18ECE201J – PYTHON AND SCIENTIFIC PYTHON

Laboratory Manual (2020-2021- ODD)

College of Engineering & Technology
SRM Institute of Science and Technology
Kattankulathur – 603 203,
Chengalpattu District, Tamilnadu.

Vision of the Department

1. To create and disseminate knowledge in the area of Electronics and Communication Engineering through national and international accredited educational process.
2. To facilitate a unique learning and research experience to the students and faculty.
3. To prepare the students as highly ethical and competent professionals.

Mission of the Department

1. Build an educational process that is well suited to local needs as well as satisfies the national and international accreditation requirements.
2. Attract the qualified professionals and retain them by building an environment that fosters work freedom and empowerment.
3. With the right talent pool, create knowledge and disseminate, get involved in collaborative research with reputed universities and produce competent graduands.

Program Educational Objectives (PEO) for the Electronics and Communication Engineering program describes accomplishments that graduates are expected to attain within a few years of graduation.

Graduates will be able to:

PEO1: Establish themselves as successful and creative practicing professional engineers both nationally and globally in the related fields of Electronics and Communication Engineering.

PEO2: Apply the acquired knowledge and skills in solving real-world engineering problems; develop novel technology and design products, which are economically feasible and socially relevant.

PEO3: Develop an attitude of lifelong learning for sustained career advancement and adapt to the changing multidisciplinary profession.

PEO4: Demonstrate leadership qualities, effective communication skills, and to work in a team of enterprising people in a multidisciplinary and multicultural environment with strong adherence to professional ethics.

18ECE201J- Python & Scientific Python

Exp. No.	Title of the Experiment	Contact Hrs
1.	Programming on Formula and standard mathematical function	2
2.	Program on making modules	2
3.	Programming on lists and loops	2
4.	Curve Plotting	2
5.	Animating a Function	2
6.	Sound generated by formula and Difference Equation	2
7.	Reading the student marks file into a dictionary data with student name as key and computing the average grades	2
8.	Reading the web temperature text file into dictionary and computing average temperature	2
9.	Programming on class	2
10.	Real card game: random number generation	2
11.	Sample Game: Guessing a Number and Rolling Two Dice	2
12.	Random Walk in one Dimension Space	2
13.	Numpy File Reading and Data Analysis	2
14.	Numpy File Reading and Data Analysis	2
15.	Numpy signal processing- Blurring an image with a Gaussian filter	2
	TOTAL CONTACT HRS	**30**

1. Lab Report Format

- Single side printout should be taken
- Prelab and Postlab answers should be written in the printout itself (Students should leave enough space for writing the answers and then take printout).

Laboratory Report Cover Sheet

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
Faculty of Engineering and Technology
Department of Electronics and Communication Engineering

18ECE3201J Python and Scientific Python

V Semester, 2020-21 (ODDSemester)

Name :

Register No. :

Venue :

Title of Experiment :

Date of Conduction :

Date of Submission :

Particulars	Max. Marks	Marks Obtained
Pre Lab	05	
Design, HDL Code	15	
Post Lab	5	
Output verification &viva	10	
Lab Report	05	
Total	**40**	

REPORT VERIFICATION

Staff Name :

Signature :

1. Programming on Formula and Standard Mathematical Function

1.1 Aims

- To use Python as a calculator
- To Evaluate the trajectory of the Ball Kicked/thrown
- To determine the air resistance of the football usingmathematical function.

1.2 Background

In this experiment, we have to use python to:

Solve a simple mathematical equation.

Find the trajectory of a ball that is thrown or kicked with an initial velocity vo making an angle θ

with the horizontal using the formula, $y = x \tan\theta -$ gx2

o

$2v^2 \cos^2\theta$

y = refers to the vertical position of the object inmeters
x = refers to the horizontal position of the object in meter

,where,

vo= refers to the initial velocity of the object combined withmeter per second g= refers to the acceleration due to gravity that is 9.81 m/s2
θ– refers to the initial angle from the horizontal plane in degrees or radians.

Evaluate the air resistance (hard kick velocity and soft kick velocity) on a football. The formula used are:

Drag force=(drag coefficient *density of air*cross sectional area of football*$kickvelocity$2)/2

Hard kick velocity=120*speed conversion

Soft kick velocity=10* speed conversion

For the formula of trajectory following URL was used.
https://www.toppr.com/guides/physics-formulas/trajectory-formula /

TABLE 1. Data to be used in the Experiment

	Exp. 1.1	Exp. 1.2	Exp. 1.3

	Exp. 1.1	Exp. 1.2	Exp. 1.3
Imported data	pi, exponential, square-root	Pi, tan, cos	Pi
Pre-defined data	m=0, s=2.0, x=1.0	Y0=1, g=9.8 m/s^2	Drag coefficient(dc)= 0.2, Air density(ad)= 1.2, mass(m)= 0.43 kg, g=9.8 m/s^2
User-defined data	None	Degree, 'u', 'x'	None
Formula used	f=1/(sqrt(2*pi)*s) *exp(-0.5*((x-m)/s)**2)	1. rad=(pi/180)*deg 2. u=u/3.6 3. y= (x*tan(rad))+((g*(x**2))/(2*(u * *2)*(cos(rad)**2)))+y0	1. a=pi*0.11**2 2. F=m*g 3. sc=1000.0/3600 4. v=120*sc 5. v=10*sc 6. f=0.5*dc*ad*a*v **2

1.3 Software Used

1. Anaconda Navigator
2. Jupyter Notebook

1.4 Procedure

1. After installing anaconda navigator, open anaconda navigator and then select Jupyter Notebook and click on 'Launch'.
2. In Jupyter Notebook click on 'New Launcher' and then single click on 'Python3' under Notebook.
3. Type your program to get the desired output.
4. To view the output, click on 'Run' or press 'Shift+Enter' to execute program of theselected cell. Note: In case of error, refer to the error message and do the required changes.

