Estimating π using the Monte Carlo Method

**Q)** Write a Python program to simulate the calculation of π using Monte Carlo method. Your program draws the circle and the square and plot the random points in the graph as shown in Figure

1. You need simulate different numbers of points (at least include 10, 500, 1,000, 5,000, 10,000, 50,000, 80,000, 100,000, 200,000) and calculate the value π for each different number of points. Then draw a line chart to show the change of the accuracy of the value of π calculated when the number of points is increased. Submit Python code (5 points, no points will be given for a program that doesn’t correctly implement the simulation algorithm), and simulation results including the graph plots (2 point), π values (2 point), and line chart (1 point).

# Python Code:

import numpy as np

import matplotlib.pyplot as plot pi\_value =[]

r=[ 10, 500, 1000, 5000, 10000, 50000, 80000, 100000, 200000]

for k in r: i=np.random.uniform(0,1, k ) j=np.random.uniform(0,1, k )

distance=(i-0.5)\*\*2+(j-0.5)\*\*2<0.25 pi\_value\_itr =4\*np.sum(distance)/k pi\_value.append(pi\_value\_itr) x=i[distance]

x1=j[distance] fig=plot.figure(figsize=(5,5)) y = fig.add\_subplot(1, 1, 1)

rectangle=plot.Rectangle((0,0), 1, 1,edgecolor='k',fill=False) cir=plot.Circle((0.5,0.5),radius=0.5,edgecolor='k',fill=False) y.add\_patch(rectangle)

y.add\_patch(cir) plot.scatter(i,j,s=2,color='#DAA520') plot.scatter(x,x1,s=2,color='#9ACD32') print(pi\_value\_itr)

# Scatter Plots for Different Samples

* + Samples = [ 10, 500, 1000, 5000, 10000, 50000, 80000, 100000, 200000]
  + If the point's distance is less than zero, then it is considered within the circle. Else, the point lies outside the circle.
  + The points that lie inside the circle are in yellow green and the points that lie outside are in goldenrod.

