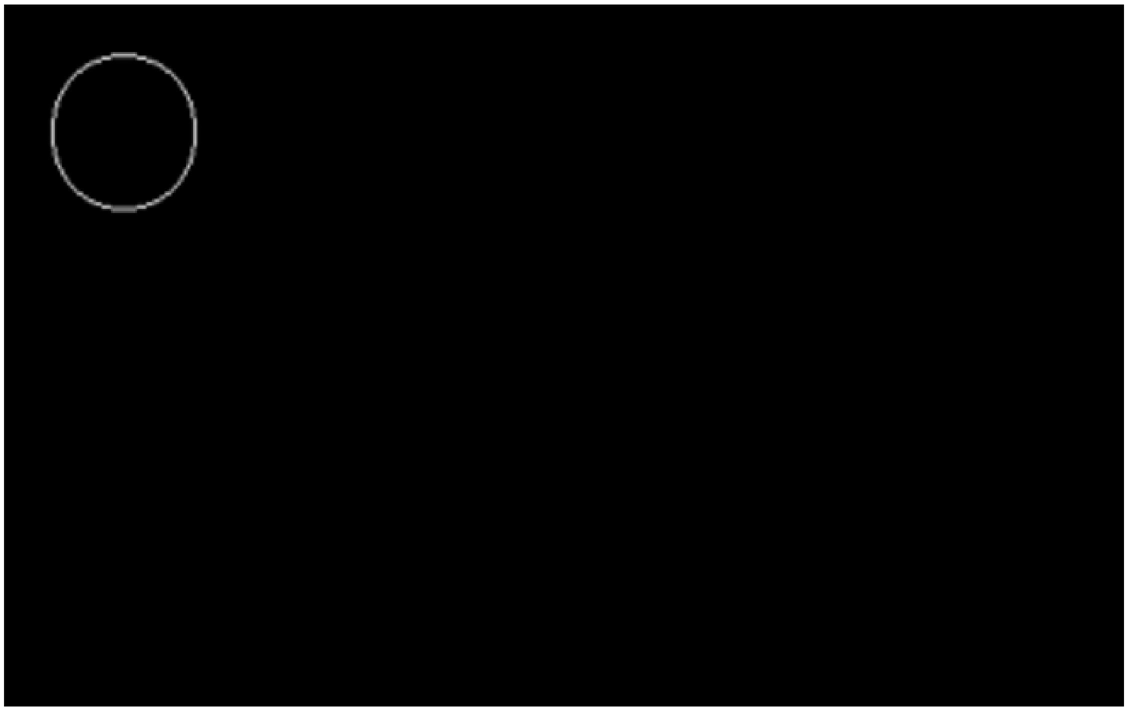
For C program

```
gcc .c - lgraph
```

SAMPLE PROGRAM

```
#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
int main()
{
	int gd = DETECT, gm;
	initgraph(&gd, &gm, NULL);

	circle(50, 50, 30);

	delay(500000);
	closegraph();
	return 0;
}
```

OUTPUT:



Experiment-2

OBJECTIVE: Write a program to implement DDA algorithm and use same to draw a STAR on screen.

DDA (Digital Differential Analyzer) Algorithm

In computer graphics, the **DDA algorithm** is the simplest algorithm among all other line generation algorithms. Here, the **DDA** is an abbreviation that stands for **"Digital Differential Analyzer"**. It is an incremental method, i.e. it works by incrementing the source coordinate points according to the values of the slope generated.

Hence, we can define DDA as follows,

"DDA stands for Digital Differential Analyzer. This algorithm is incremental and is used for the rasterization of lines, triangles, and polygons."

Working of the DDA Algorithm

Suppose we have to **draw a line PQ with coordinates P (x1, y1) and Q (x2, y2)**.

1. First, Calculate **dx = (x2 – x1) and dy = (y2 – y1)**
2. Now calculate the slope **m = (dy / dx)**
3. Calculate the number of points to be plotted (i.e. **n**) by finding the maximum of **dx** and **dy**, i.e. **n = abs (max (dx , dy))**
   To draw an accurate line, more number of points are required. Therefore, the maximum of the two values is used here.
4. Now as the n is calculated, to know by how much each source point should be incremented, calculate x_{inc} and y_{inc} as follows: **x_{inc} = (dx / n) and y_{inc} = (dy / n)**
5. Now we draw the points from **P** to **Q**. The successive points are calculated as follows: **(x_{next}, y_{next}) = (x + x_{inc}, y + y_{inc})**
   Start plotting the points from **P** and stop when **Q** is reached. In case the incremented values are decimal, use the round off values.

ALGORITHM:

1. Start.

2. Declare variables x,y,x1,y1,x2,y2,k,dx,dy,s,xi,yi and also declare gdriver=DETECT, mode.

3. Initialize the graphic mode with the path location in TurboC3 folder.

4. Input the two line end-points and store the left end-points in (x1,y1).

5. Load (x1, y1) into the frame buffer; that is, plot the first point. put x=x1,y=y1. 6. Calculate dx=x2-x1 and dy=y2-y1.