1.5Observation

Exp. 1.1

Code:

```
from math import *
m=0
a=2.0
s=1.0
f=1/(sqrt(2*pi)*a)*exp(-0.5*((s-m)/a)**2)
print(f)
```

Output:

```
0.17603266338214976
```

Exp. 1.2

Code:

```
from math import *
y1=1
deg=float(input("Enter the angle="))                    #Taking input for angle in degree
g=9.81
u=float(input("Enter the initial velocity="))           #Taking input for velocity in km/hr
x=float(input("Enter the value of x="))                 #horizontalposition of object in meters

rad=(pi/180)*deg                                        #Converting degree to radian
u=u/3.6                                                 #Converting km/hr to m/sec
y=(x*tan(rad))+((g*(x**2))/(2*(u**2)*(cos(rad)**2)))+y1
print("The trajectory of the ball is=",y)
```

Output:

```
Enter the angle=45
Enter the initial velocity=10
Enter the value of x=2.5
The trajectory of the ball is= 11.446099999999998
```

Exp. 1.3

Code:

```
from math import *
dc=0.2                          #drag coefficient
ad=1.2                          #density of air(kg m^-3)
a=pi*0.11**2                    #football cross sectional area
m=0.43                          #mass of football
g=9.81                          #gravitational acceleration
Fg=m*g
sc=1000.0/3600
v=120*sc                        #hard kick velocity
fd=0.5*dc*ad*a*v**2
print("For hard kick velocity=",v,"the gravitational force is",Fg,"and the drag force is",fd)
v1=10*sc                        #soft kick velocity
fd1=0.5*dc*ad*a*v1**2
print("For a soft kick velocity=",v1,"the gravitational force is",Fg,"and drag force is",fd1)
```

Output:

```
For hard kick= 33.333333333333336 the gravitational force is 4.2183 and the drag force is 5.068436147
791534
For a soft kick velocity= 2.7777777777777777 the gravitational force is 4.2183 and drag force is 0.03
519747324855231
```

1.6Result

Thus Python was used as a calculator, code was written to evaluate the trajectory of a ball and code was written find air resistance of a football.

2. Program on Making Modules

2.1 Aims

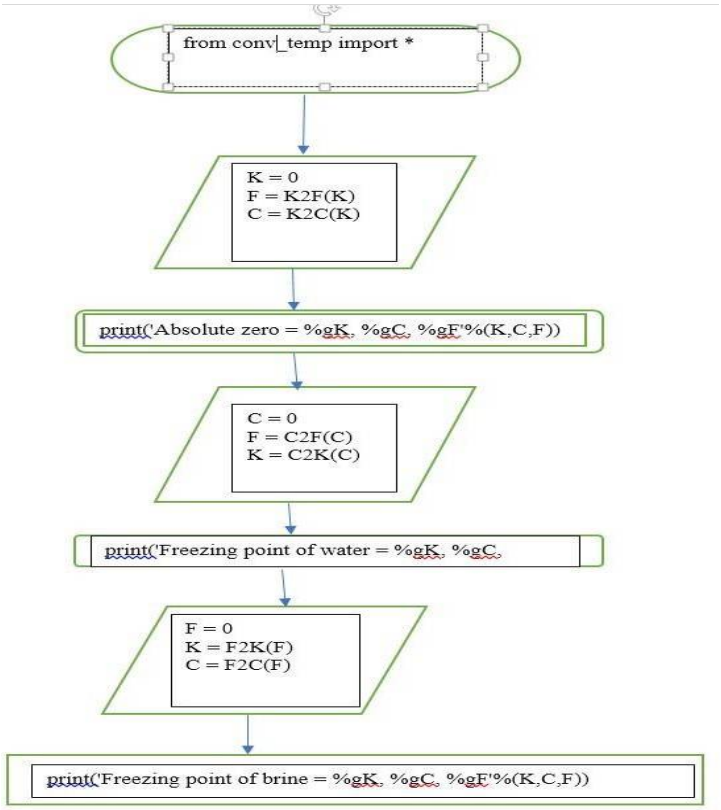
- Creating a module in Python for temperature conversions.
- Creating a module in Python for computing with interest rates.
- Creating a module in Python using Elif Statementto evaluate the values of a given function.

2.2 Software Used

1. Anaconda Navigator
2. Jupyter Notebook

SECTION A – Module for Temperature Conversions

2.3 Flowchart & Algorithm



1. Import all functions from con_tempmodule.
2. Give K=0 and use the K2F AND K2C functions to performconversions.
3. Print the absolute zero value after conversions in threeunits.
4. Give C=0 and use the C2F AND C2K functions to performconversions.
5. Print the freezing point of water value after conversions in threeunits.
6. Give F=0 and call the F2K AND F2C functions to performconversions.
7. Print the freezing point of brine after conversions in threeunits.

