Assignment 3

`Course: MM2090`
`Instructor: Prof.Gandham Phanikumar`
`Roll no.: MM20B020`
`Name: Gokul C`
`Batch(Group No.): 5`
`Date of submission: 15-06-2021`

Q1. Consider the following example of a time table with th....
→
(a)
Link for handmade table pdf

	8	9	10	11	12	13	14	15	16	17
Mon		Slot-1	Slot-1				Slot-2	Slot-2		
Tue			Slot-3	Slot-3						
Wed					Slot-4	Slot-4				
Thur	Slot-5	Slot-5							Slot-6	Slot-6
Fri					Slot-7	Slot-7	Slot-8	Slot-8		

(b)

table.txt

```
Timetable is-a table
Timetable has-row Mon
Timetable has-row Tue
Timetable has-row Wed
Timetable has-row Thur
Timetable has-row Fri
Mon has-slot Slot-1
Mon has-slot Slot-2
Tue has-slot Slot-3
```

```
Wed has-slot Slot-4
Thur has-slot Slot-5
Thur has-slot Slot-6
Fri has-slot Slot-7
Fri has-slot Slot-8
Slot-1 starts-at 9:00 Hrs
Slot-1 ends-at 10:59 Hrs
Slot-2 starts-at 14:00 Hrs
Slot-2 ends-at 15:59 Hrs
Slot-3 starts-at 10:00 Hrs
Slot-3 ends-at 11:59 Hrs
Slot-4 starts-at 12:00 Hrs
Slot-4 ends-at 13:59 Hrs
Slot-5 starts-at 8:00 Hrs
Slot-5 ends-at 9:59 Hrs
Slot-6 starts-at 16:00 Hrs
Slot-6 ends-at 17:59 Hrs
Slot-7 starts-at 12:00 Hrs
Slot-7 ends-at 13:59 Hrs
Slot-8 starts-at 14:00 Hrs
Slot-8 ends-at 15:59 Hrs
```

(c)

> q1.py

```python
#!/usr/bin/python3
row_lines=[]
slot_lines=[]
position_lines=[]
filename = "table.txt"
with open(filename) as f:
    line=f.readline()
    while(line):
        word=line.split()
        if(word[1]=='is-a'):
            word=line.split()
            table_name=word[0]
        elif(word[1]=='has-row'):
            row_lines.append(line)
        elif(word[1]=='has-slot'):
            slot_lines.append(line)
        elif(word[1]=='starts-at' or word[1]=='ends-at'):
            position_lines.append(line)
        line = f.readline()

row_names=["8","9","10","11","12","13","14","15","16","17"]
column_names=[]
slot_list=[]

for line in row_lines:
```

```
    word=line.split()
    column_names.append(word[2])

def column_num(column_name):
    for i in range(0,len(column_names)):
        if(column_names[i]==column_name):
            break
    return i

for line in slot_lines:
    slot=[]
    rows=[]
    word=line.split()
    slot.append(word[2])
    slot.append(column_num(word[0]))
    for i in range(0,len(position_lines)):
        word_2=position_lines[i].split()
        word_3=word_2[2].split(':')
        if(word_2[0]==word[2] and word_2[1]=='starts-at'):
            rows.append(int(word_3[0])-8)
        elif(word_2[0]==word[2] and word_2[1]=='ends-at'):
            rows.append(int(word_3[0])-8)
    slot.append(rows)
    slot_list.append(slot)

cells=[['' for c in range(len(row_names))] for r in range(len(column_names))]

for slot in slot_list:
    for j in range(slot[2][0],slot[2][1]+1):
        cells[slot[1]][j]=slot[0]

import matplotlib.pyplot as plt

table = plt.table(cellText = cells,
                  rowLabels = column_names,
                  colLabels = row_names,
                  loc='center')

plt.axis('off')
plt.grid('off')

plt.savefig('q1-{}.jpg'.format(table_name))

print("Table saved as","q1-{}.jpg".format(table_name),"in the current directory")
mergeabl=[]
notmergeabl=[]

for i in range(0,len(cells)-1):
    for j in range(i+1,len(cells)):
        clash=0
        for a in range(0,len(cells[0])):
            if(cells[i][a] and cells[j][a]):
                clash=clash+1
```

```
        if(clash==0):
            mergeabl.append([column_names[i],column_names[j]])
        else:
            notmergeabl.append([column_names[i],column_names[j]])
print("Mergeable: ",mergeabl)
print("Not Mergeable: ",notmergeabl)
```

