Physics -1 Lab SM203P

Experiment - 3

Understanding Non-Linear Systems with Logistic Maps

Group24

IMT2020065 Shridhar Sharma

IMT2020553 Abhinav Mahajan

IMT2020539 Shaurya Agrawal

IMT2020085 Harshadeep Donapati

IMT2020126 Ayushmaan Singh

IMT2020057 Vishnutha Sheela

Summary :

In this experiment we have observed a system that grows into chaos with the variation of only one variable in a seemingly simple equation. Every member has plotted a graph in Python for the equation:

$$X_{n+1} = r \cdot X_n(1 - X_n)$$

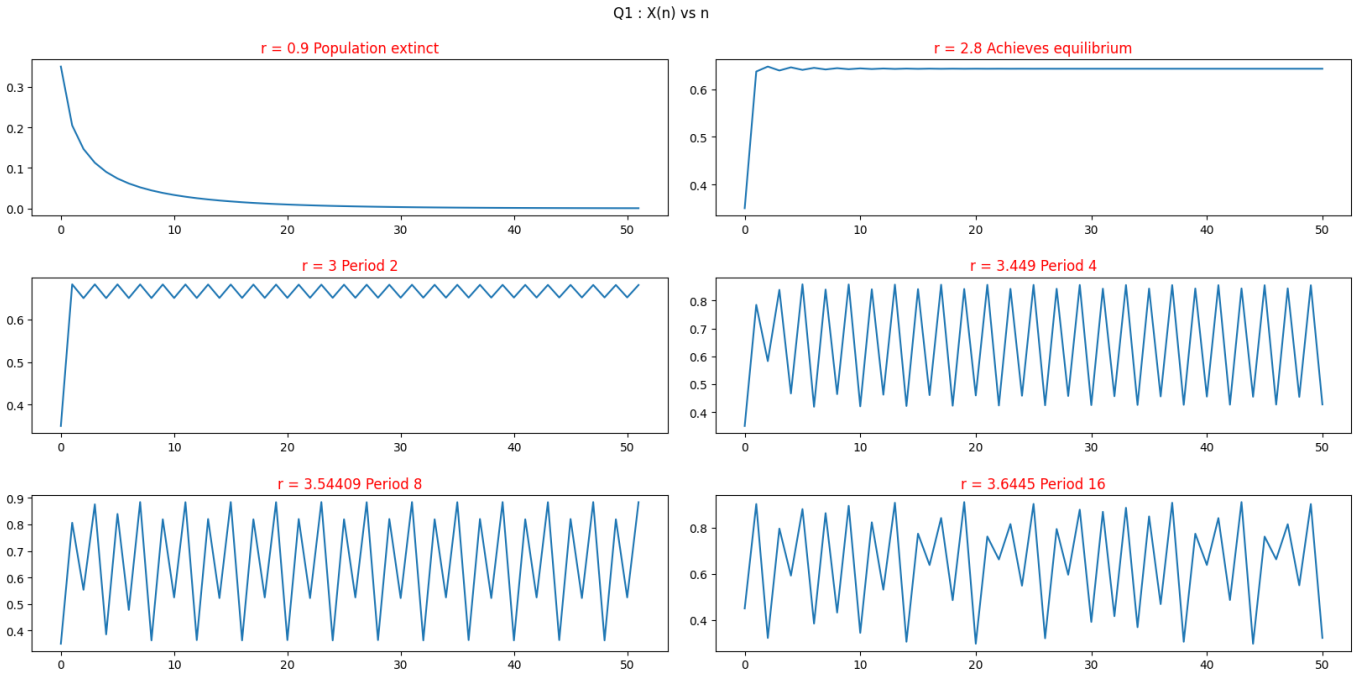
The time-series plots and bifurcation graphs for values of r given in the assignment have been plotted separately by everyone, as well as the code has been attached.

Observations:

1. For values of r in the range of 1 to 3, there is a single value that X takes, independent of the number of iterations.
2. As r increases beyond 3, the values that X can take up increases to 2 first, and then 4, and then 8, and so on. The number of values keeps increasing with the increase of r.
3. The X values repeat themselves in a periodic fashion.
4. The aforementioned increase, however, ceases at certain values of r, where it merges to fewer values and the bifurcation starts again.
5. Although the system seems chaotic at first glance, there is a certain periodicity to it which can be seen in the Feigenbaum's constant. Taking the ratio of the difference between consecutive values of r where the bifurcations happen yields a constant value.
6. Therefore, this constant can be useful in predicting chaotic behaviours of this model.

Shridhar Sharma (IMT2020065) -

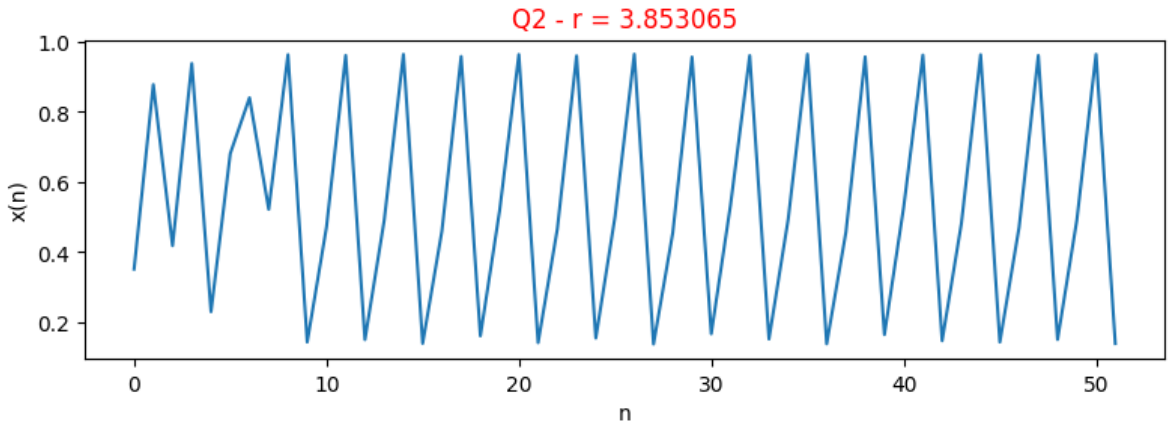
Question 1 -



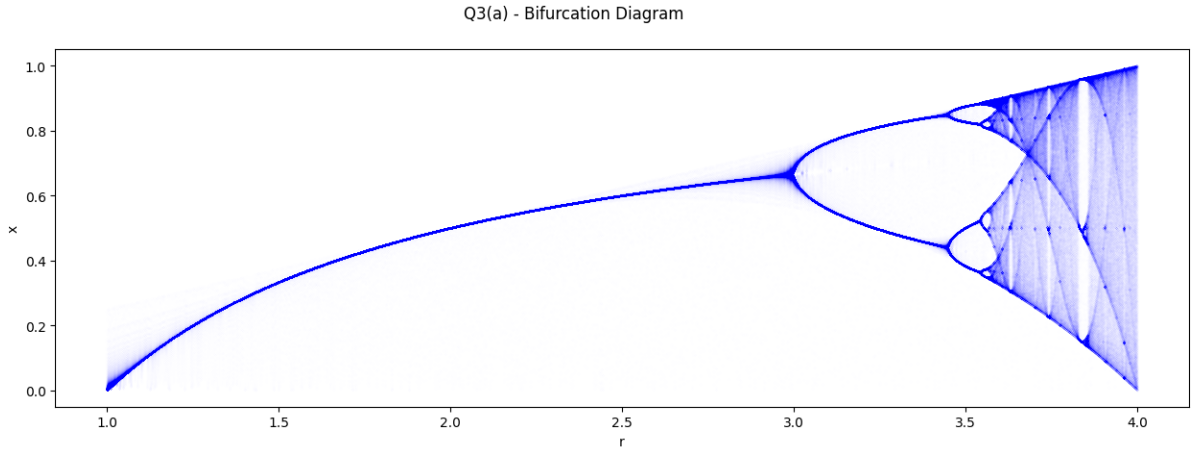
Code for Q1 and Q2 -

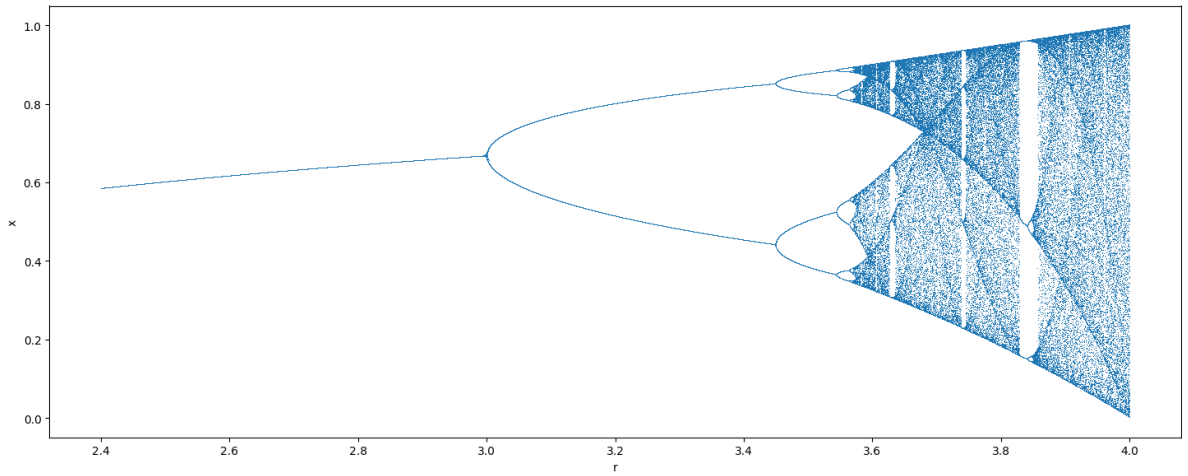
```
import matplotlib.pyplot as plt
import numpy as np

fig = plt.figure()
r = 3.853065
y = [0.35]
for i in range(51):
    y.append(r*y[i]*(1 - y[i]))

plt.plot(y)
plt.show()
```