2.4 Procedure

1. After installing anaconda navigator, open anaconda navigator and then select Jupyter Notebook and click on 'Launch'.
2. In Jupyter Notebook click on 'New Launcher' and then single click on 'Python3' under Notebook.
3. Type your program to get the desired output.
4. To view the output, click on 'Run' or press 'Shift+Enter' to execute program of theselected cell. Note: In case of error, refer to the error message and do the required changes.

2.4 Program

Creating functions in the module con_temp.py

```
def C2F(C):
    F=(9./5)*C+32
    return
F def
F2C(F):
   C=(5./9)*(F-32)
   return C
```

```
def C2K(C):
    K=C+273.
    15
    return
K def
K2C(K):
    C=K-273.15
    return
C def
F2K(F):
    K=F2C(F)+273.15
    return
K def
K2F(K):
    F=C2F(K-273.15)
    return F
```

After importing the module and uploading to jupyter as con_temp.py

```
from con_temp
import * K = 0
F =
 K2F(K)
 C =
 K2C(K)
print('Absolute zero = %gK, %gC,
%gF'%(K,C,F)) C = 0
F =
 C2F(C)
 K =
```

C2K(C)

print('Freezing point of water = %gK, %gC,

%gF'%(K,C,F)) F = 0

K =

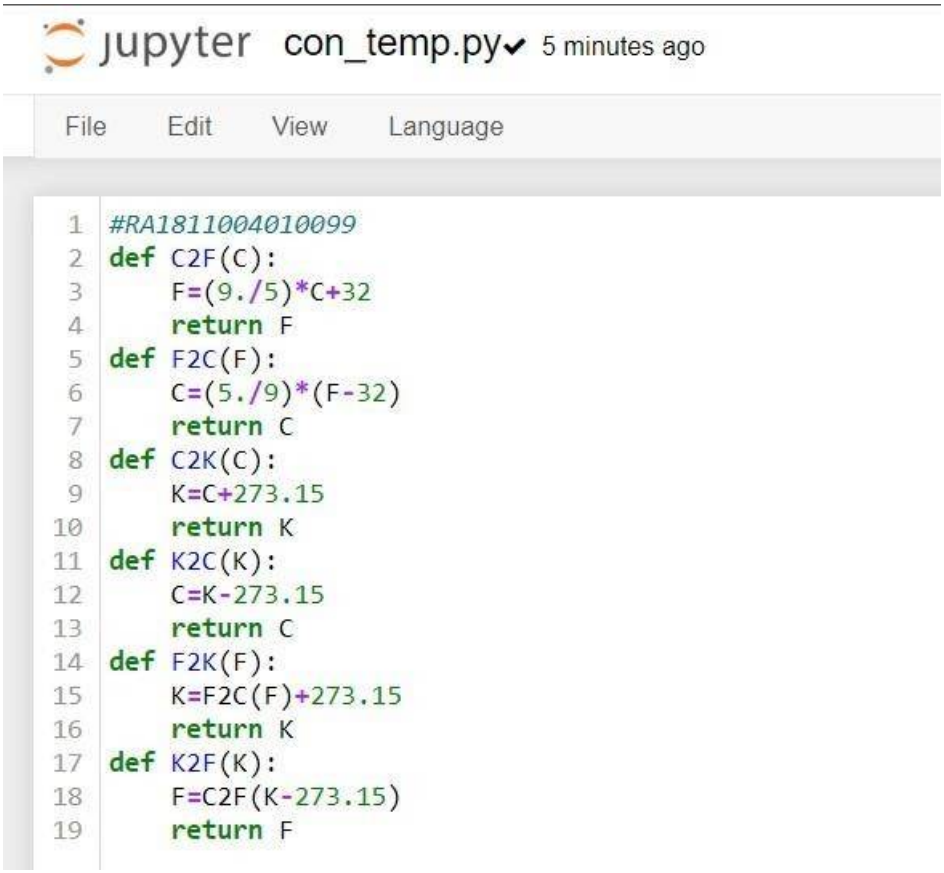
F2K(F)

C =

F2C(F)

print('Freezing point of brine = %gK, %gC, %gF'%(K,C,F))

2.5 Observation



```
#RA1811004010099
from con_temp import *
K = 0
F = K2F(K)
C = K2C(K)
print('Absolute zero = %gK, %gC, %gF'%(K,C,F))
C = 0
F = C2F(C)
K = C2K(C)
print('Freezing point of water = %gK, %gC, %gF'%(K,C,F))
F = 0
K = F2K(F)
C = F2C(F)
print('Freezing point of brine = %gK, %gC, %gF'%(K,C,F))
```

```
Absolute zero = 0K, -273.15C, -459.67F
Freezing point of water = 273.15K, 0C, 32F
Freezing point of brine = 255.372K, -17.7778C, 0F
```

2.6 Result

Thus, the python functions were created in a module convert_temp.py to declare the different formulae to convert temperatures and their outputs were obtained by calling the respective function from the module.