	8	9	10	11	12	13	14	15	16	17
Mon		Slot-1	Slot-1				Slot-2	Slot-2		
Tue			Slot-3	Slot-3						
Wed					Slot-4	Slot-4				
Thur	Slot-5	Slot-5							Slot-6	Slot-6
Fri					Slot-7	Slot-7	Slot-8	Slot-8		

(d)

> Output

```
Table saved as q1-Timetable.jpg in the current directory
Mergeable:  [['Mon', 'Wed'], ['Tue', 'Wed'], ['Tue', 'Thur'], ['Tue', 'Fri'], ['Wed', 'Thur'], ['Thur', 'Fri']]
Not Mergeable:  [['Mon', 'Tue'], ['Mon', 'Thur'], ['Mon', 'Fri'], ['Wed', 'Fri']]
```

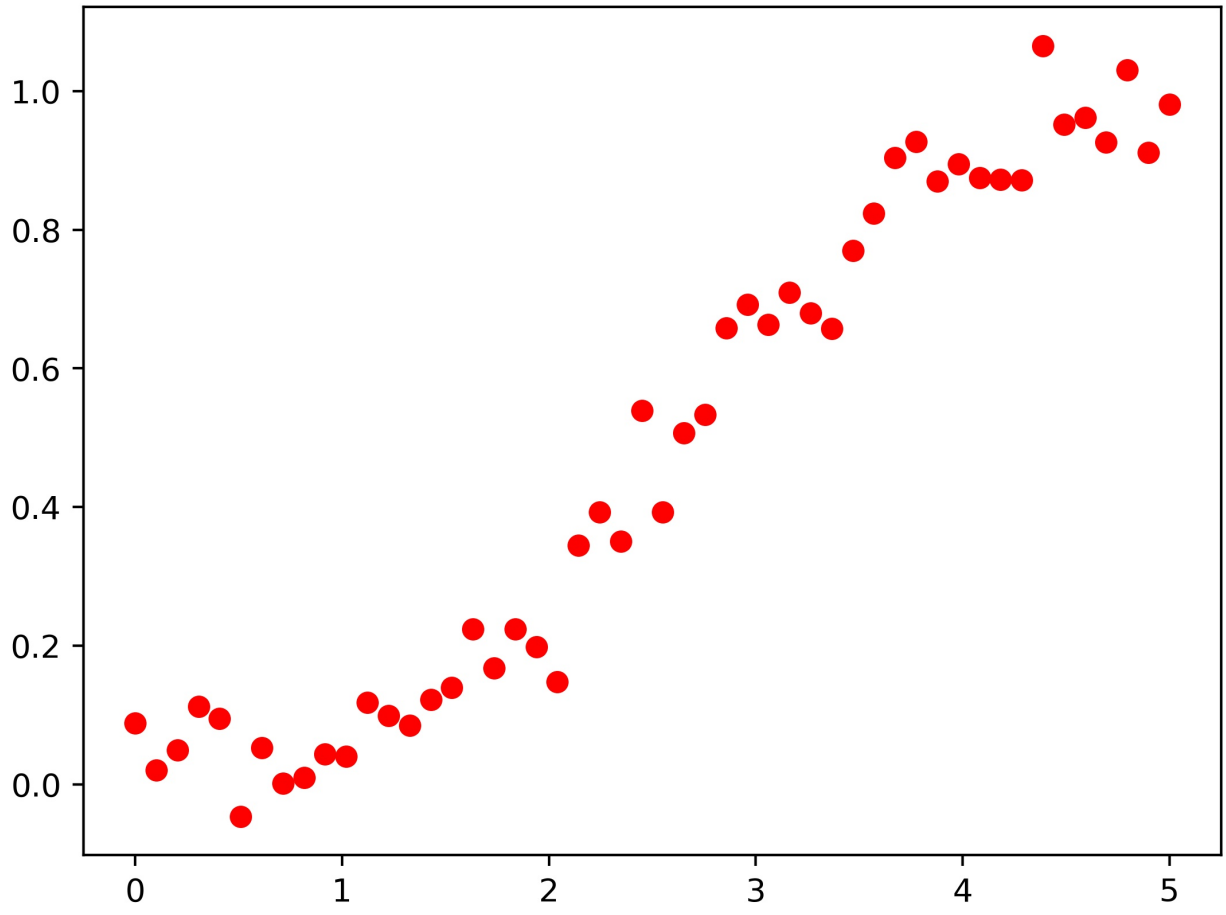
```
gokul@gokul-Inspiron-3250:~/Documents/mm2090_assignment/assignment3$ ./q1.py
Table saved as q1-Timetable.jpg in the current directory
Mergeable:  [['Mon', 'Wed'], ['Tue', 'Wed'], ['Tue', 'Thur'], ['Tue', 'Fri'], ['Wed', 'Thur'], ['Thur', 'Fri']]
Not Mergeable:  [['Mon', 'Tue'], ['Mon', 'Thur'], ['Mon', 'Fri'], ['Wed', 'Fri']]
gokul@gokul-Inspiron-3250:~/Documents/mm2090_assignment/assignment3$
```