7. If abs (dx) > abs (dy), do s=abs(dx).

8. Otherwise s= abs(dy).

9. Then xi=dx/s and yi=dy/s.

10. Start from k=0 and continuing till k<s,the points will be

i. x=x+xi.

ii. Y=y+yi.

11. Plot pixels using putpixel at points (x,y) in specified colour.

12. Close Graph and stop.

Advantages of the DDA algorithm

Now, we will be looking at the advantages that the DDA algorithm offers over other line drawing algorithms.

1. It is the simplest line generation algorithm.
2. Implementation of the DDA algorithm is very easy as compared to other line generation algorithms.
3. It does not use multiplication which reduces the time complexity of implementation.
4. It is a faster and a better method than using the direct method of the line equation: i.e. **y = mx + c**

Disadvantages of the DDA algorithm

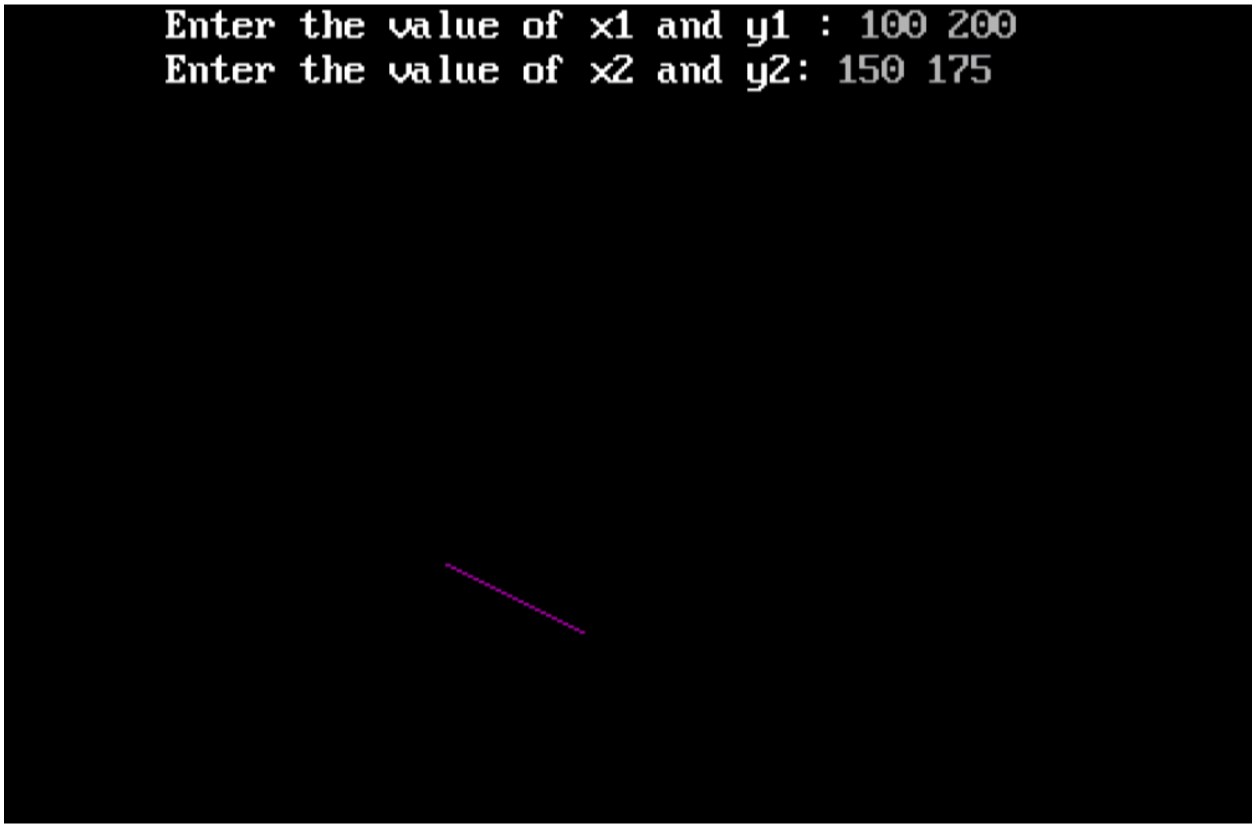
- DDA algorithm use floating-point arithmetic as it involves the use of division in the calculation of x_{inc} and y_{inc}. This floating-point arithmetic makes the algorithm time-consuming.
- The use of floating-point arithmetic decreases the accuracy of the generated points. Hence the points that we get are not accurate, i.e. they may not lie accurately on the line.
- As the points that we get from the DDA algorithm are not accurate, the lines generated by this algorithm are not smooth, i.e. some discontinuation and zigzag nature can be commonly seen in the lines drawn through this algorithm.