# Simulation Results:

2.8

3.16

3.152

3.1432

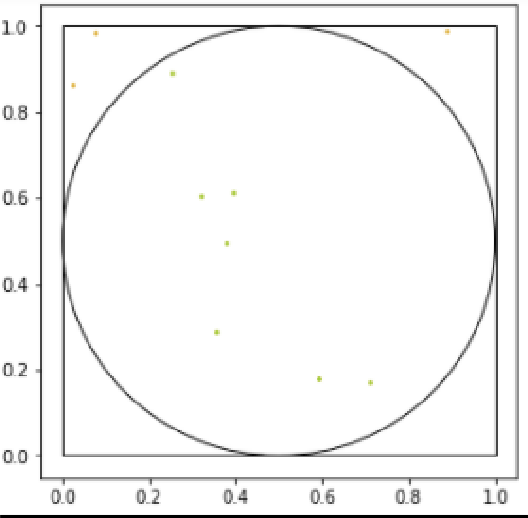
3.1528

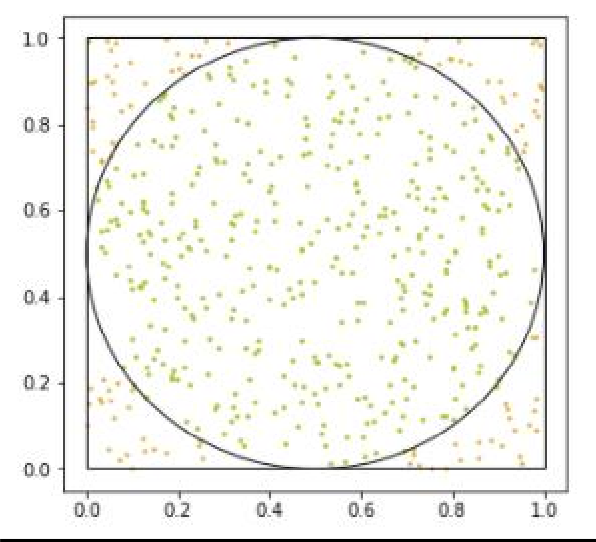
3.14072

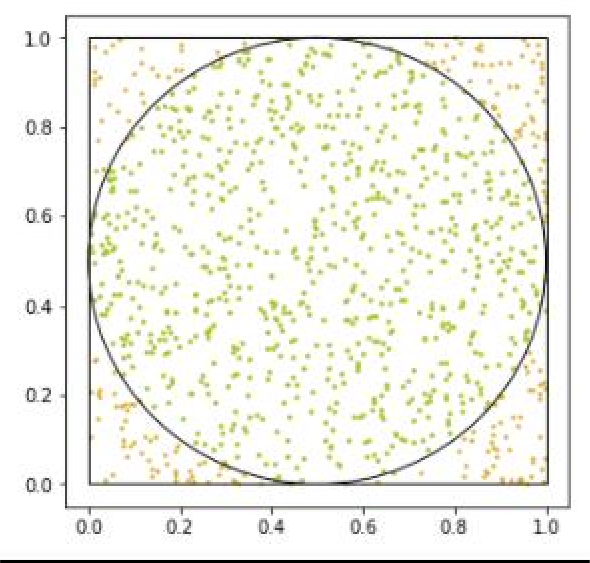
3.1312

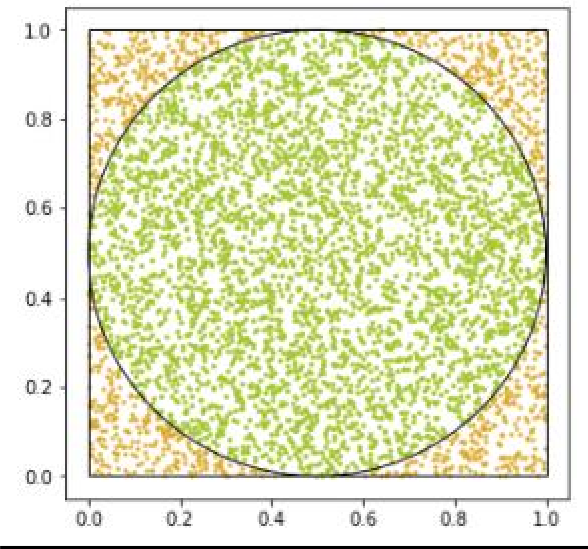
3.14528

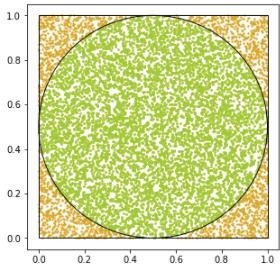
3.13994

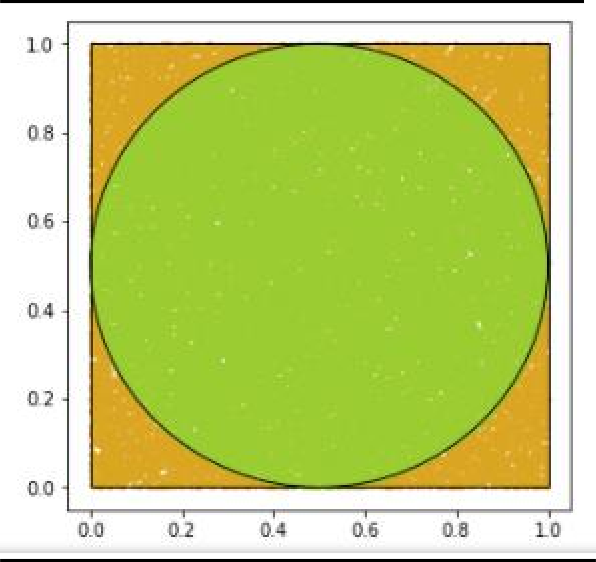


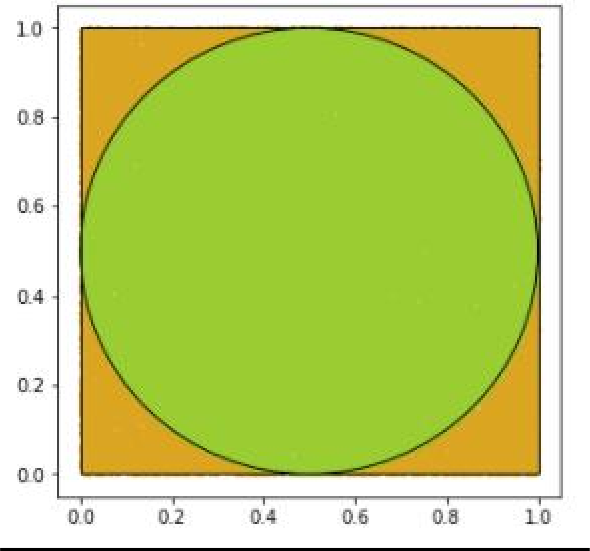


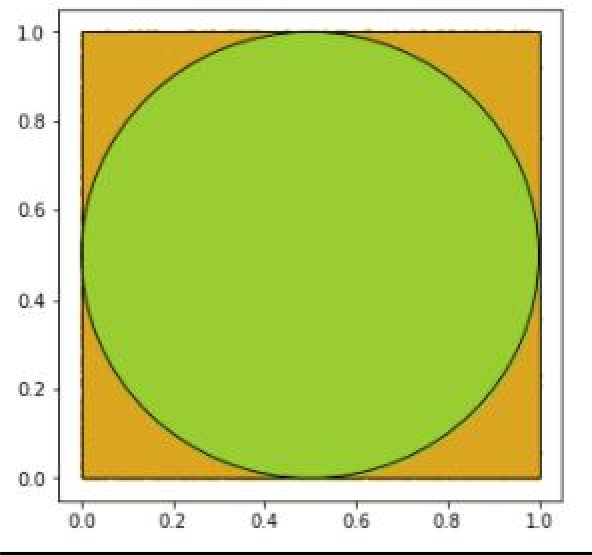


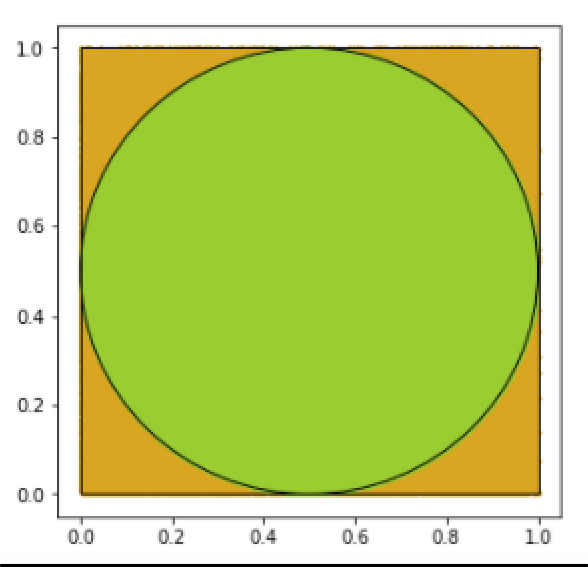












# For pi\_values

## Code:

pi\_value

## Output:

[2.8,

3.16,

3.152,

3.1432,

3.1528,

3.14072,

3.1312,

3.14528,

3.13994]

**Line Chart:**

## Code:

pi\_value plot.plot(r,pi\_value)

## Output:

[<matplotlib.lines.Line2D at 0x1a1db861ee0>]