Q2. There are a number of natural phenomena that can be describ.....

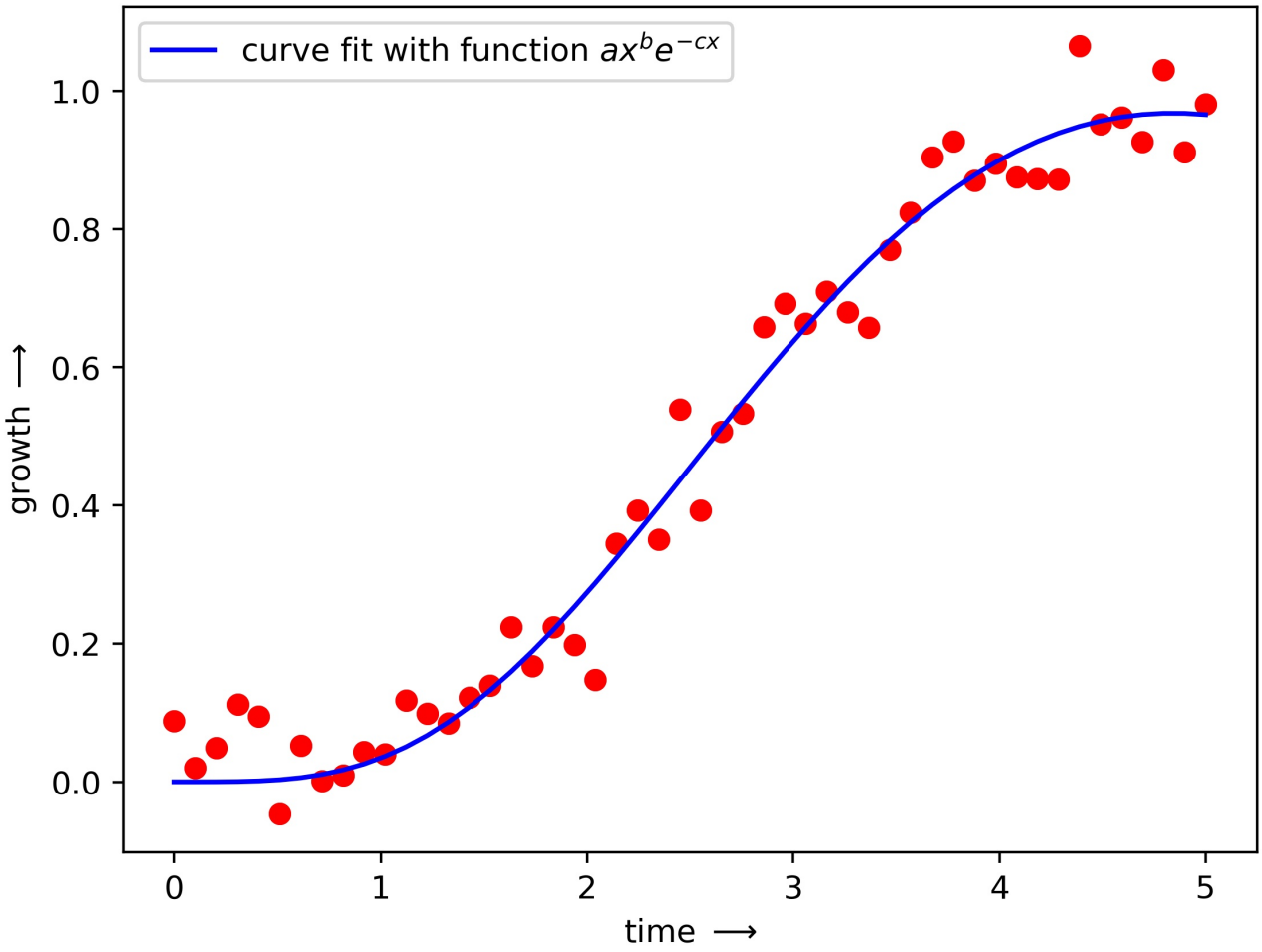
> Code

```
#!/usr/bin/python3
import numpy as np
import math
x=np.linspace(0,5,num=50)
np.random.seed(0)
y=0.075*np.power(x,4.4)*np.power(np.e,-0.9*x)+0.05*np.random.normal(size=50)

import matplotlib.pyplot as plt
plt.scatter(x,y,color='r')
plt.savefig("q2_scatterplt.jpg",dpi=400)
print("Scatter plot is saved as q2_scatterplt.jpg in current directory")

from scipy import optimize
def func(x,a,b,c):
    return a*np.power(x,b)*np.power(np.e,-1*c*x)

par, par_var=optimize.curve_fit(func,x,y,p0=[0.1,2,1])

plt.plot(x,func(x,par[0],par[1],par[2]),'b-',label="curve fit with function $ax^be^{-cx}$")
plt.scatter(x,y,color='r')
plt.legend(loc='best')
plt.xlabel('time $\\longrightarrow$')
plt.ylabel('growth $\\longrightarrow$')
plt.savefig("q2_curvefitplt.jpg",dpi=400)
print("Curve-fit plot is saved as q2_curvefitplt.jpg in current directory")

max_index=0
max_slope=0
for i in range(49):