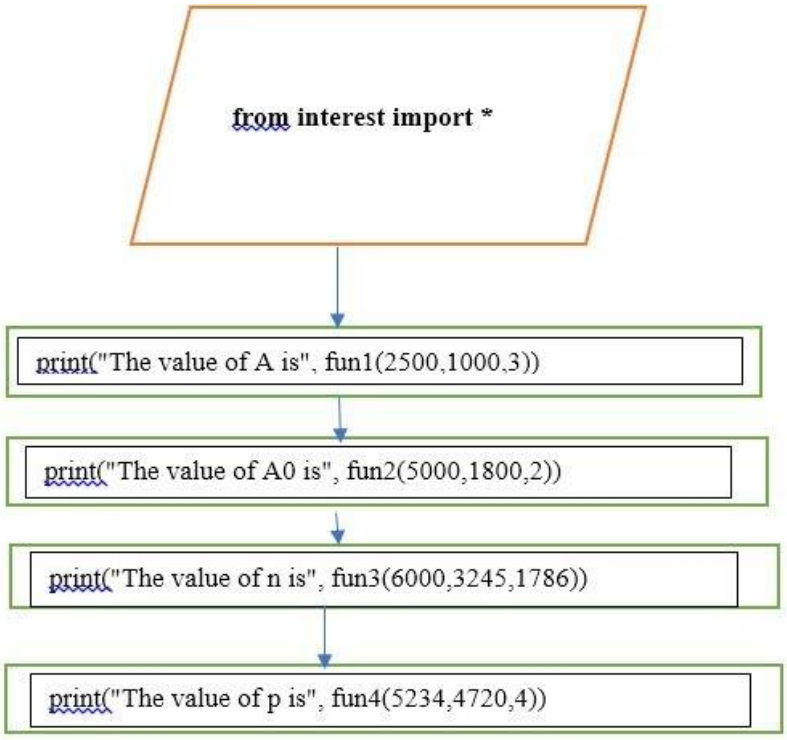
S.NO	INPUT	OUTPUT
1	K=0	Absolute zero = 0K, -273.15C, -459.67F
2	C=0	Freezing point of water = 273.15K, 0C, 32F
3	F=0	Freezing point of brine = 255.372K, -17.7778C, 0F

The outputs that are generated for the different sets of input values are represented in the table.

SECTION B – Module for Computing Interest Rates

Flowchart/Algorithm

The module with the formulas for calculations is created and uploaded in Jupyter in .py format so that we can import it in our programs.



1. Import all the functions that we defined in the moduleamount.

2. Hardcode values of A0,p and n in function call A to calculate and print final amountA.

3. Hardcode values of A,p and n in function call A0 to calculate and print initial amountA0.

4. Hardcode values of A, A0 and p in function call n to calculate and print number of yearn.

5. Hardcode values of A, A0 and n in function call A to calculate and print the percentagep.

The flow chart explains the order in which the program is written and executed.

Introduction/Procedure

1. Open the Jupyter notebook from AnacondaNavigator.
2. Define the various functions that work on problem related to interests. 3) Rename the file as amount. Download it as '.py' file and upload in Jupyter Notebook. This is the createdmodule.
3. Call the function within the print statement by giving corresponding input values to thefunction.
4. Click on the RUNbutton.

5. The desires output is now displayed on thescreen.

Program Code:

Creating interest.py module

```
from math import

* def

fun1(A0,p,n):

    A=A0*((1+(p/36000))**n)
    return A
def fun2(A,p,n):

   A0 = A * ((1+(p/36000))**(-n))

    return A0

 def fun3(A,A0,p):

    n=log(A/A0)/(log(1+(p/36000)))
    return n
def fun4(A,A0,n):
    p = 36000 * ((A/A0)**(1/n) - 1)
    return p
```

After importing the module and uploading to Jupyter as interest.py

```
from interest import *

print("The value of A is",fun1(2500,1000,3))
print("The value of A0 is",fun2(5000,1800,2))
print("The value of n is",fun3(6000,3245,1786))
print("The value of p is",fun4(5234,4720,4))
```

interest,py module

```
#RA1811004010099
from math import *
def fun1(A0,p,n):
    A = A0 * ((1+(p/36000))**n)
    return A
def fun2(A,p,n):
    A0 = A * ((1+(p/36000))**(-n))
    return A0
def fun3(A,A0,p):
    n = log(A/A0)/(log(1+(p/36000)))
    return n
def fun4(A,A0,n):
    p = 36000 * ((A/A0)**(1/n) - 1)
    return p
```

Program that imports module

```
#RA1811004010099
from interest import *
print("The value of A is",fun1(2500,1000,3))
print("The value of A0 is",fun2(5000,1800,2))
print("The value of n is",fun3(6000,3245,1786))
print("The value of p is",fun4(5234,4720,4))
```

```
The value of A is 2714.1739540466388
The value of A0 is 4535.147392290249
The value of n is 12.694085243079137
The value of p is 942.4275849421413
```

Screenshot of the program performed in jupyter notebook.

Results and Conclusion

Thus, the python functions were created in a module interest.py to declare the different formulae to compute with interest and their outputs were obtained by calling the respective function from the module.