Question 2 -



Question 3(a) -

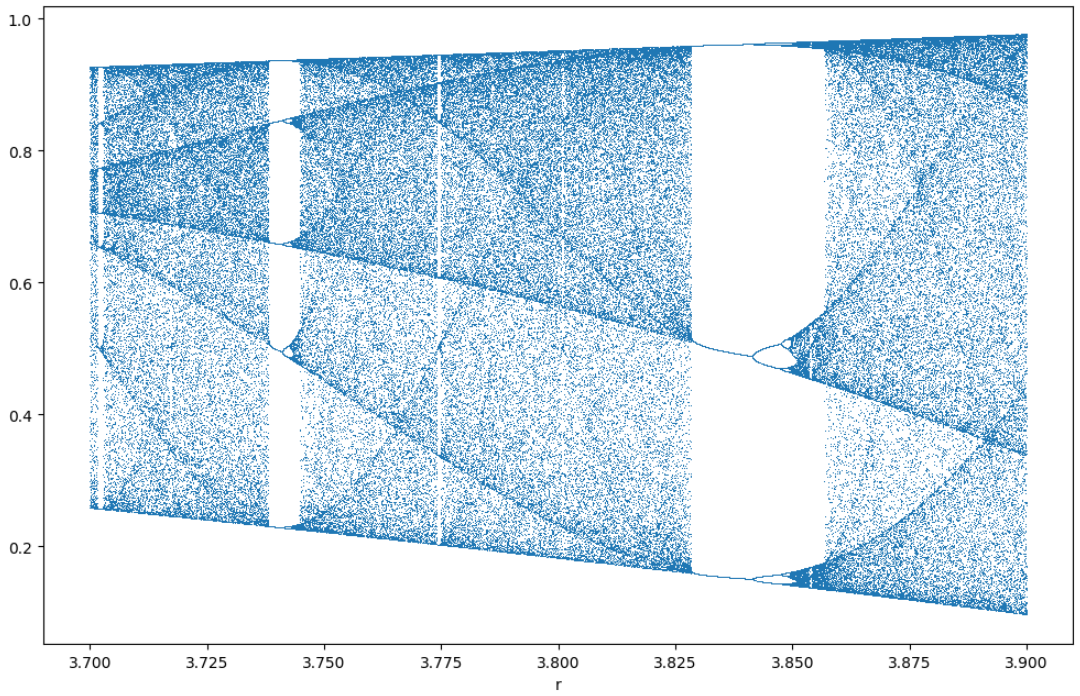




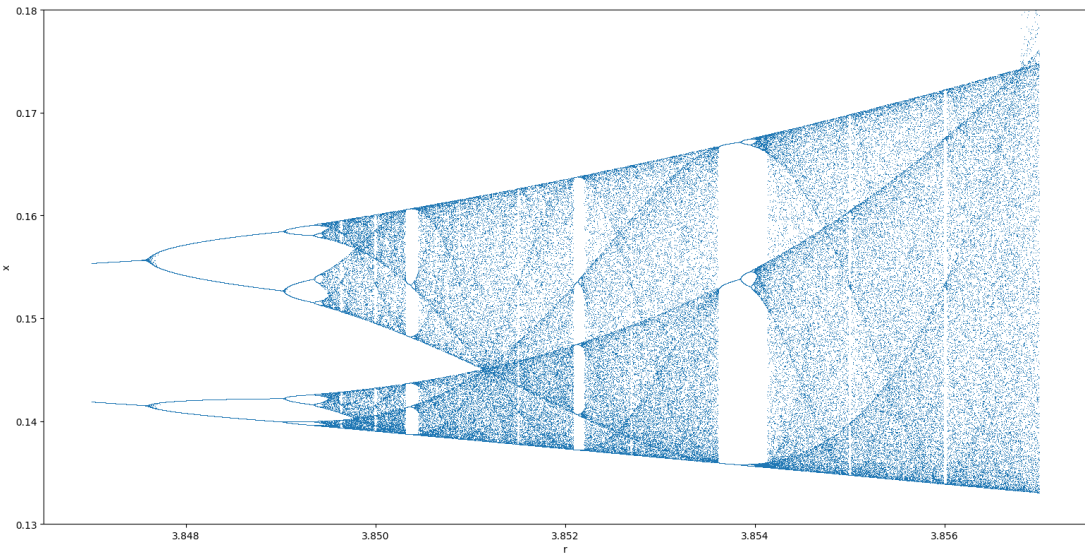
Code for Q 3(a) and 3(b) -

```
import matplotlib.pyplot as plt
import numpy as np


r = np.linspace(1, 4, 10000)


y = []
for i in r:
    temp = []
    x = np.random.random()
    for j in range(500):
        x = i*x*(1-x)
        temp.append(x)
    y.append(temp)


plt.subplot(2,1,1)
plt.suptitle("Q3(a) - Bifurcation Diagram")
plt.xlabel("r")
plt.ylabel("x")
Plt.plot(r,y, ls='', marker='.', color='b', ms = 0.009, alpha=.25)
plt.show()
```

Question 3(b) -



From the above zoomed image we can see that a $r > r_{\infty}$ exists, which has a stable 3-period cycle. Further zooming in we can see that all 3 branches have a self similar structure like that of a fractal.



Question 4 -

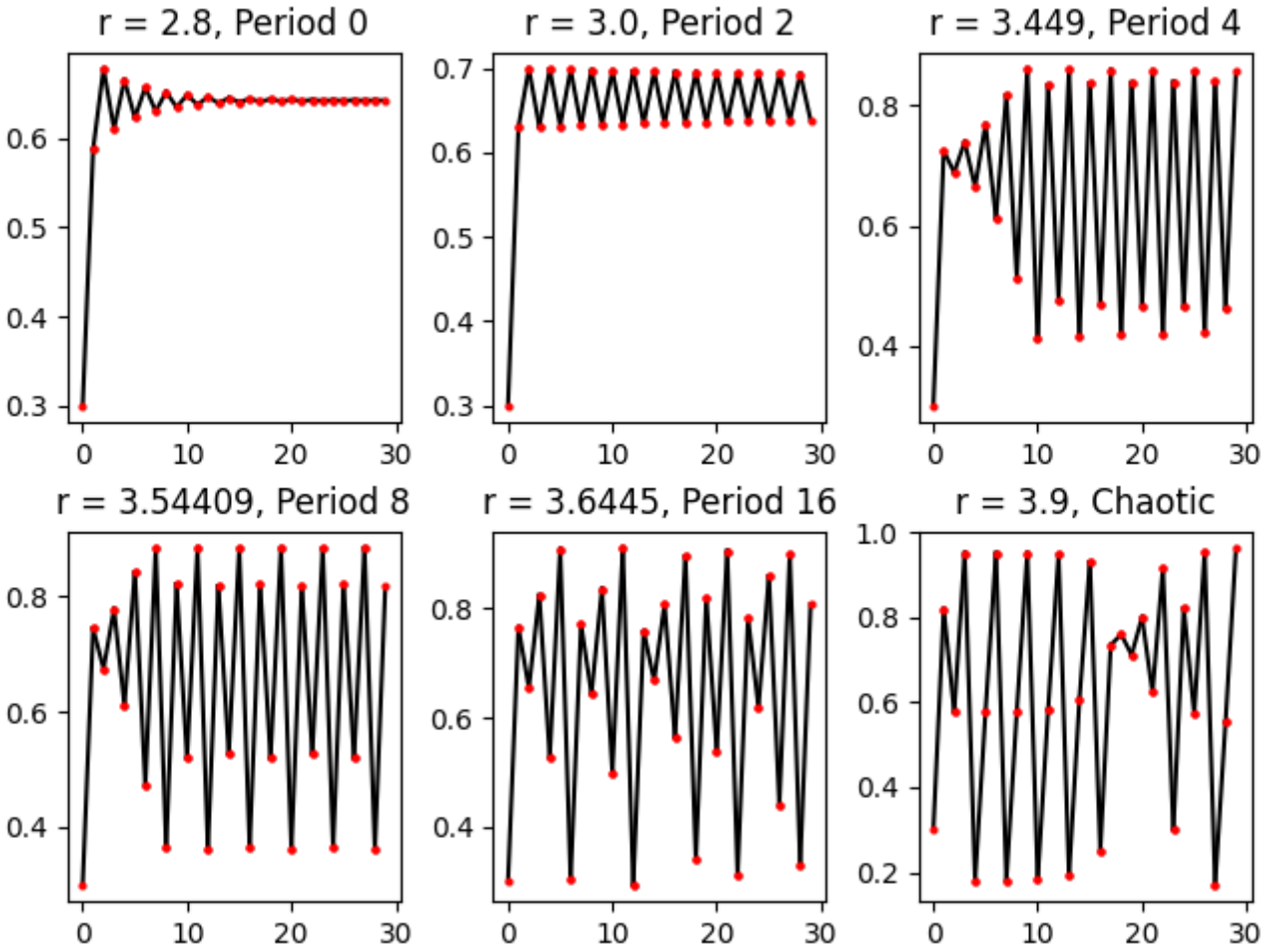
1st bifurcation = 3.000 2nd bifurcation = 3.448 3rd bifurcation = 3.544