    if((func(x[i+1],par[0],par[1],par[2])-func(x[i],par[0],par[1],par[2]))/(x[i+1]-x[i])>max_slope):
        max_slope=(func(x[i+1],par[0],par[1],par[2])-func(x[i],par[0],par[1],par[2]))/(x[i+1]-x[i])
        max_index=i
print("Maximum growth @ x=",x[max_index],"at the rate of",max_slope)
```

(a)
Scatter plot



(b)
Curve fit function




```
gokul@gokul-Inspiron-3250:~/Documents/mm2090_assignment/assignment3$ ./q2.py
Scatter plot is saved as q2_scatterplt.jpg in current directory
./q2.py:15: RuntimeWarning: divide by zero encountered in power
  return a*np.power(x,b)*np.power(np.e,-1*c*x)
Curve-fit plot is saved as q2_curvefitplt.jpg in current directory
Maximum growth @ x= 2.4489795918367347 at the rate of 0.37442848855681454
gokul@gokul-Inspiron-3250:~/Documents/mm2090_assignment/assignment3$
```

Model used: ax^be^{-cx}

Approach to find maximum rate: I Calculated $\frac{func(x_2)-func(x_1)}{x_2-x_1}$ for every point and found the maximum of all values, which is maximum rate of growth.

> Note: $func$ is a function defined in above code, x_i is time.