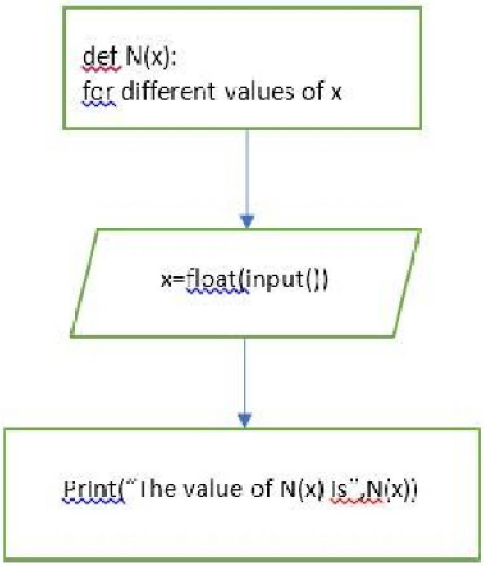
S.NO	INPUT	OUTPUT
1	A0 = 2500 p = 1000 n = 3	`The value of A is 2714.1739540466388`

2	A = 5000 p = 1800 n = 2	`The value of A0 is 4535.147392290249`
3	A=6000 A0=3245 p=1786	`The value of n is 12.694085243079137`
4	A=5234 A0=4720 n=4	`The value of p is 942.4275849421413`

The outputs that are generated for the different sets of input values are represented in the table.

: Elif Statement: To evaluate the values of a givenfunction.

Flowchart:



Source code

```
deffun(x):
   if x <0:
          return 0
   elif 0 <= x <1:
          return x
   elif 1 <= x <2:
          return 2-x
   elif x >= 2:
          return 0
y=fun(1)
```

print(y)

output:

```
#RA1811004010099
def fun(x):
    if x < 0:
        return 0
    elif 0 <= x < 1:
        return x
    elif 1 <= x < 2:
        return 2-x

    elif x >= 2:
        return 0

y=fun(1)
print(y)
```

```
1
```

References:

https://www.tutorialspoint.com/python/python_functions.html

3. Programs on lists and loops

Aim

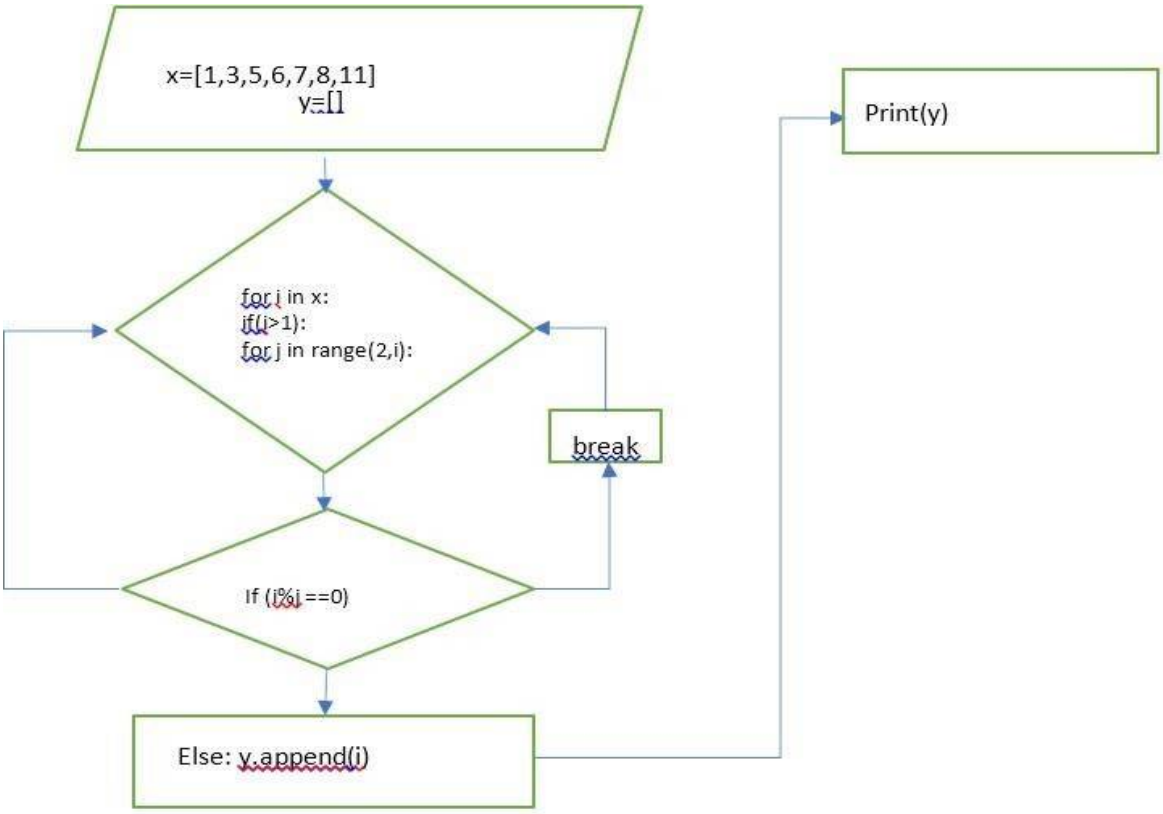
LIST: To print all the prime numbers from the list usingloops.

