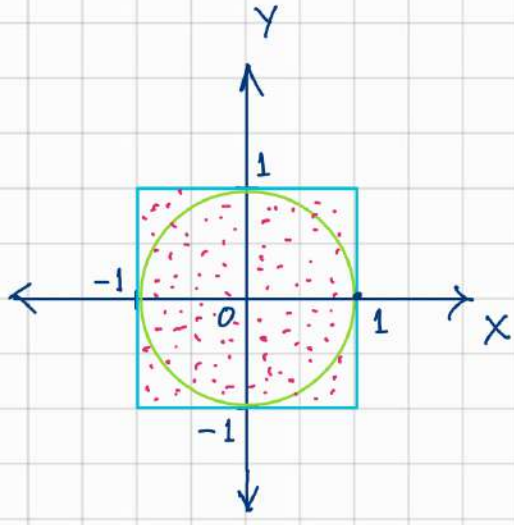
Aim : To find the value of π by calculating the area of a circle using random number generation.

Theory :



We consider a circle with radius of 1 unit centred at the origin. The area of the circle is -

$$A_c = \pi 1^2 = \pi \longrightarrow (i)$$

The area of the square in which the circle is enclosed has a side of 2 units. The area of the square is -

$$A_s = 2^2 = 4. \longrightarrow (ii)$$

$$(i) \div (ii) \Rightarrow \frac{\pi}{4} = \frac{A_c}{A_s}$$

$$\Rightarrow \pi = 4 \frac{A_c}{A_s}$$

If we randomly generate N number of points with coordinates (x, y) such that $x \in [-1, 1]$ and $y \in [-1, 1]$ then

(i) the area of the circle is proportional to the number of randomly generated points inside the circle.

If $\sqrt{x_i^2 + y_i^2} \leq 1$ then (x_i, y_i) lies inside the circle.

Suppose M is the number of points lying inside the circle.

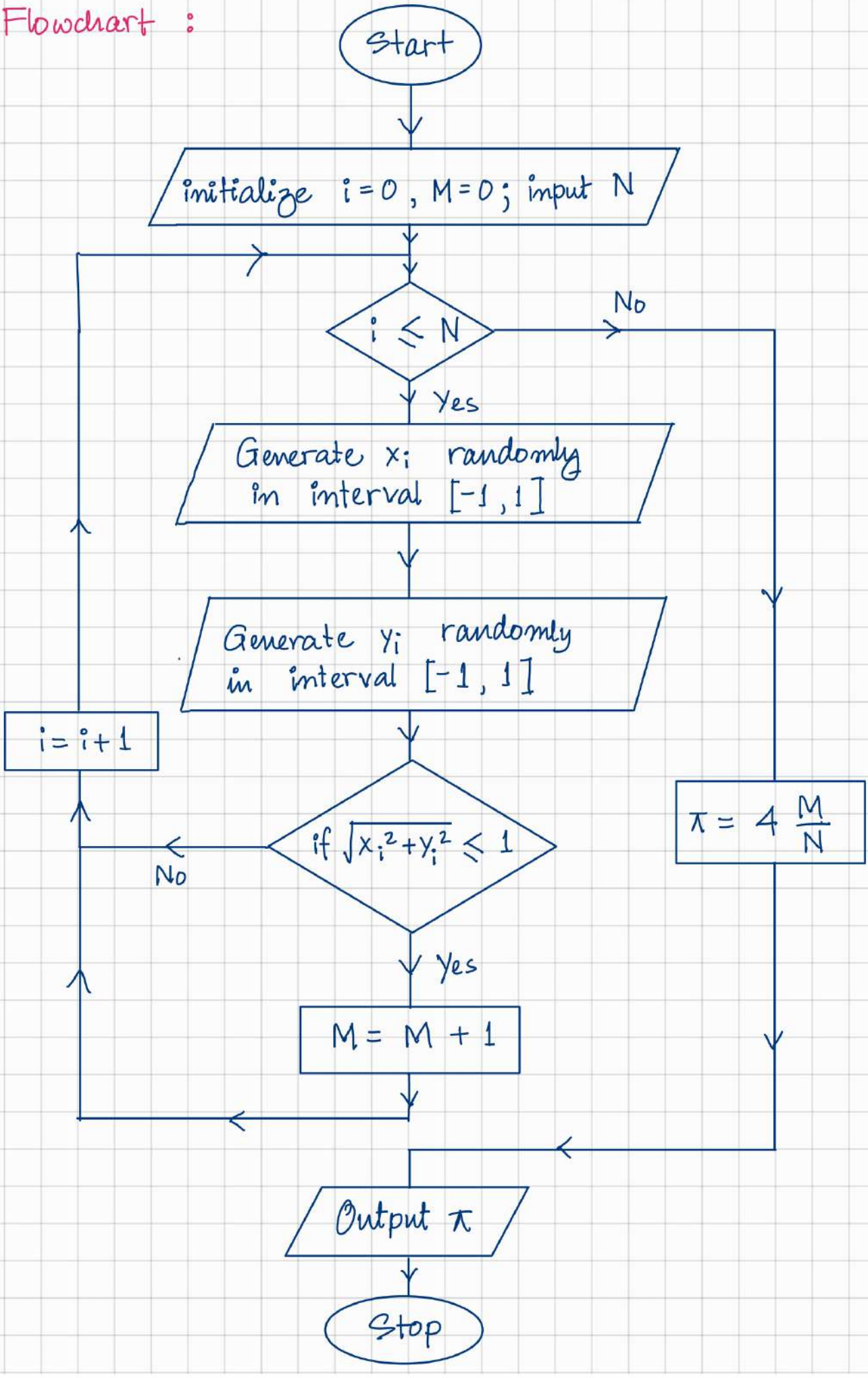
Then, $A_c \propto M$

(ii) the area of the square is proportional to total number of points N.

i.e. $A_s \propto N$.

Thus, $\boxed{\pi = 4 \frac{M}{N}}$

Flowchart :


Start
initialize i = 0 , M = 0 ; input N
i ≤ N
No
Yes
Generate x_i randomly in interval [-1 , 1]
Generate y_i randomly in interval [-1 , 1]
i = i + 1
if $\sqrt{x_i^2 + y_i^2} \leq 1$
No
Yes
M = M + 1
$\pi = 4 \frac{M}{N}$
Output π
Stop

```c
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
#include<time.h>


int main()
{
    srand(time(NULL));
    int i,n,count;
    double x,y,dist,pi;
    printf("Enter the value of N : \n");
    scanf("%d",&n);
        for(i=0;i<n;i++)
        {
            x = (double) rand()/RAND_MAX;
            y = (double) rand()/RAND_MAX;
            dist = sqrt(x*x + y*y);
            if(dist<=1)
            count++;
        }

    pi = ((double) count/n)*4;

    printf("Pi = %lf \n",pi);
    return 0;
}
```

Observation :

Serial No.	N	π
1	1	
2	10	
3	1000	
4	1000000	

Result : The value of π is ____

Discussions :

1. The accuracy in the calculated value of π increases when the number of randomly generated points increases
2. The number of randomly generated points should be uniformly distributed.