Ratio of difference = $\frac{3.448 - 3}{3.544 - 3.448}$ = 0.448/0.095 = 4.666

Therefore, Feigenbaum Constant = 4.666

IMT2020553 Abhinav Mahajan

Question 1

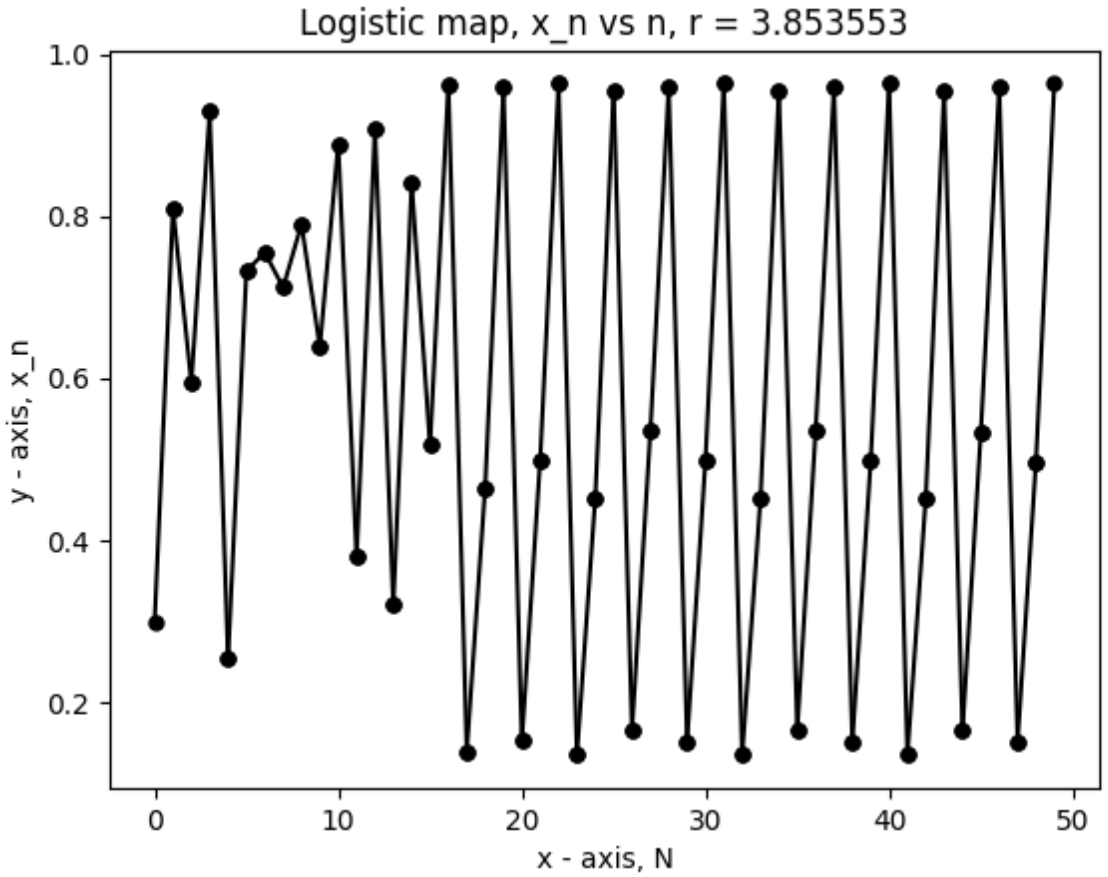


Code for Q1 and Q2 -

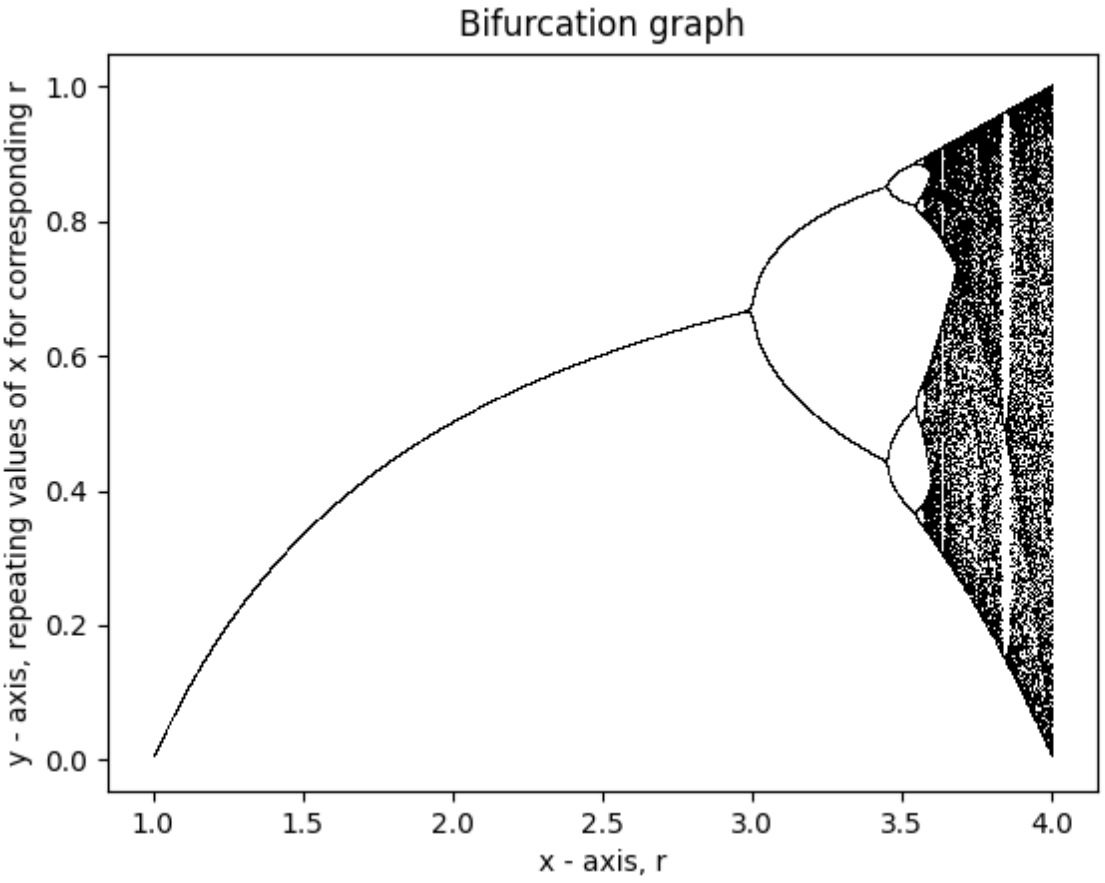
```
from ast import Str

import matplotlib.pyplot as plt


def Plot(r, range_y, plot_name, start_x, subplot_x = 1, subplot_y = 1, subplot_num =
1):

   x = [start_x]

   for i in range(range_y - 1):

       x_1 = x[len(x) - 1]

       x_2 = r * x_1 * (1 - x_1)
```

```
        x.append(x_2)
    n = [i for i in range(range_y)]
    for i in range(range_y - 1):
        x_coordinates = [n[i], n[i+1]]
        y_coordinates = [x[i], x[i+1]]
        plt.subplot(subplot_x, subplot_y, subplot_num)
        plt.plot(x_coordinates, y_coordinates, color = "black")
        x_1 = [x_coordinates[0]]
        y_1 = [y_coordinates[0]]
        plt.subplot(subplot_x, subplot_y, subplot_num)
        plt.plot(x_1, y_1, marker = "o", markersize = 2, markeredgecolor = "red",
markerfacecolor = "red")
        x_1 = [x_coordinates[1]]
        y_1 = [y_coordinates[1]]
        plt.subplot(subplot_x, subplot_y, subplot_num)
        plt.plot(x_1, y_1, marker = "o", markersize = 2, markeredgecolor = "red",
markerfacecolor = "red")
    plot_name = "r = " + str(r) + ", " + str(plot_name)
    plt.title(str(plot_name))
if __name__ == "__main__":
    fig, ax = plt.subplots(2, 3, constrained_layout = True)
    plt.title("Q1.) x_n vs r")
    Plot(2.8, 30, "Period 0", 0.3, 2, 3, 1)
    Plot(3.0, 30, "Period 2", 0.3, 2, 3, 2)
    Plot(3.449, 30, "Period 4", 0.3, 2, 3, 3)
    Plot(3.54409, 30, "Period 8", 0.3, 2, 3, 4)
    Plot(3.6445, 30, "Period 16", 0.3, 2, 3, 5)
    Plot(3.9, 30, "Chaotic", 0.3, 2, 3, 6)
    plt.savefig("Q1")
    plt.show()
```

Question 2 -


Logistic map, x_n vs n, r = 3.853553
y - axis, x_n
x - axis, N
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40
50

Question 3(a) -

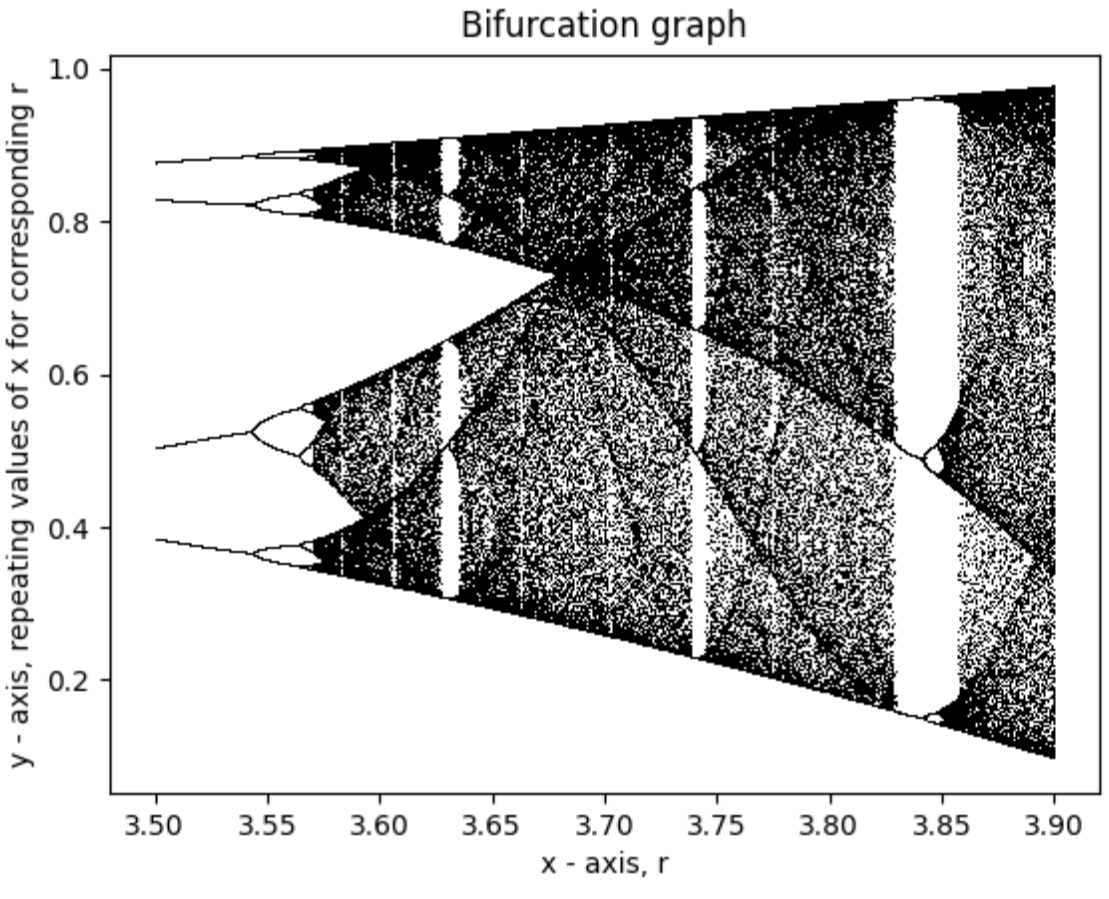


Code for Q 3(a) and 3(b) -

```
import matplotlib.pyplot as plt
import numpy as np


def Plot_Bifurcation(start_x, scale, start_r, end_r):
   #r_list = [i/scale for i in range(int(start_r*scale), int(end_r*scale))]
   r_list = np.linspace(start_r, end_r, scale)
   for r in r_list:
       x = start_x
       for k in range(200):
           x = r * x * (1 - x)
       plt.plot(r, x, ls = '', marker = ',', markeredgecolor = "black", markerfacecolor
= "black")
```

```
        for k in range(16):
            x = r * x * (1 - x)
            plt.plot(r, x, ls = '', marker = ',', markeredgecolor = "black",
markerfacecolor = "black")


    plt.xlabel('x - axis, r')
    plt.ylabel("y - axis, repeating values of x for corresponding r")
    plt.title("Bifurcation graph")
    #plt.savefig("Q3a")
    plt.show()


if __name__ == "__main__":
    Plot_Bifurcation(0.6, 10000, 2.5, 4)
```