LOOP: Make a table of values from a formula usingloops.

List: To print all the prime numbers from the list usingloop

Flow Chart /Introduction / Background :

In this program a list is created from which only the prime numbers are extracted and appended into a new list and the new list is printed.



1. Create a list with random input values and an empty list to store primevalues.
2. Create a loop in range of the list values and then check for the condition of 'i' in list, if 'i'>1 and start another loop for the numbers in range 2 to the'i'.
3. The condition is true, it breaks and goes to first loop if not then else condition is checked.
4. Else condition is satisfied allows the number to be appended to secondlist.
5. The entire list is printed at thelast.

The flow chart explains the order in which the program is typed and executed.

Materials / Equipment :

1. Anaconda Navigator
2. JupyterNotebook

Procedure:

1) Open the Jupyter notebook from AnacondaNavigator.
2) Create a list 'x' with the values,x=[1,3,5,6,7,8,11].
3) Write the code to obtain the prime numbers from the list using forloop.
4) Append the prime numbers to a empty list'y'.
5) Print the prime numbers using print statement.
6) Run the program to get desiredoutput.

Program Code:

```
x=[1,3,5,6,7,8,11]

y=[]

print("theprimenumbersinthelistx=",x,",are:")
defprime(x):
    for i
     inx:
     if(i>1
     ):
       for j in
           range(2,i):
           if(i %
           j==0):
             break
        else:
          y.append(i)
prime(x)
print(y)
```

Results and Conclusion :

Thus, a list with the 1,3,5,6,7,8,11 were given as the input and the prime numbers from the list were displayed as the output.

INPUT	Output
x=[1,3,5,6,7,8,11]	[3, 5, 7, 11]

The outputs that are generated for the given sets of input values are represented in the table.

OUTPUT(Jupyter notebook):

```
x=[1,3,5,6,7,8,11]
y=[]
print("the prime numbers in the list x=",x,", are: ")
def prime(x):
    for i in x:
        if(i>1):
            for j in range(2,i):
                if(i % j==0):
                    break
            else:
                y.append(i)
prime(x)
print(y)
```

```
the prime numbers in the list x= [1, 3, 5, 6, 7, 8, 11] , are:
[3, 5, 7, 11]
```

LOOP: MAKE A TABLE OF VALUES FROM AFORMULA

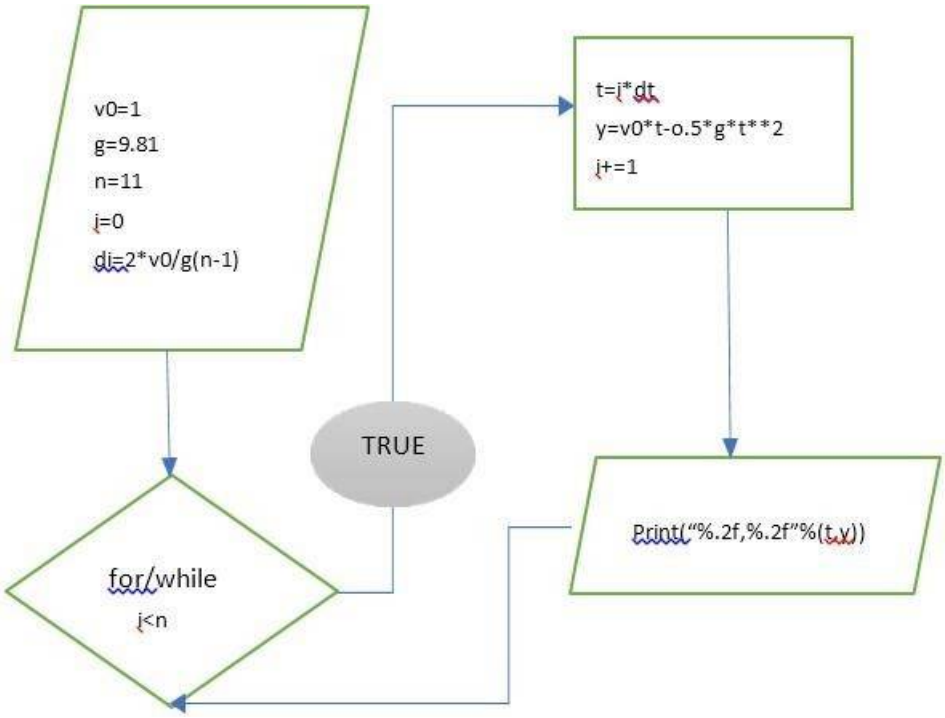
Introduction / Background

Towritecodethatprintsaformattedtableof‘t‘andy(t)values,where,y(t)= $v_o(□)-{}^{1}2\_g□^{2}$

Use (n) uniformly spaced t values throughout the interval [0, $2v_0/g$].

(i) Using for loop to produce the table.

(ii) Using while loop to produce thetable.



1. Hard code the values of v0, g, n and i(when using while loop). Give the formula for the small difference in timedt.
2. Check for the condition i<n using either for or whileloop.

3. If condition is TRUE, perform the operations given in the code for t,y andi.

4. Print the value of the resultant values of t,y in atable.
The flow chart above explains the order in which the program is written and executed.