PROGRAM 2.1

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

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

cout<<"Enter the value of x1 and y1 : ";
cin>>x1>>y1;
cout<<"Enter the value of x2 and y2: ";
cin>>x2>>y2;

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

OUTPUT 2.1:



```
        Enter the value of x1 and y1 : 100 200
        Enter the value of x2 and y2: 150 175
```

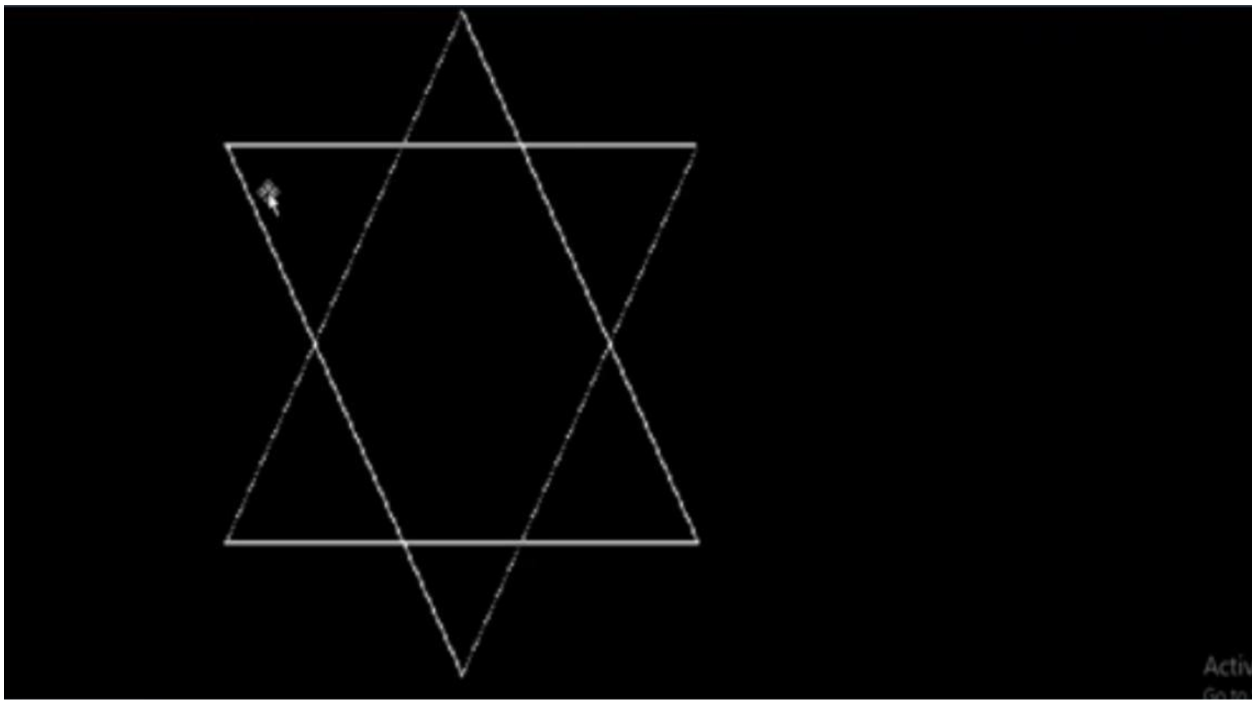
PROGRAM 2.2

```
#include<iostream>
```

```
#include<graphics.h>
using namespace std;
void star(float,float,float,float);
int main()
{
   int i,gd=DETECT,gm;
   char path[] = "";
   initgraph(&gd,”C:\\TURBOC3\\BGI”);
   star(300,200,200,400);
   star(200,400,400,400);
   star(300,200,400,400);
   star(200,250,400,250);
   star(200,250,300,450);
   star(400,250,300,450);
   getch();
   closegraph();
}

void star(float x1,float y1,float x2,float y2)
{
   float x,y,dx,dy,step;
   int i;
   dx=(x2-x1);
   dy=(y2-y1);

   if(dx>=dy)
   {
      step=abs(dx);
```

```
    }
    else
    {
       step=abs(dy);
    }
    dx=dx/step;
    dy=dy/step;
    x=x1;
    y=y1;
    i=1;
    while(i<=step)
    {
        putpixel(x,y,WHITE);
        x=x+dx;
        y=y+dy;
        i=i+1;
    }
}
```