Question 3(b) -



Question 4 -

Bifurcation 1: -

x_coordinate = 2.981

Bifurcation 2: -

x_coordinate = 3.443

Bifurcation 3: -

x_coordinate = 3.542

Feigenbaum constant = $\frac{3.443 - 2.981}{3.542 - 3.443}$

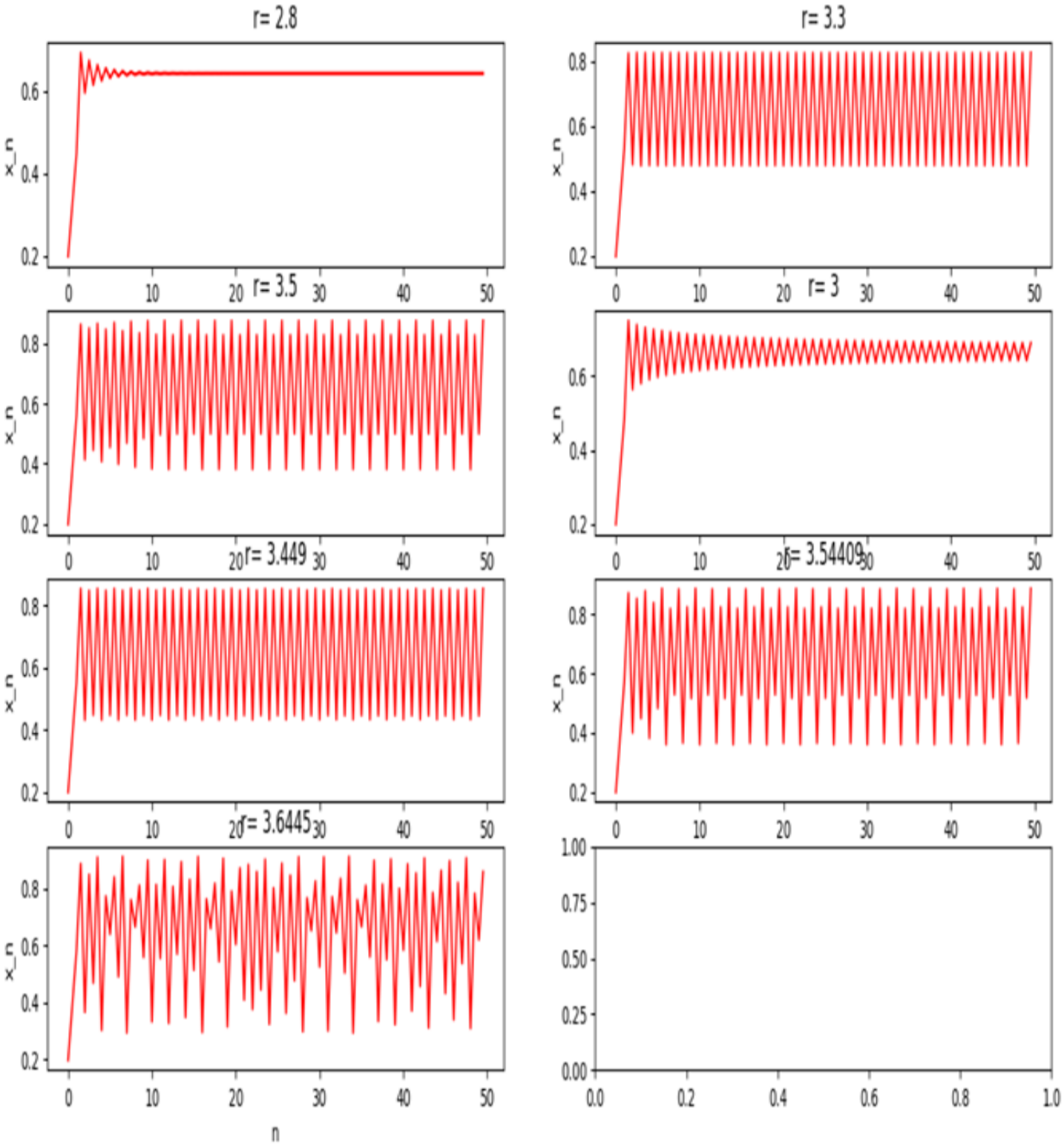
Feigenbaum constant = 4.714

IMT2020539 Shaurya Agrawal

1). CODE

```
import matplotlib.pyplot as plt
import numpy as np
n=50
r=[2.8,3.3,3.5,3,3.449,3.54409,3.6445]
fig ,ax=plt.subplots(4,2)
a=0
b=0
for j in range(0,7):
    y=[];
    t=[];
    y.append(0.2)
    t.append(0)
    i=1
    k=1
    while(i!=n):
        y.append(r[j]*y[k-1]*(1-y[k-1]))
        t.append(i)
        i=i+0.5
        k=k+1
    ax[a][b].plot(t,y,'r')
    ax[a][b].set(xlabel='n',ylabel='x_n')
    ax[a][b].set_title("r= "+str(r[j]))
    if(b!=1):
        b=b+1
    else:
        b=0
        a=a+1
plt.show()
```

PLOT

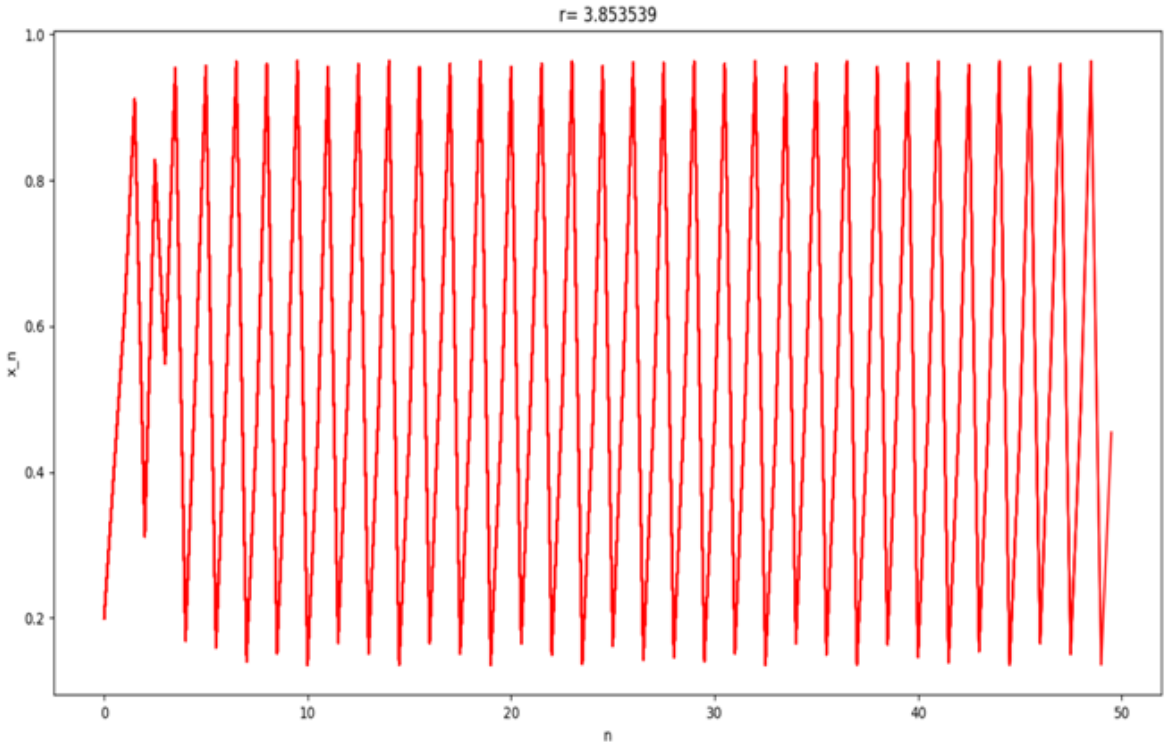


2).

Code

```
import matplotlib.pyplot as plt

import numpy as np

n=50

r=[3.853539]



for j in range(0,1):

    y=[];

    t=[];

    y.append(0.2)

    t.append(0)

    i=1

    k=1

    while(i!=n):

        y.append(r[j]*y[k-1]*(1-y[k-1]))

        t.append(i)

        plt.plot(t,y,'r')

        i=i+0.5

        k=k+1

    plt.xlabel('n')

    plt.ylabel('x_n')

    plt.title("r= "+ str(r[j]))

    plt.show()
```

PLOT



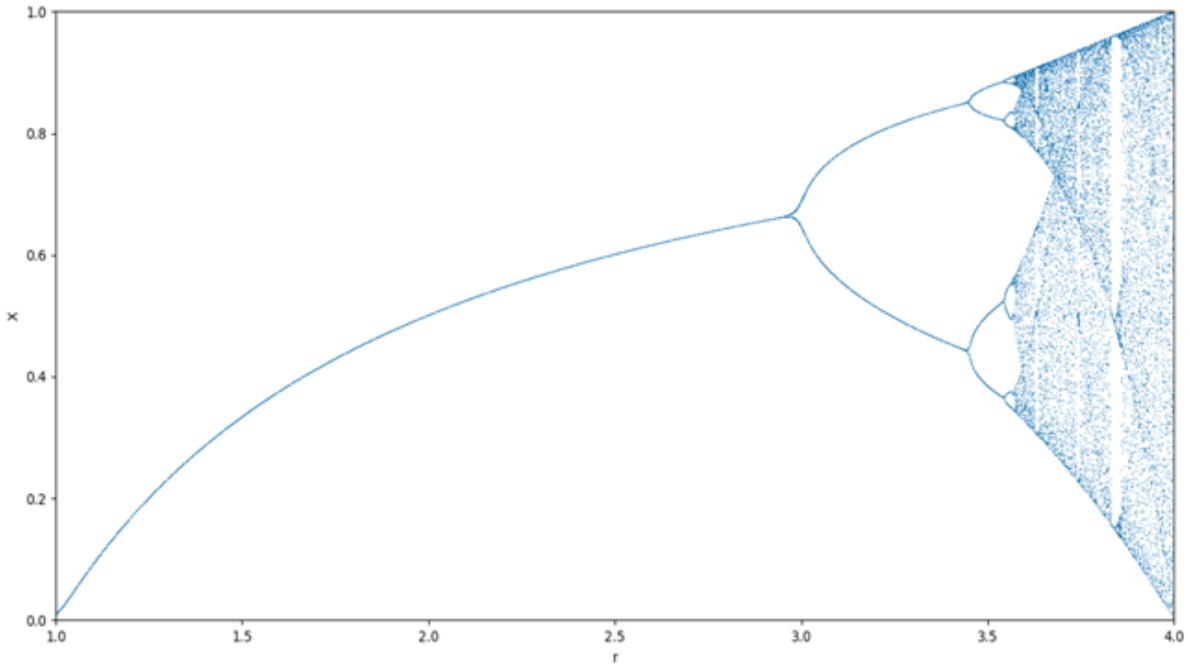
3).

a).

