Objective:

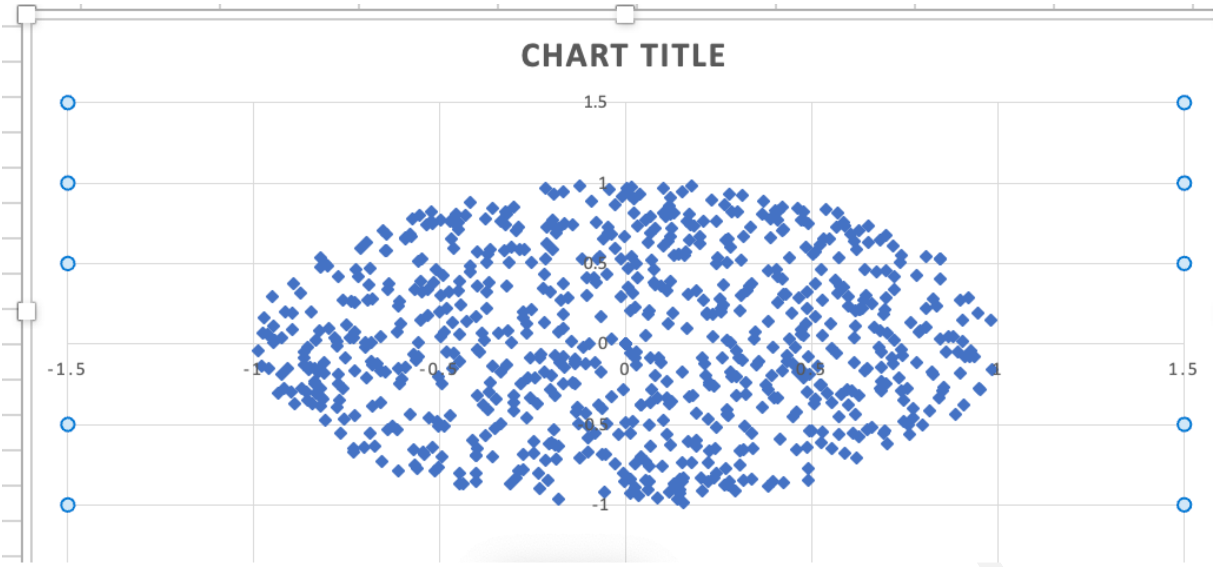
- **To implement a Monte Carlo simulation in a spreadsheet and visualize the results using a graph.**
- **To generate Random Number in a certain range.**

Experiment 1: Monte Carlo Simulation using Spreadsheet

Procedure:

1. **Generate Random Numbers:**
    - Created two columns (Column C and Column D) with 500 random numbers each.
    - Formula used to generate random numbers:
        - =1-2*RAND()
2. **Circle Equation Validation:**
    - Added another column to check if points fall inside the unit circle.
    - Formula used:
        - =IF((A1*A1+B1*B1) < 1, A1, 0)

        This checks if the point lies within the circle by evaluating if x^2 + y^2 < 1.
    - A similar column was created for the second coordinate:
        - =IF((A1*A1+B1*B1) < 1, B1, 0)
3. **Graphical Representation:**
    - Plotted a graph using the validated data points.
    - The resulting graph displayed a circle consisting of 1000 discrete points.



Results: The visualization successfully showed a circle formed by random points within the boundary of x^2 + y^2 = 1, validating the implementation.

Experiment 2: Random Number Generation in a Range

Objective: To generate a series of random numbers within a user-defined range using C.

Code Implementation:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include<conio.h>

void printRandoms(int min, int max, int count)
{
   srand(time(0));
   printf("Random number between %d and %d\n", min, max);
   for (int i = 0; i < count; ++i)
   {
      int randomNum = min + rand() % (max - min + 1);
      printf("%d ", randomNum);
   }
   printf("\n");
}

int main()
{
   int min, max, count;

   printf("Enter minimum value: ");
   scanf("%d", &min);

   printf("Enter maximum value: ");
   scanf("%d", &max);

   printf("Enter number of random numbers to generate: ");
   scanf("%d", &count);

   printRandoms(min, max, count);

   return 0;
}
```

Output:

```
Enter minimum value: 100
Enter maximum value: 200
Enter number of random numbers to generate: 20
Random number between 100 and 200
184 195 186 194 180 106 155 149 156 103 145 172 168 139 161 180 149 178 113 121
```

Results:

- Successfully generated a series of random numbers within the specified range.
- The user can define the range and count of numbers as inputs.

Conclusion:

In this lab session, we explored Monte Carlo simulations and random number generation using both spreadsheet tools and C programming. The key takeaways include: The use of random number generation for simulation purposes.

- Estimating mathematical constants like π using the Monte Carlo method.
- Implementing practical programs to generate random data for various applications.
- Visualizing results through graphs and interpreting them effectively.

These exercises provide a foundation for understanding simulation and modeling techniques in computational and statistical applications.