OUTPUT 2.2


Activ
Go to

PROJECT

TOPIC: ROCKET LAUNCHING

PROGRAM

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<dos.h>
void rocket();
void text();
int gd=DETECT,gm,i,j,k;
void main()
{
text();
getch();
cleardevice();
rocket();
getch();
```

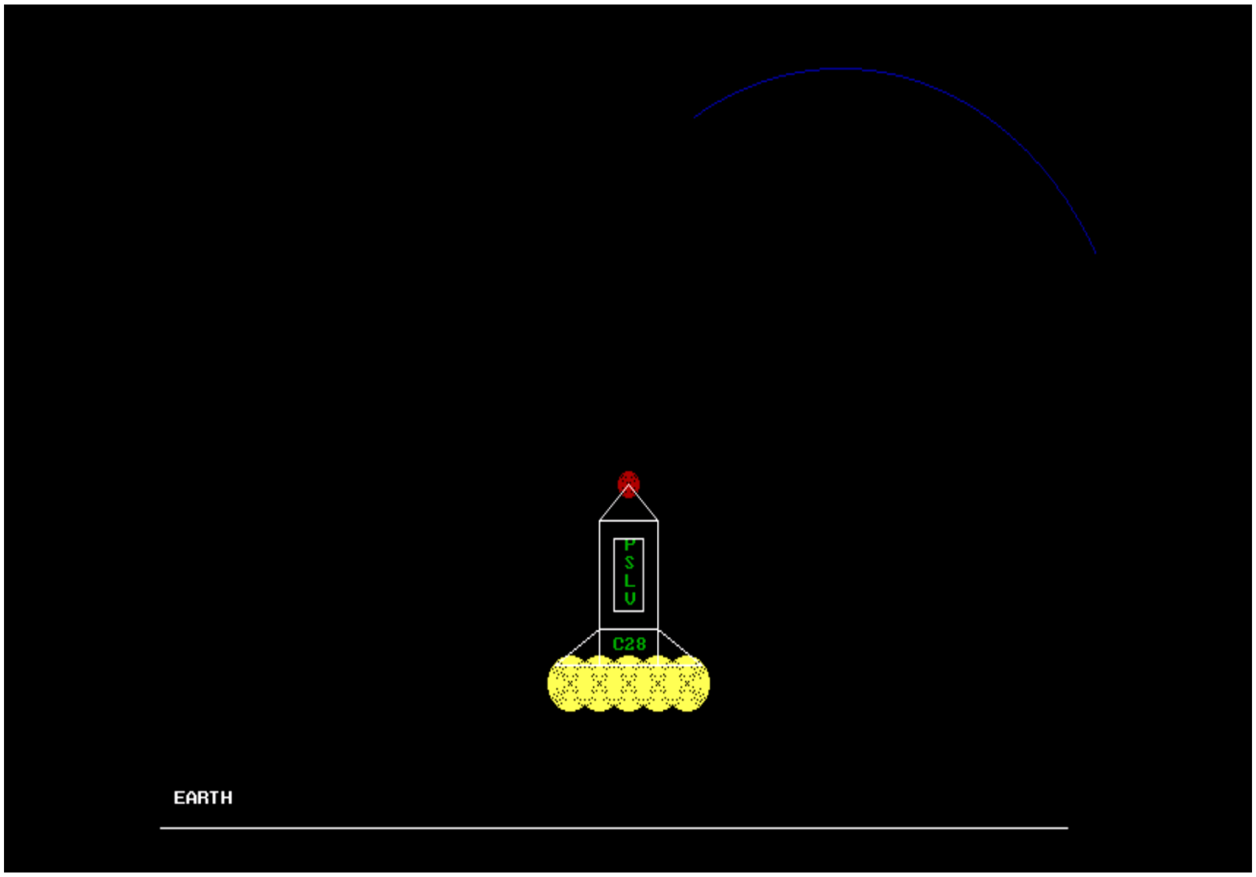
```
}