CODE

```
import matplotlib.pyplot as plt

import numpy as np

y=[]

x=[]

r=np.linspace(1,4,10000)

for i in r:

    xc=0.2

    for j in range(111):

        if j>=100:

            x.append(i)

            y.append(xc)

        xc=i*xc*(1-xc)



plt.plot(x,y,ls='',marker=',')

plt.ylim(0, 1)

plt.xlim(1, 4)

plt.xlabel('r')

plt.ylabel('X')

plt.show()
```

PLOT

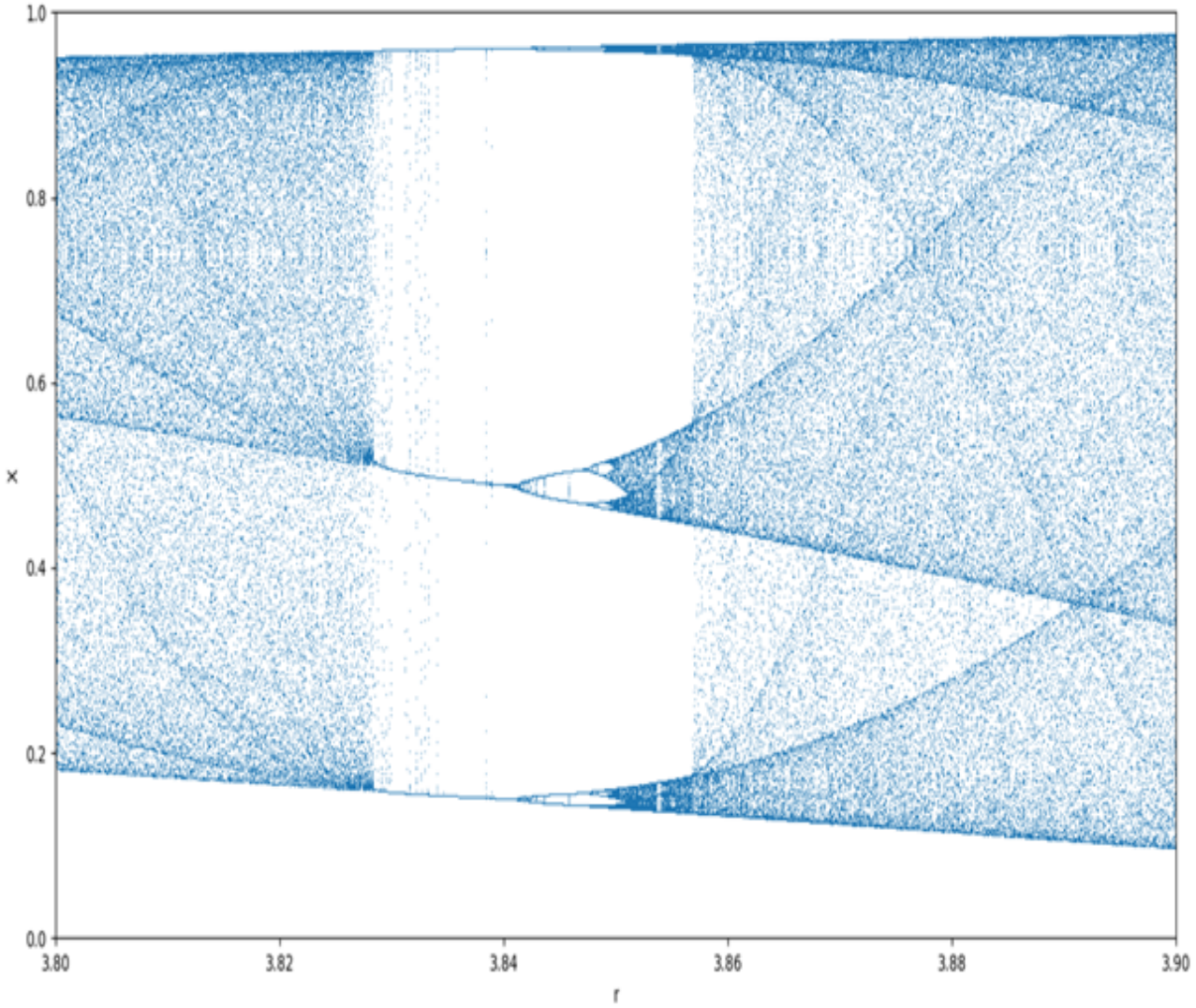


b).

CODE

```
y=[]

x=[]

r=np.linspace(1,4,100000)

for i in r:

    xc=0.2

    for j in range(150):

        if j>=100:

            x.append(i)

            y.append(xc)

        xc=i*xc*(1-xc)



plt.plot(x,y,ls='',marker=',')

plt.ylim(0, 1)

plt.xlim(3.8,3.9)

plt.xlabel('r')

plt.ylabel('X')

plt.show()
```

PLOT



4).(values taken from graph)

1st bifurcation = 3.466 2nd bifurcation = 3.544 3rd bifurcation = 3.565

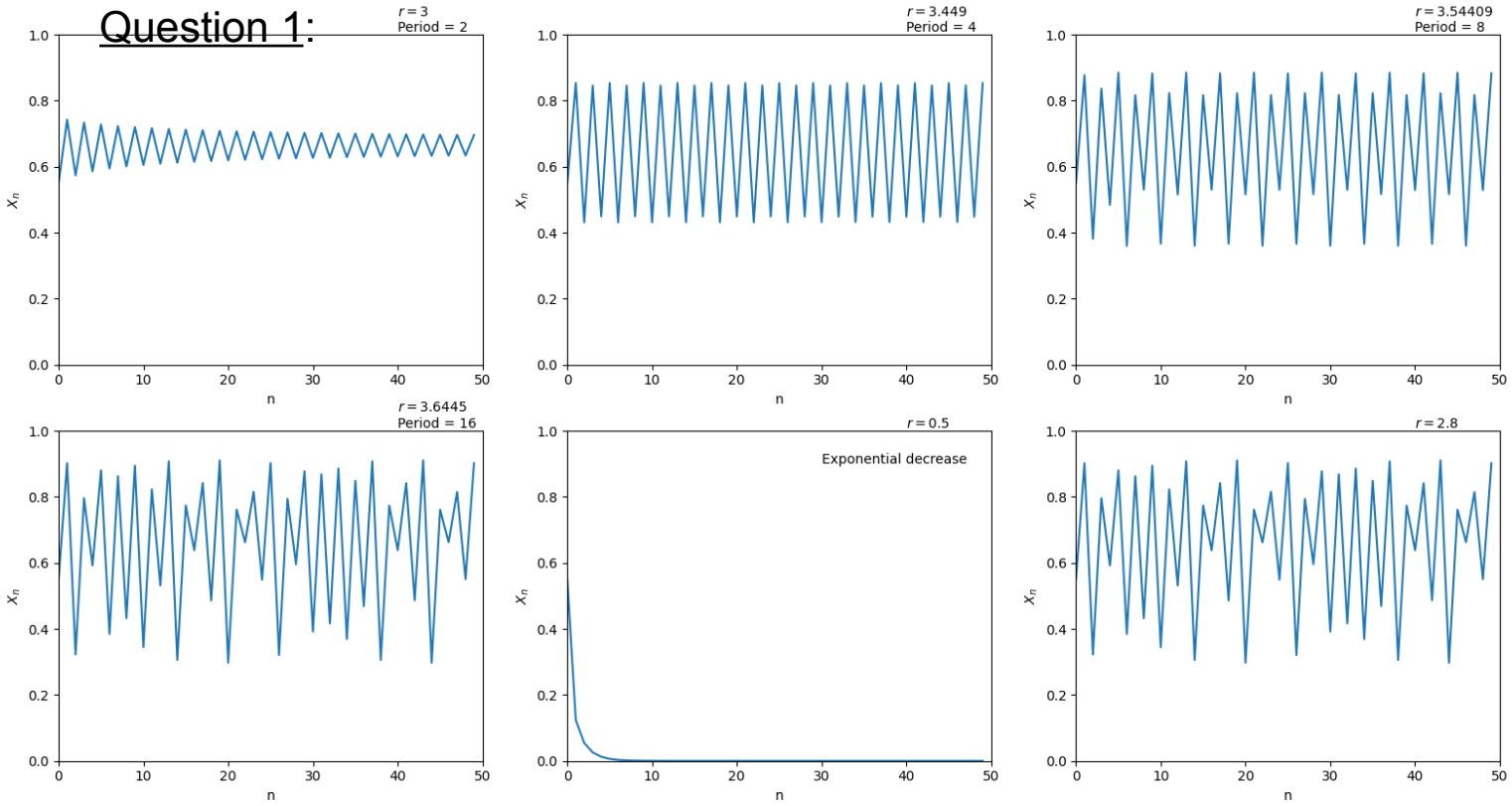
Feigenbaum constant = (3.544-3.446)/ (3.565-3.544)

= 4.66……..

IMT2020085 Harshadeep Donapati

Logistic Map: $X_{n+1} = rX_n(1 - X_n)$

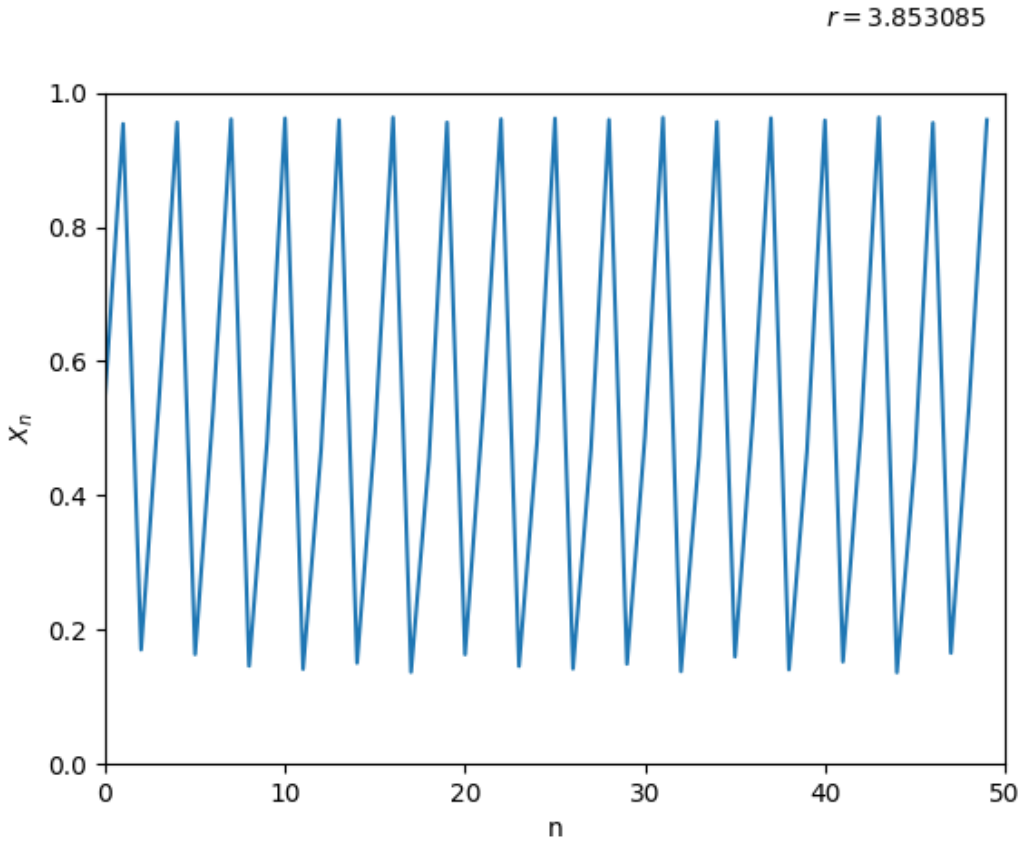
Question 1:



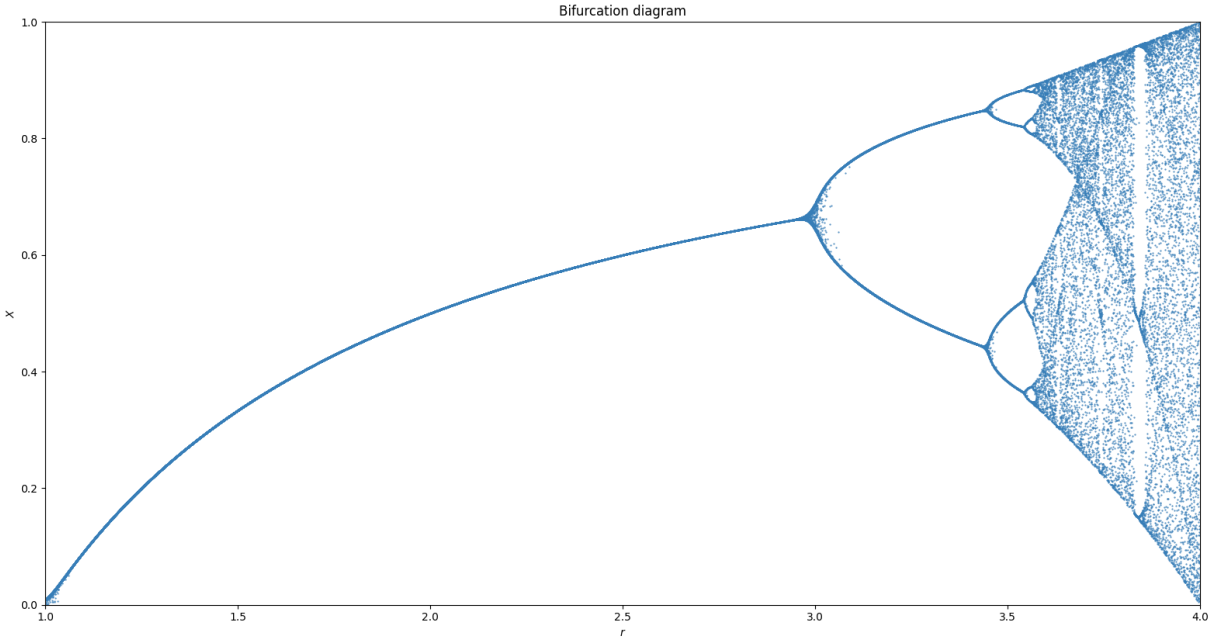
Code for Q1, Q2:

```
import matplotlib.pyplot as plt
import numpy as np


def f(N, r):
      x = np.zeros(N)
      x[0] = 0.55
      for i in range(N - 1):
      x[i + 1] = r * x[i] * (1 - x[i])
      return x

plt.plot(f(50, 3.853085))
plt.axis([0, 50, 0, 1])
plt.xlabel('n')
plt.ylabel(r'$X_{n}$')
plt.text(40, 1.01, r'$r = 3.853085$')
plt.show()
```

Question-2 :



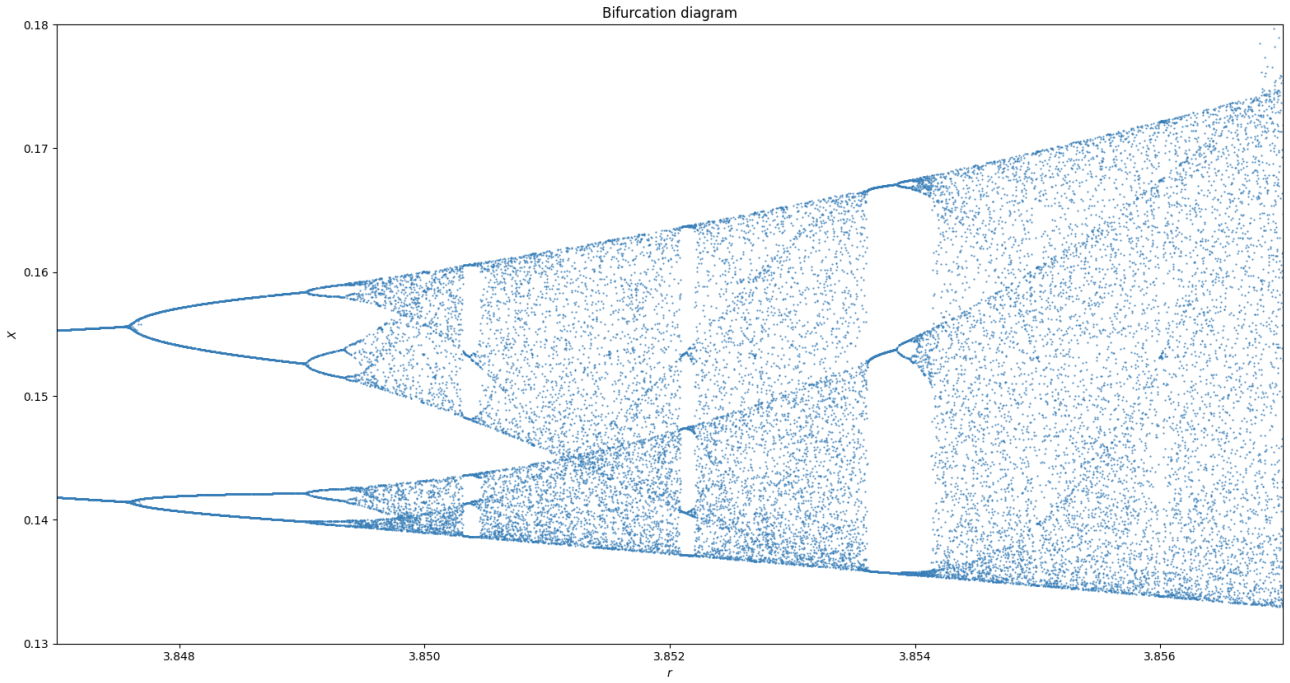
Question 3(a):



Code for Q(3a) and Q(3b) :

```
import matplotlib.pyplot as plt
import numpy as np
import random

Y = []
X = []

rv = np.linspace(1, 4, 100000)


for r in rv:
    X.append(r)

    x = random.uniform(0, 1)

    for i in range(1000):
        x = r * x * (1 - x)

    Y.append(x)

plt.plot(X, Y, '.', c='#377eb8', markersize=1)
plt.title('Bifurcation diagram')
plt.xlabel(r'$r$')
plt.ylabel(r'$X$')
plt.axis([1, 4, 0, 1])
plt.show()
```

Question 3(b):



Around the interval x ≅ 3.854, we have a stable period - 3 cycle.

We can see that from here the plot behaves the same as in the beginning.
∴ This plot shows self similarity.

Question 4:

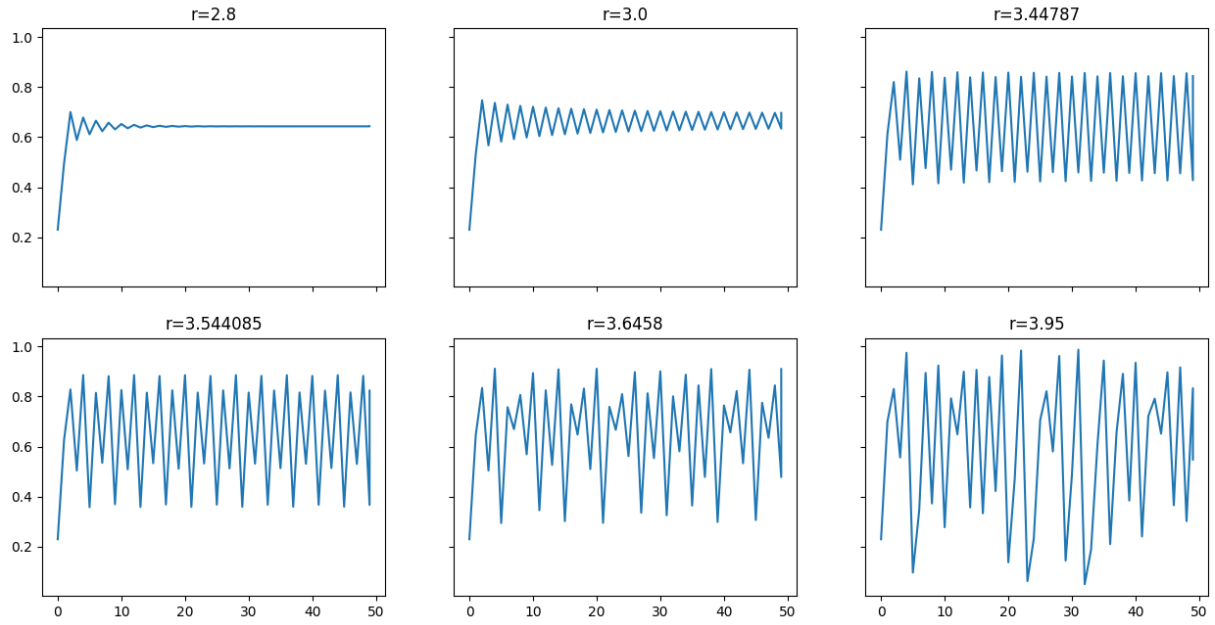
n	Bifurcation parameter (a_n)
1	3
2	3.44859
3	3.54486

Ratio of difference = $\frac{a2 - a1}{a3 - a2} = \frac{3.44859 - 3}{3.54486 - 3.44859} = \frac{0.44859}{0.09627}$

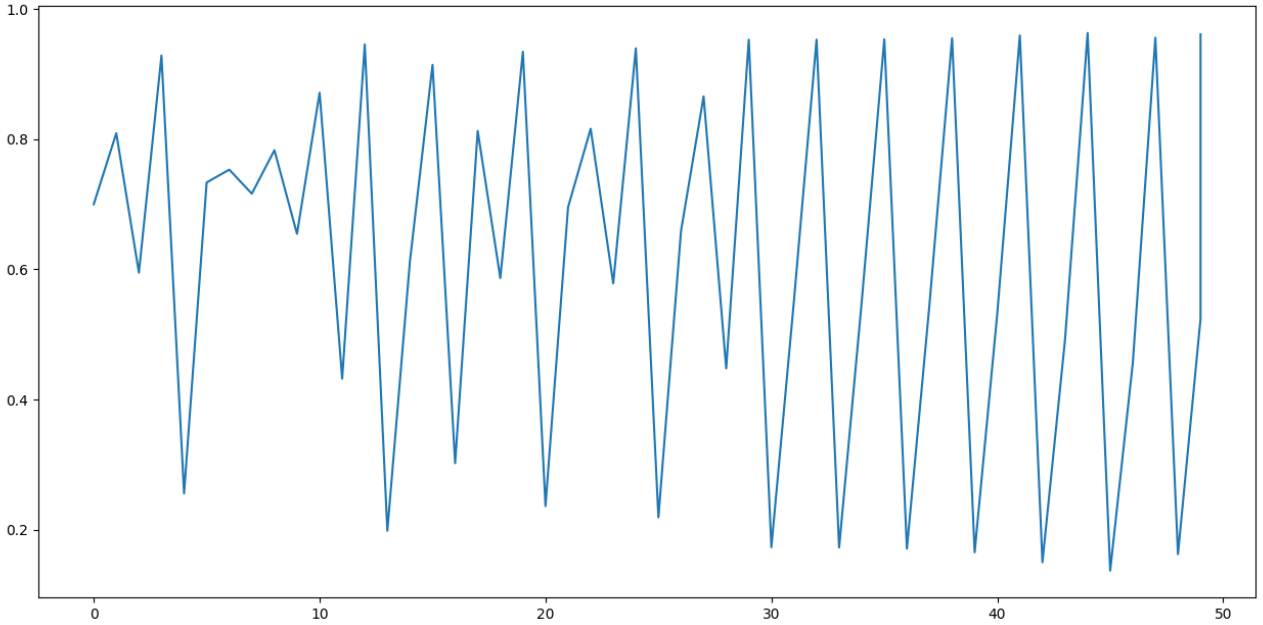
= 4.659707. ∴ Feigenbaum constant = 4.659707.

