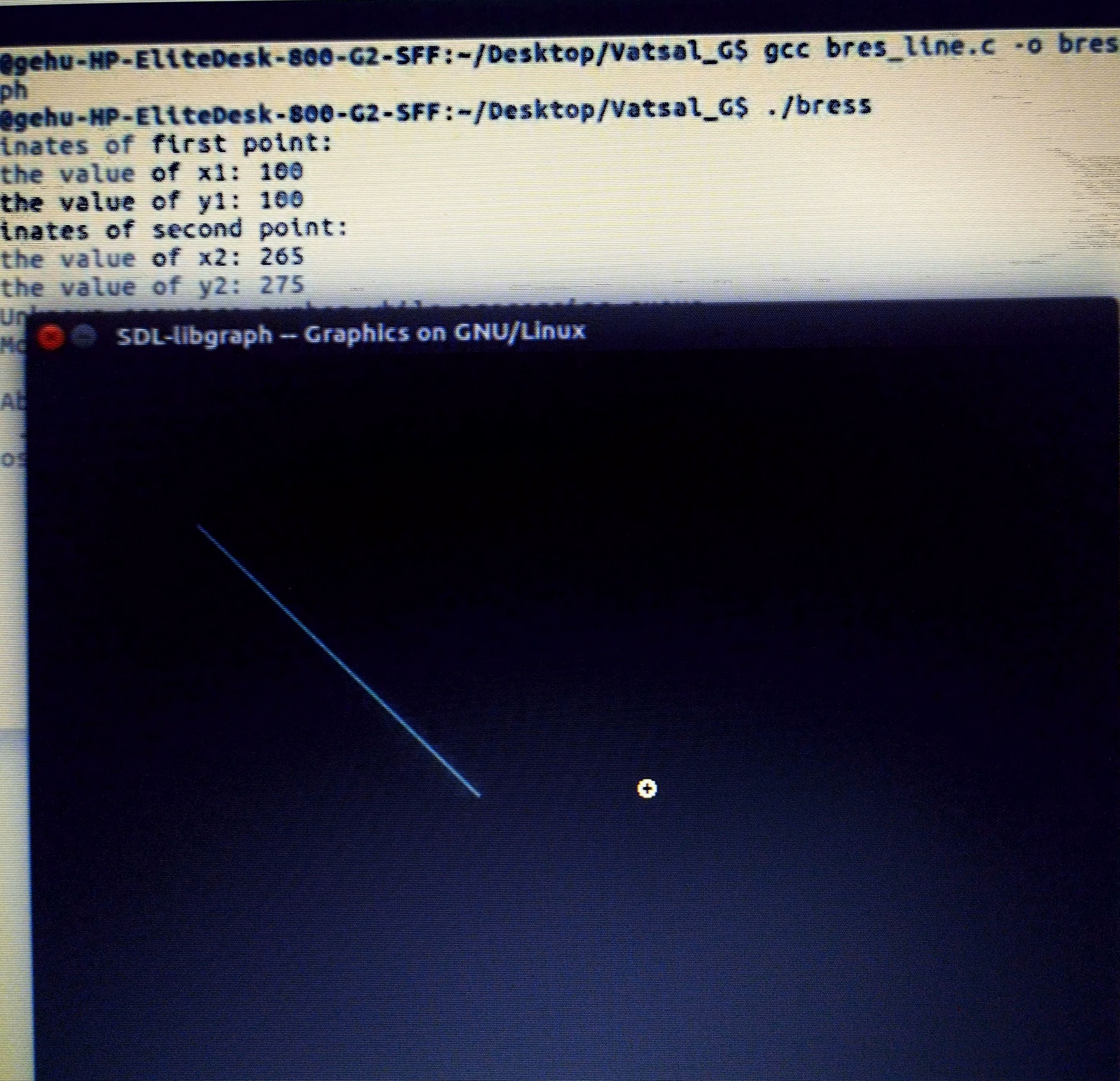
Name - Vipin Rawat University RollNo - 1121169 Subject Code - PBC-602
Subject Name - Computer Graphics and Animation

Ans:1-

```
#include <stdio.h>
#include <graphics.h>
int main()
{
int rou (float num)
{
return num<0? num-0.5: num+0.5;
}
int x1=100, x2=300, y1=100, y2=200;
int gd = DETECT, gm;
float pk, Pk, x, y, step;
int dx = x2-x1;
int dy = y2-y1;
pk = 2*dx-dy;
if (dx>dy)
step = dx;
else
step = dy;
initgraph(&gd&gm,"");
outtextxy(x1,y2,"A");
outtextxy(x2,y2,"B");
putpixel(x1,y1,WHITE);
x=x1, y=y1;
while (step>0)
{
```

```
if(pk<0)
{
pkk=pk+2*dy;
}
else
{
pkk = pk+2*dy-2*dy;
 y++;
}
putpixel(rou(x),rou(y),WHITE);
 x++;
step--;
}
getch();
return 0;
}
```



```
@gehu-HP-EliteDesk-800-G2-SFF:~/Desktop/Vatsal_G$ gcc bres_line.c -o bres
ph
@gehu-HP-EliteDesk-800-G2-SFF:~/Desktop/Vatsal_G$ ./bress
inates of first point:
the value of x1: 100
the value of y1: 100
inates of second point:
the value of x2: 265
the value of y2: 275
Un
Mo

Ab

os
```

~~A~~ Bresenham's line Algorithm :-

Step1: Start algorithm

Step2: Declare variable $x_1, x_2, y_1, y_2, i_1, i_2, dx, dy$

Step3: Enter value of x_1, y_1, y_2, x_2

where x_1, y_1 are coordinates of starting point

and x_2, y_2 are coordinates of ending point

Step 4: Calculate $dx = x_2 - x_1$

Calculate $dy = y_2 - y_1$

Calculate $i_1 = 2*dy$

Calculate $i_2 = 2*(dy - dx)$

```
Calculate d = $i_1 - dx$
Step5: Consider (x,y) as starting point and x_end as maximum
  possible value of x
      if $dx < 0$
          then $x = x_2$
          $y = y_2$
          $x_{end} = x_1$
      if $dx > 0$
          then $x = x_1$
          $y = y_1$
          $x_{end} = x_2$
Step6: Generate point at (x,y) coordinates.
Step7: check if whole line is generated.
      if $x >= x_{end}$
      Stop.
Step8: calculate coordinates of the next pixel
      if $d < 0$
          then $d = d + i_1$
      if $d > 0$
          then $d = d + i_2$
          increment $y = y + 1$
Step9: increment $x = x + 1$
Step10: Draw a point of latest (x,y) coordinates
Step11: Go to Step 7
Step12: end of algorithm
```