void text()
{
initgraph(&gd,&gm,"C:\\Turboc3\\BGI");
settextstyle(0,EMPTY_FILL,1);
setcolor(YELLOW);
outtextxy(50,50,"I");
outtextxy(50,60,"S");
outtextxy(50,70,"R");
outtextxy(50,80,"O");
settextstyle(0,EMPTY_FILL,2);
setcolor(BLUE);
outtextxy(100,50,"Indian Space");
outtextxy(100,70,"Research Organisation");
settextstyle(0,EMPTY_FILL,1);
setcolor(WHITE);
outtextxy(50,120,"WELCOME TO INDIAN SPACE
REASERCH ORGANISATION.");
delay(150);
outtextxy(50,150,"TODAY 10-JULY-2015 PSLV - C28
IS READY TO LAUNCH.");
```

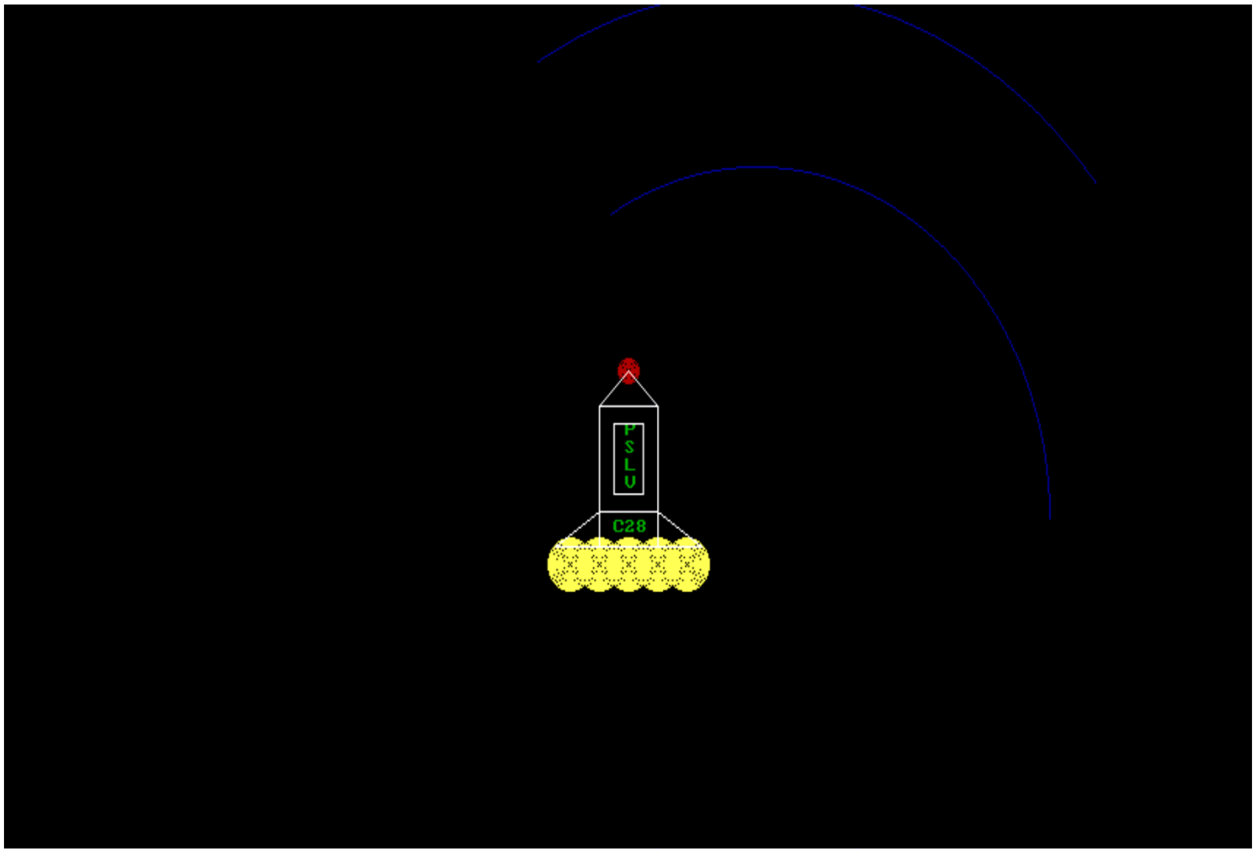
```
	delay(150);
	outtextxy(50,180,"IT CARRIES 5 SATELLITES TO
DEPLOYE IN THE SUN-");
	delay(150);
	outtextxy(50,210,"SYNCHRONOUS ORBIT.");
	delay(150);
	setcolor(RED);
	outtextxy(50,240,"LAUNCH MASS => 320,000 KG");
	delay(150);
	outtextxy(50,270,"PAYLOAD MASS => 1,440 KG");
	delay(150);
	outtextxy(50,300,"LAUNCH SITE => SATISH
DHAWAN SPACE CENTER");
	delay(150);
	outtextxy(50,330,"DISTANCE TRAVELL => 647 KM");
	delay(150);
	outtextxy(50,360,"PAYLOAD => 3 DMCE Satellites, 1
CBNT-1 (Technology Demonstrator)");
	delay(150);
	outtextxy(50,390,"& 1 D-O (TD Nano Satellite)");
	delay(150);
	outtextxy(50,420,"SO... COUNTDOWN IS BEGIN");
```

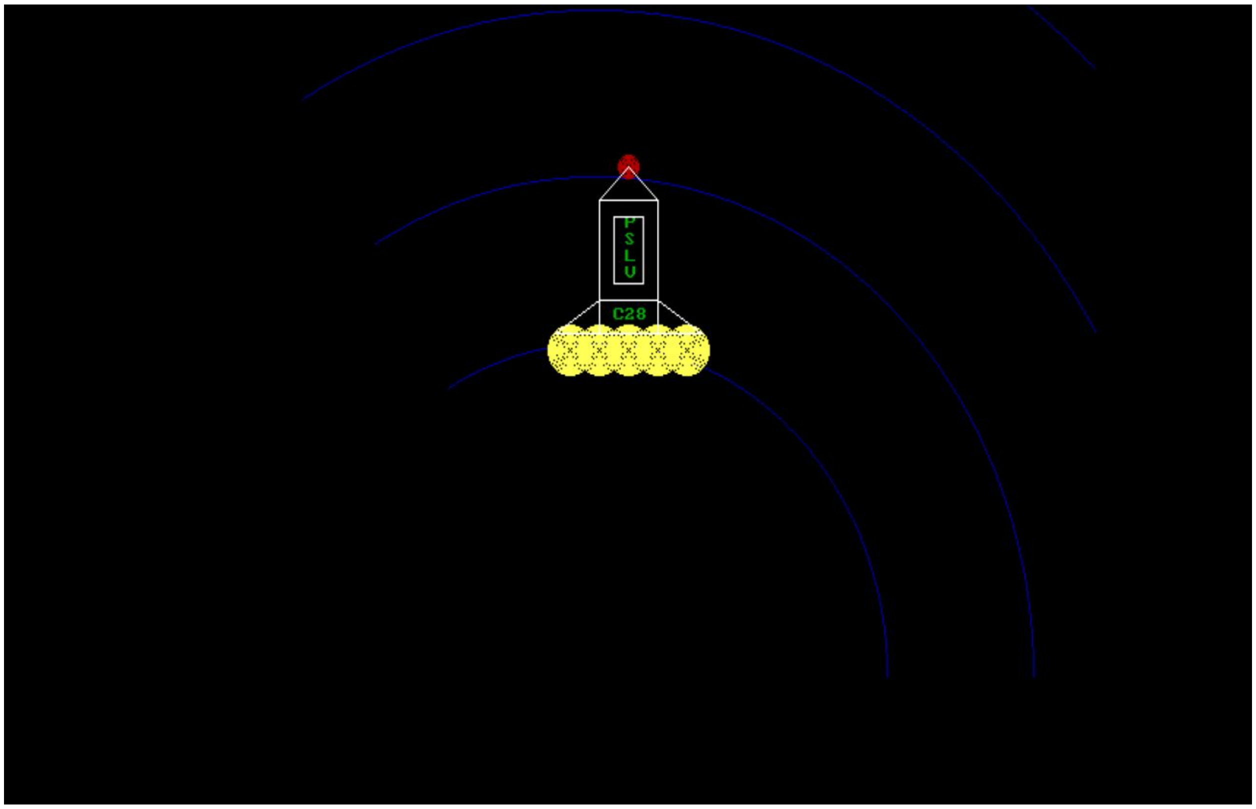
```
delay(150);
outtextxy(50,450,"ENTER TO CONTINUE");
delay(150);
}