Ayushmaan Singh (IMT2020126)

Question 1.



Question 2.

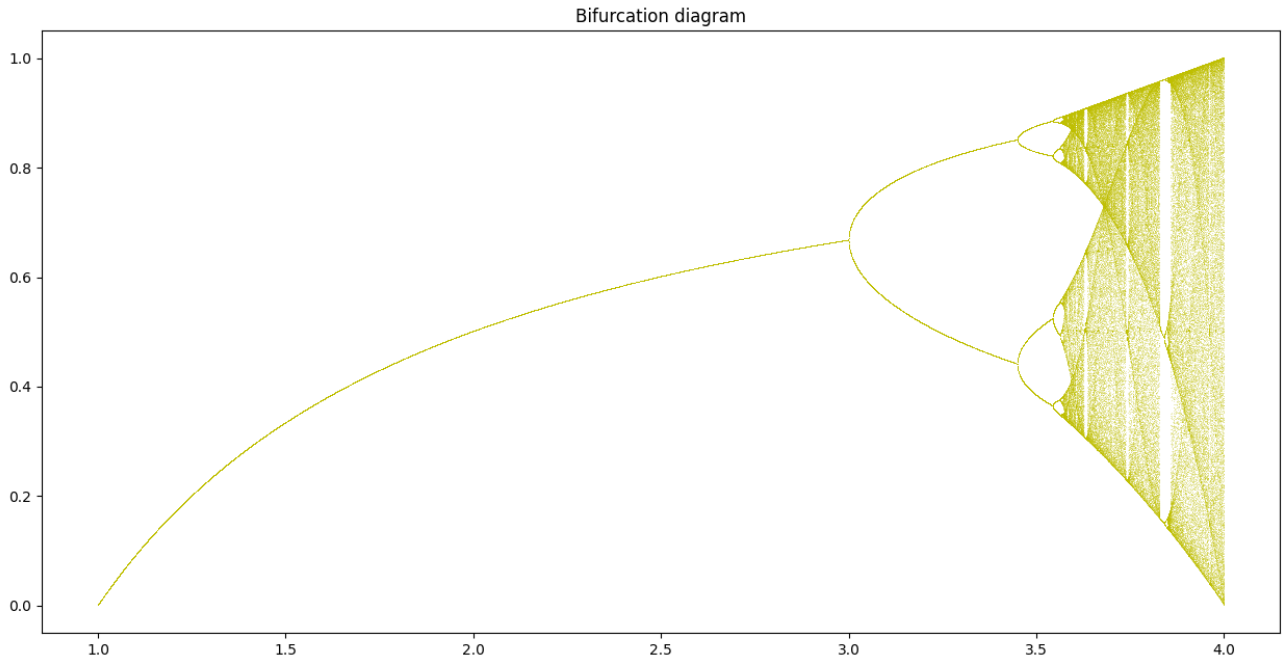


Code for Question 2:

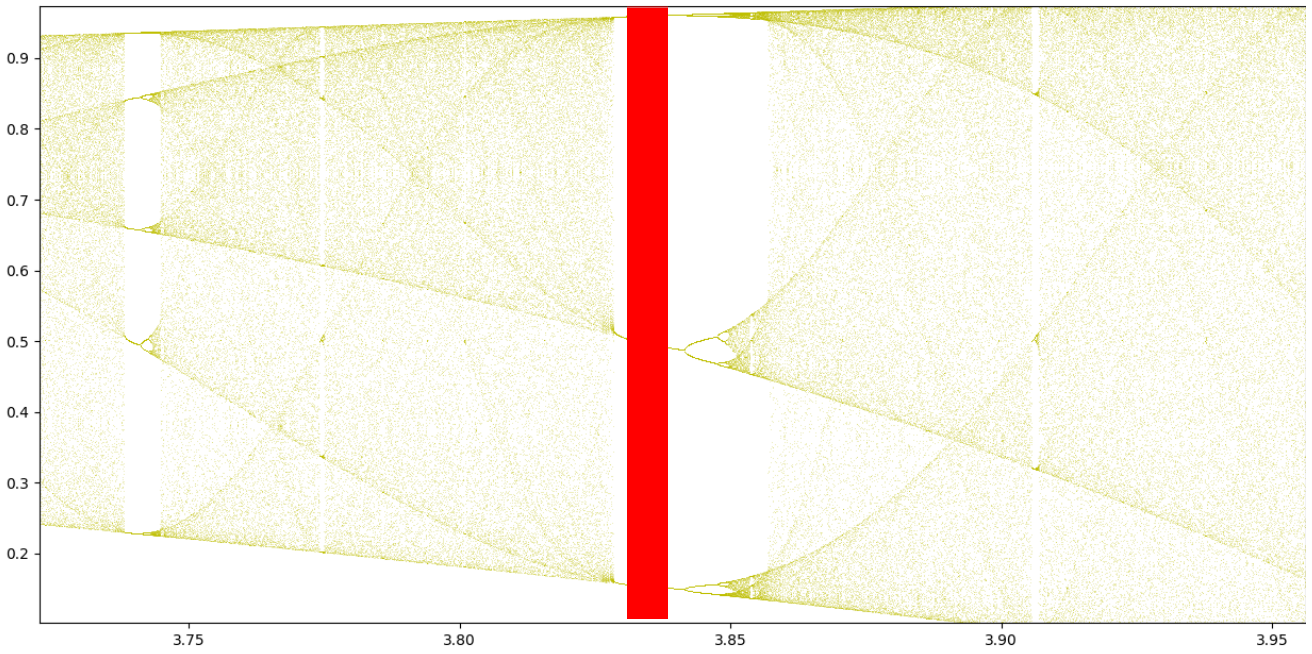
```
import matplotlib.pyplot as plt

def make(r):
    x=[]
    y=[]
    p0=0.7
    y.append(p0)
    for i in range(50):
        pn=(r*p0)*(1-p0)
        p0=pn
        y.append(pn)
        x.append(i)
    x.append(i)
    plt.plot(x,y)

make(3.853126)
plt.show()
```

(Similar code was used for Question 1, but with minor changes for subplots and different values of r)

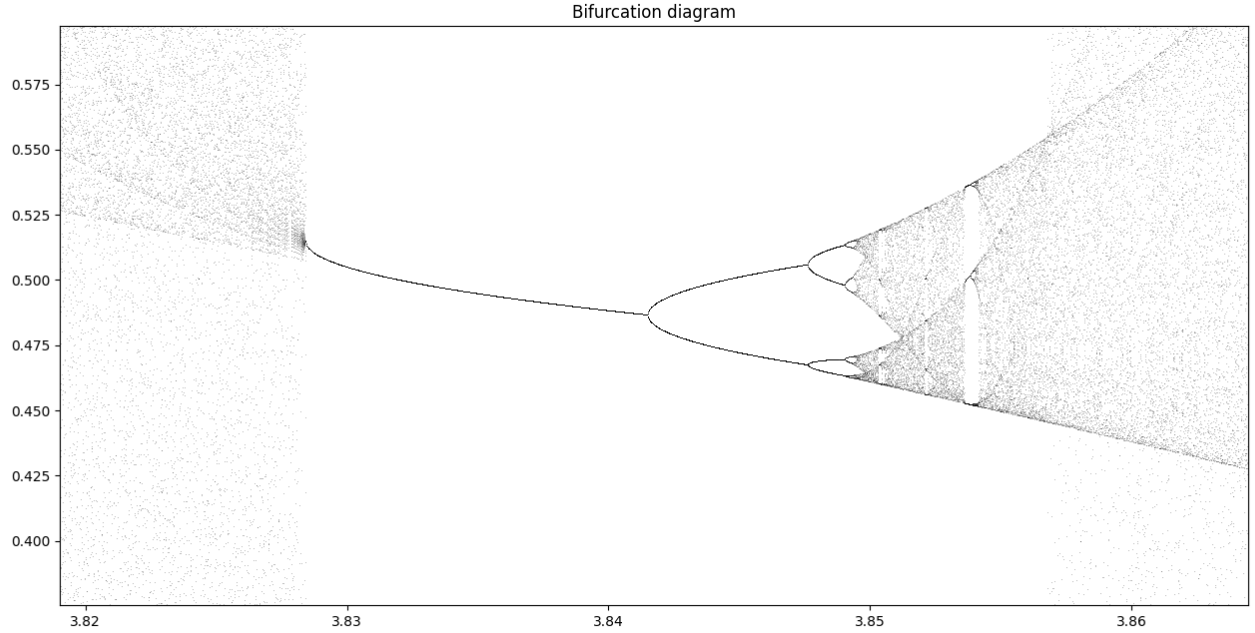
Question 3.(a)



Question 3.(b)



The values of r that lie in the red strip of the plot are the values that will give a stable 3-cycle period. A close-up of one of the three branches is shown below:



It can be observed that this section of the plot shows similarity to the original plot. Hence this bifurcation graph shows signs of self-similarity.

Code for Question 3.

```
import matplotlib.pyplot as plt
import numpy as np

cycles = 10000
r=np.linspace(1,4,10000)
x=0.23
for i in range(cycles):
    x = r*x*(1-x)
    if i >= (cycles - 100):
        plt.plot(r, x, ',k', alpha=.25)
plt.title('Bifurcation diagram')
plt.show()
```

Question 4.

Values of r at which graph bifurcates
3.000000
3.448244
3.543654
3.564237
3.568664

For Feigenbaum's constant, we can calculate as follows:

$$\frac{(n+1)^{th}\ bifurcation\ value - n^{th}\ bifurcation\ value}{(n+2)^{th}\ bifurcation\ value - (n+1)^{th}\ bifurcation\ value}$$

Therefore, using the values observed, we have:

$$F.C = \frac{(3.448244 - 3.000000)}{(3.543654 - 3.448244)} = \frac{0.448244}{0.09541} = 4.698040$$

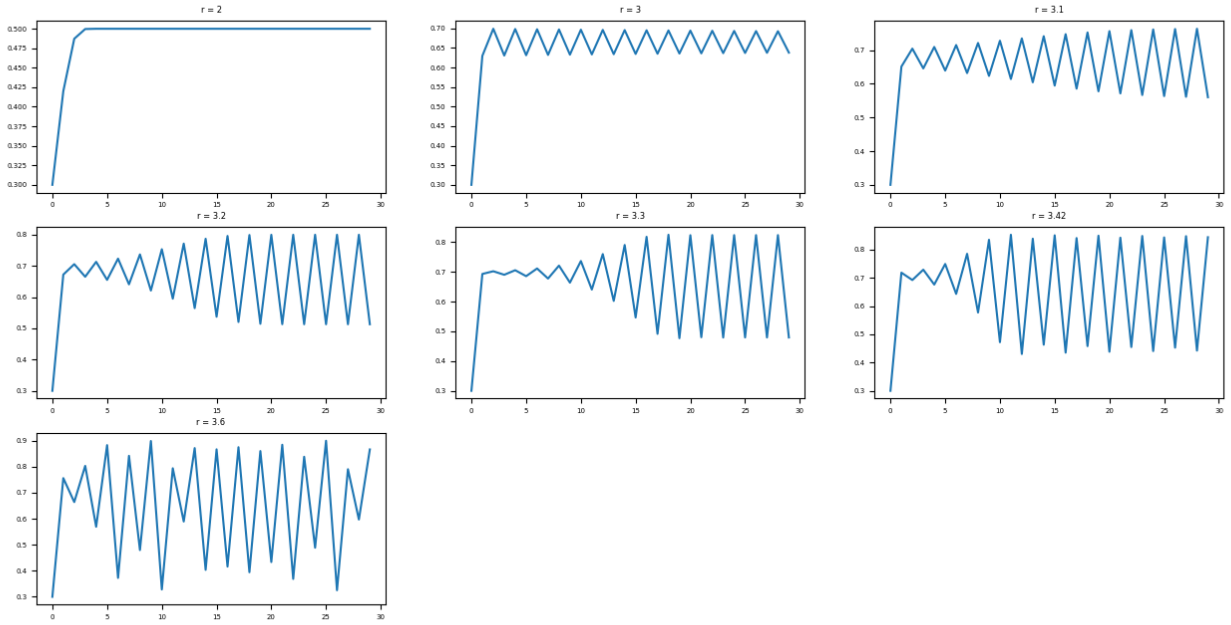
$$F.C = \frac{(3.543654 - 3.448244)}{(3.564237 - 3.543654)} = \frac{0.09541}{0.020583} = 4.635379$$

$$F.C = \frac{(3.564237 - 3.543654)}{(3.568664 - 3.564237)} = \frac{0.020583}{0.004427} = 4.649424$$

$$Average\ Feigenbaum's\ constant = \frac{(4.698040 + 4.635379 + 4.649424)}{3} = 4.660948$$

Vishnutha Sheela (IMT2020057)

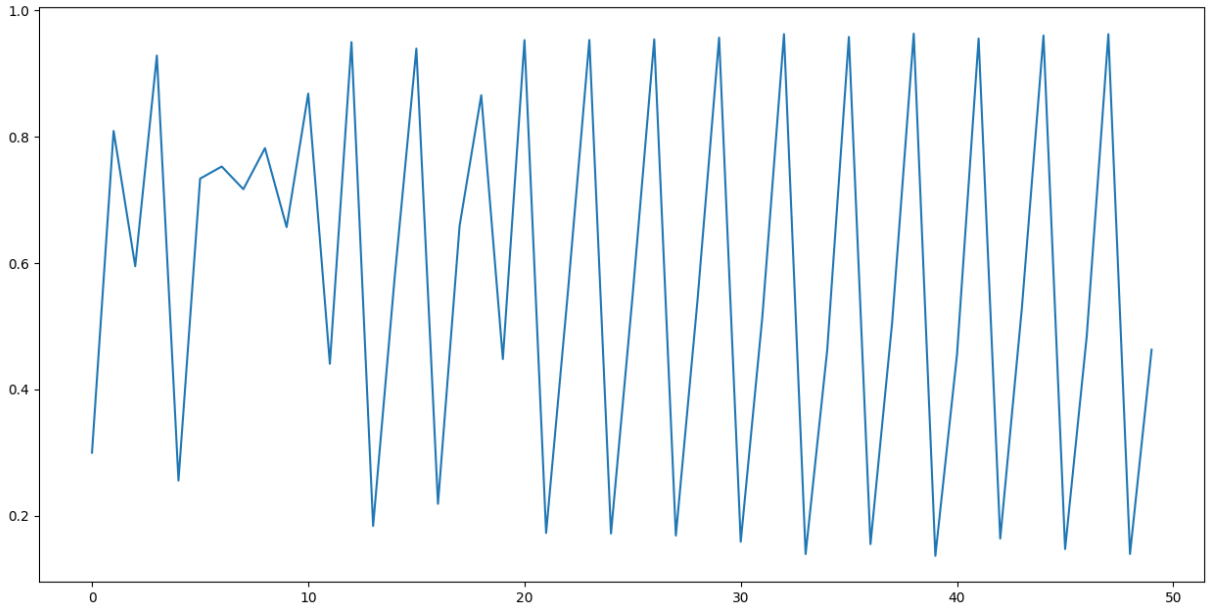
Question 1.



Code for Question 1:

```
from matplotlib import pyplot as plt
from math import *
x = range(0,30)
r = [2,3,3.1,3.2,3.3,3.42,3.6]
k = 0
plt.rcParams['font.size'] ='5'
for j in r :
    y = [0.3]
    for i in x[:29]:
        y.append( j*(y[i]*(1-y[i])))
    k+=1
    plt.subplot(3,3,k)
    plt.gca().set_title("r = "+str(j))
    plt.plot(x,y)
plt.show()
```

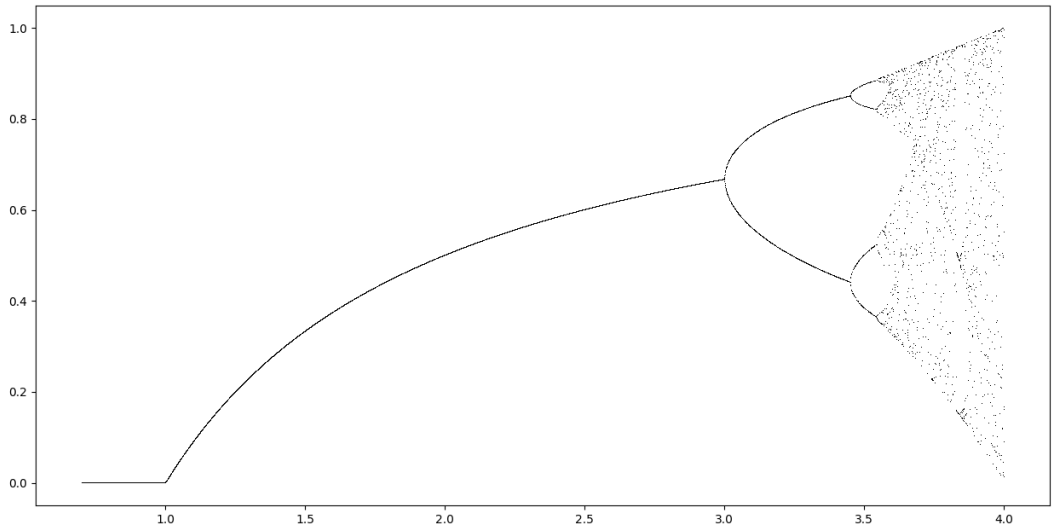
Question 2.



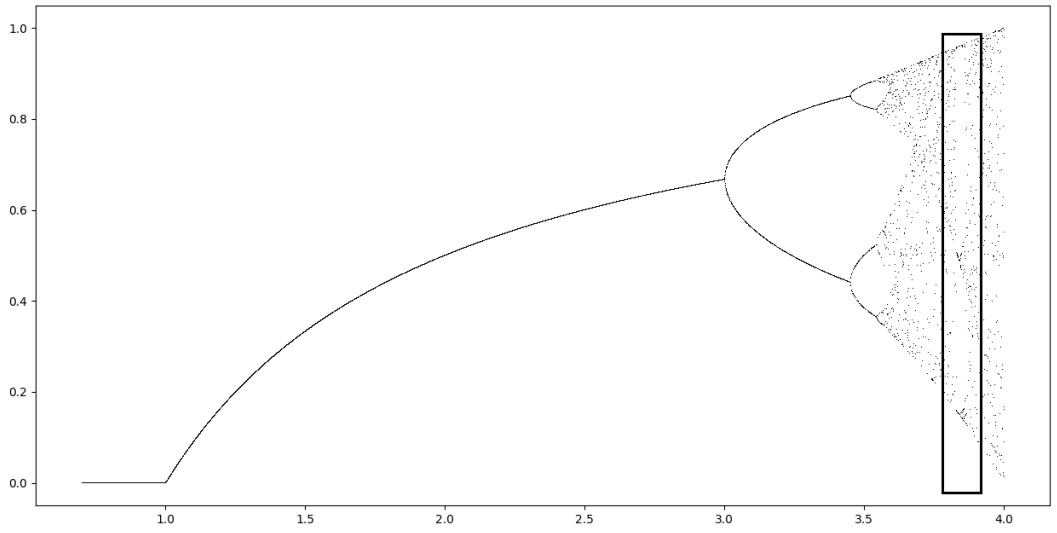
Code for Question 2:

```
from matplotlib import pyplot as plt
from math import *
x = range(0,50)
y = [0.3]
r = 3.853057
for i in x[:49]:
    y.append( r*(y[i]*(1-y[i])))
plt.plot(x,y)
plt.show()
```

Question 3.(a)



Question 3.(b)



It can be observed that the section of the plot in the black box shows similarity to the original plot. Hence this bifurcation graph shows signs of self-similarity.

Code for Question 3.

```
import matplotlib.pyplot as plt
import numpy as np
P=np.linspace(0.7,4,10000)
m=0.7
X = []
Y = []
for r in P:
    X.append(r)
    m = np.random.random()
    for n in range(1001):
      m=(r*m)*(1-m)
    for l in range(1051):
      m=(r*m)*(1-m)
    Y.append(m)
plt.plot(X, Y, ls='', marker=',',color = 'black')
plt.show()
```

Question 4.

Values of r at which graph bifurcates
3.000
3.453
3.548

For Feigenbaum's constant, we can calculate as follows:

$$\frac{(n+1)^{th}\ bifurcation\ value - n^{th}\ bifurcation\ value}{(n+2)^{th}\ bifurcation\ value - (n+1)^{th}\ bifurcation\ value}$$

Therefore, using the values observed, we have:

$$F.C = \frac{(3.453 - 3.000)}{(3.548 - 3.453)} = \frac{0.451}{0.096} = 4.697917$$