void rocket()
{
setcolor(BLUE);
 //earth
    for(i=30;i<=400;i++)
    {
setcolor(BLUE);

arc(500-i,200+i,0,120,200);
arc(500-i,200+i,0,120,300);
arc(500-i,200+i,0,120,400);
arc(500-i,200+i,0,120,500);
setcolor(GREEN);
outtextxy(318,330-i,"P");
outtextxy(318,340-i,"S");
outtextxy(318,350-i,"L");
outtextxy(318,360-i,"V");
outtextxy(310,385-i,"C28");
//rocket
setcolor(WHITE );
outtextxy(10,400+i,"EARTH");
for(j=0;j<=7;j++)
{
setcolor(RED);
circle(320,300-i,0+j);
}
for(k=0;k<=15;k++)
{
setcolor(YELLOW);
circle(300,410-i,0+k);
circle(340,410-i,0+k);
circle(320,410-i,0+k);
circle(280,410-i,0+k);
circle(360,410-i,0+k);
}
setcolor(WHITE);
line(0,420+i,620,420+i);
line(340,400-i,340,320-i);
line(300,400-i,300,320-i);
line(330,370-i,330,330-i);
line(310,370-i,310,330-i);
line(310,330-i,330,330-i);
line(310,370-i,330,370-i);
line(300,380-i,340,380-i);
line(270,400-i,370,400-i);
line(270,400-i,300,380-i);
line(340,380-i,370,400-i);
line(300,320-i,340,320-i);
line(300,320-i,320,300-i);
line(340,320-i,320,300-i);
delay(50);
cleardevice();
}
setcolor(GREEN);
settextstyle(0,EMPTY_FILL,5);
outtextxy(50,100,"CONGRATULATIONS");
outtextxy(50,200,"MISSION");
outtextxy(50,300,"SUCCESSFUL");
```

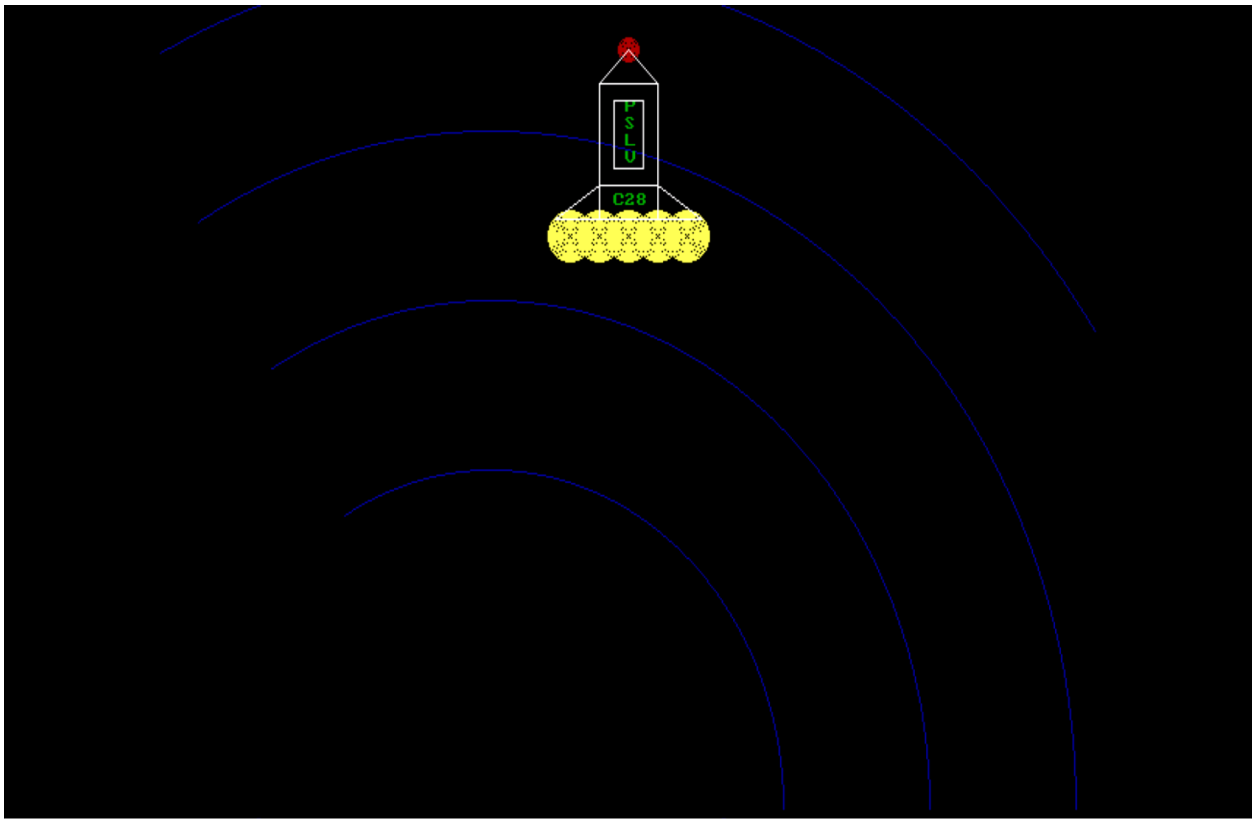
```
getch();
}
```

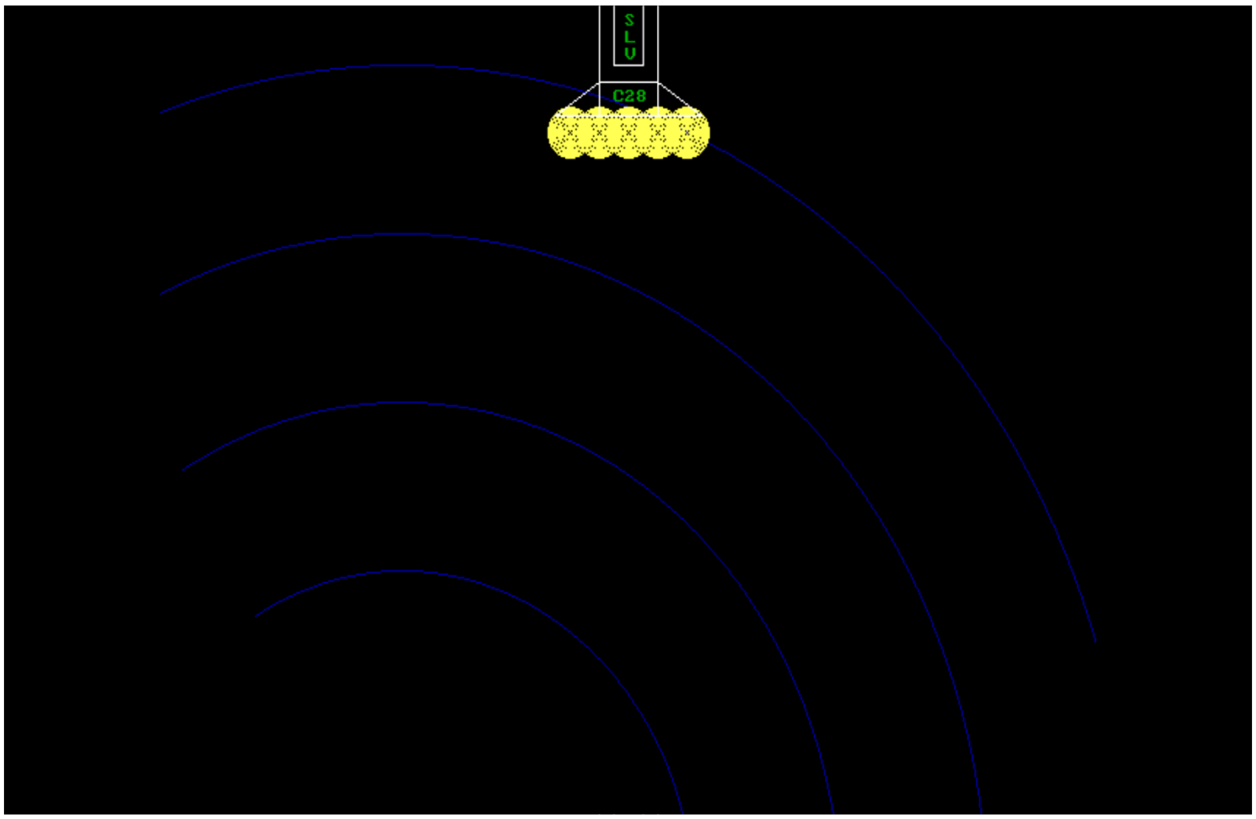
OUTPUT

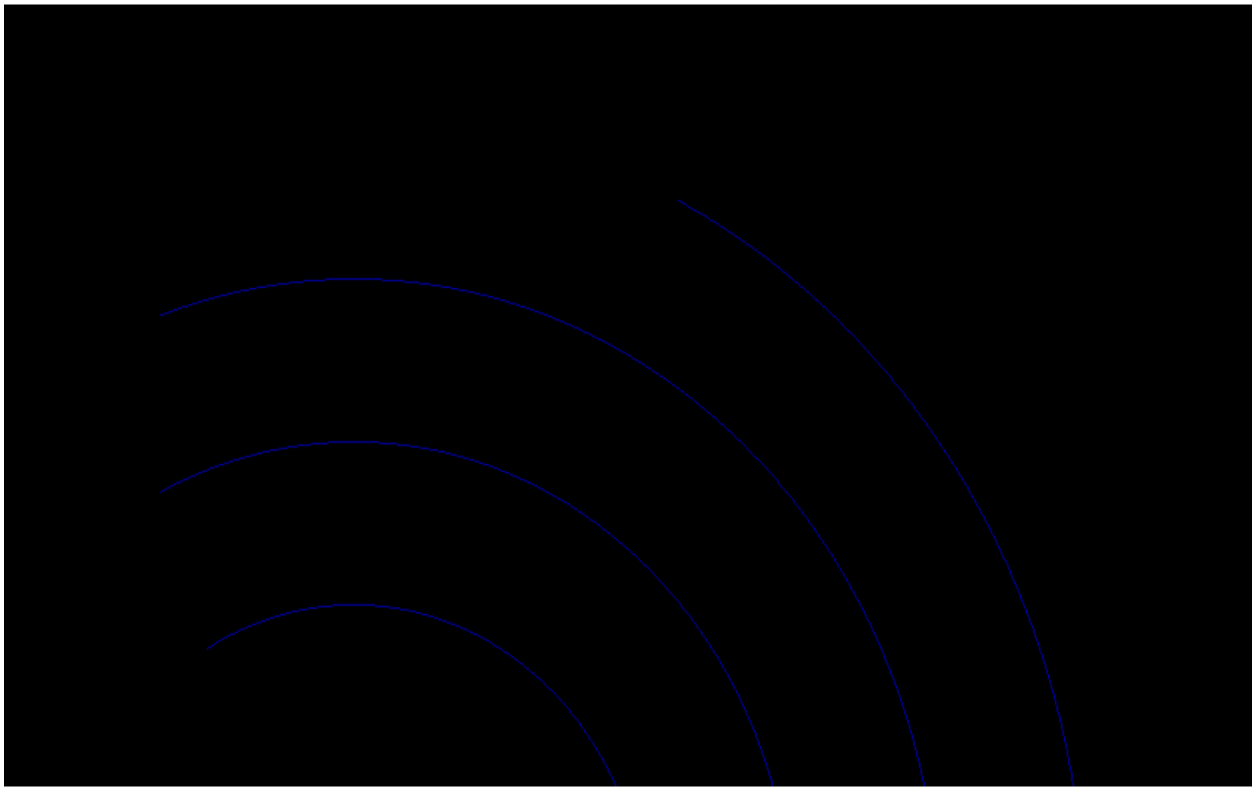


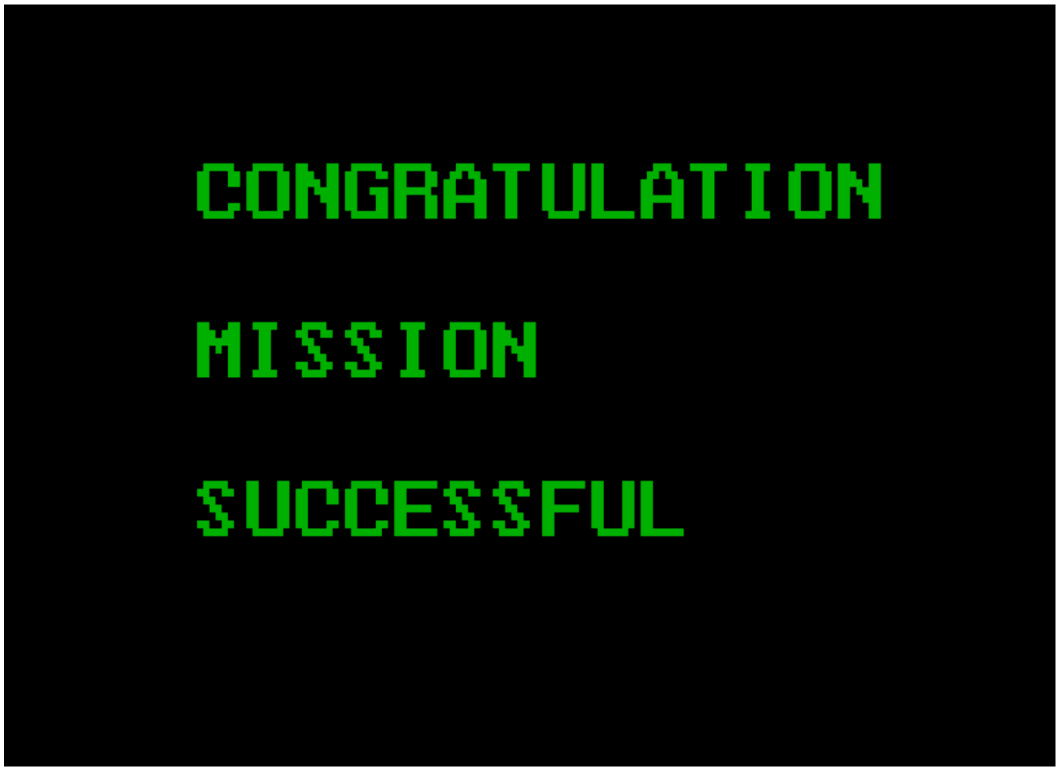

PSLV
C28


PSLV
C28


PSLV
C28


SLV
C28





CONGRATULATION

MISSION

SUCCESSFUL