Procedure

1) Open the Jupyter notebook from AnacondaNavigator.
2) Hard code the values of v0=1, g=9.81, n=11 and i=0(when using whileloop).
3) Write the code to obtain the table of values using the formula with for loop or whileloop.
4) Print the table using printstatement.
5) Run the program to get desiredoutput.

Code: Using for loop:

```
v0=1
g=9.81
n=11

dt=2*v0 / g / (n-1)
 print("%6s,%6s"%('t',
 'y')) for i
 inrange(0,n):
    t=i*dt
    y=v0*t-0.5*g*t**2
    print("%6.3f , %6.3f" % (t,y))
```

Using while loop:

```
v0=1

g=9.81

n=11

 i=0

dt=2*v0 / g / (n-

 1) print("%6s ,

 %6s"
%('t','y')) while

    i<n: t=i*dt

    y=v0*t-
0.5*g*t**2
print("%6.3f ,
%6.3f"
%

   (t,y))

   i+=1
```

Results and Conclusion

Thus, a nicely formatted table with values of t and y were created using the for and while loops with the given formula.

INPUT		OUTPUT
v0=1 g=9.8 1 n=5 i=0	t=i*dt y=v0*t-0.5*g*t**2 dt=2*v0/g/(n-1)	t, y 0.000,0.000 0.051,0.038 0.102,0.051 0.153,0.038 0.204,0.000

The output that is generated for the input values is represented in the table.

OUTPUT USING FOR LOOP(Jupyter notebook):

```
#RA1811004010099
v0=1
g=9.81
n=11
dt=2*v0 / g / (n-1)
print("%6s , %6s" %('t','y'))
for i in range(0,n):
    t=i*dt
    y=v0*t-0.5*g*t**2
    print("%6.3f , %6.3f" % (t,y))
```

```
     t ,      y
 0.000 ,  0.000
 0.020 ,  0.018
 0.041 ,  0.033
 0.061 ,  0.043
 0.082 ,  0.049
 0.102 ,  0.051
 0.122 ,  0.049
 0.143 ,  0.043
 0.163 ,  0.033
 0.183 ,  0.018
 0.204 ,  0.000
```

OUTPUT USING WHILE LOOP(Jupyter notebook):

```
#RA1811004010099
v0=1
g=9.81
n=11
i=0
dt=2*v0 / g / (n-1)
print("%6s , %6s" %('t','y'))
while i<n:
    t=i*dt
    y=v0*t-0.5*g*t**2
    print("%6.3f , %6.3f" % (t,y))
    i+=1
```

```
     t ,      y
 0.000 ,  0.000
 0.020 ,  0.018
 0.041 ,  0.033
 0.061 ,  0.043
 0.082 ,  0.049
 0.102 ,  0.051
 0.122 ,  0.049
 0.143 ,  0.043
 0.163 ,  0.033
 0.183 ,  0.018
 0.204 ,  0.000
```

Discussion:

If your teacher asks you to answer any discussion (post-Notebook) questions, this is the place.

Write out the questions then answer it below. Be very thorough, detailed and precise. Answer the questions with complete thoughts. Assume the reader does not know anything about this topic.

➔Only experiment questions were done.

References:

1. https://www.factmonster.com/math-science/mathematics/prime-numbers
2. https://www.physicsclassroom.com/class/1DKin/Lesson-6/Kinematic-Equations/

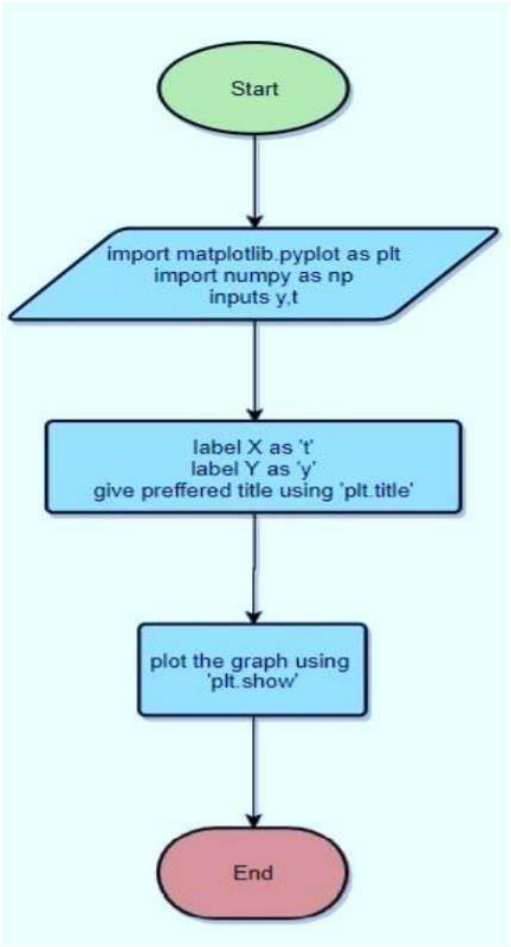
4. CURVE PLOTTING

Aim(s)

INTRODUCTION/BACKGROUND:

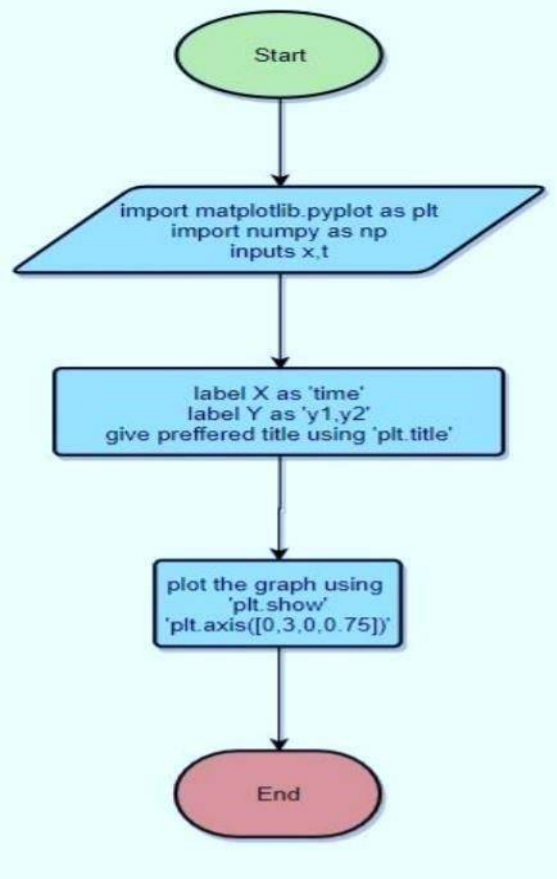
We need to use the matplotlib.pyplot to plot the curve and use the numpy module to do the mathematical computation required to plot the curve.

Plotting Single curve.



1. Import 2 modules matplotlib.pyplot andnumpy.
2. Define a function f(t) and returnt**2*exp(-t**2).
3. Assign t= np.linspace(0,3,51) to indicate 51 points between 0 and3.
4. Assign y= length oft.
5. To make it clear what our data represents ,lets include a title by using plt.title()command.
6. Plot the graph usingplt.plot(t,y).
7. Show the plot by usingplt.show().

Plotting Multiplecurve.



1. Import 2 modules matplotlib.pyplot and numpy. 2.

Define a function f1(t) and return t**2*np.exp(t**2).

3. Define a function f2(t) and return

t**2*f1(t). 4. Assign t= np.linspace(0,3,51)

to indicate 51 points between 0 and3.

5. To make it clear what our datarepresents

, lets include a title by using plt.title() command.

6. Plot the graph using plt.plot(t,y1,"r-")

for red colour.

7.Plot the graph using plt.plot(t,y2,"bo")

forblue colour.

8. Show the plot by usingplt.show().

Program and Input Data:

Plotting Single Curve

SOFTWARE USED:

- Anaconda Navigator
- JupyterNotebook

PROCEDURE:

1) Open the Jupyter notebook from the AnacondaNavigator.

2) First we have to import 2 modules matplotlib.pyplot and numpy ,check that the matplotlib module is installed in the command line ,check for matplotlib by running the "import matplotlib"command.

3) Every data set we may want to play around with different scales,using different ranges for our axes.

4) Let t=linspace(0,3,51) and the function f(t)= t**2*exp(-t**2).

5) Now we have our data,we can startplotting.

6) To make it clear what our data represents ,lets include a title by using plt.title()command.

7) Run the program to get the desiredoutput

8) We want to show the plot by usingplt.show().

DATA:

The curve □ = t**2*exp(-t**2) is plotted in this program.

CODE:

```
import numpy as np

import matplotlib.pyplot as plt

def f(t):
   return t**2*np.exp(-t**2)
t=np.linspace(0,3,51) y=np.zeros(len(t))
   for i in range(len(t)):
   y[i]=f(t[i])
plt.plot(t,y)
plt.show()
```

Plotting Multiple Curve SOFTWAREUSED:

- Anaconda Navigator
- JupyterNotebook

PROCEDURE:

1) Open the Jupyter notebook from the AnacondaNavigator.

2) First we have to import 2 modules matplotlib.pyplot and numpy ,check that the matplotlib module is installed in the command line ,check for matplotlib by running the "import matplotlib"command.

3) Every data set we may want to play around with different scales,using different ranges forour axes.

4) Let us consider two functions , f1(t) = t**2*np.exp(-t**2) and f2(t) =t**2*f1(t).

5) Write the function for each curve .Now we have our data,we can start plotting both in a single plot.

6) To make it clear what our data represents ,lets include a title by using plt.title()command.

7) Run the program to get the desiredoutput

8)We want to show the plot by using plt.show().

DATA:

The curve f1(t) = t**2*np.exp(-t**2) and f2(t) = t**2*f1(t) are plotted in this program.

CODE:

```
def
f1(t):
   return t**2*np.exp(-
t**2) def f2(t):
   return t**2*f1(t) t=np.linspace(0,3,51) y1=f1(t) y2=f2(t)
plt.plot(t,y1,"r-") plt.plot(t,y2,"bo") plt.xlabel("t")
plt.ylabel("y") plt.legend(["t^2*exp(-t^2)","t^4*exp(-
t^2)"],bbox_to_anchor=(1.1, 1.2)) plt.title("plotting two
curves") plt.show()
```

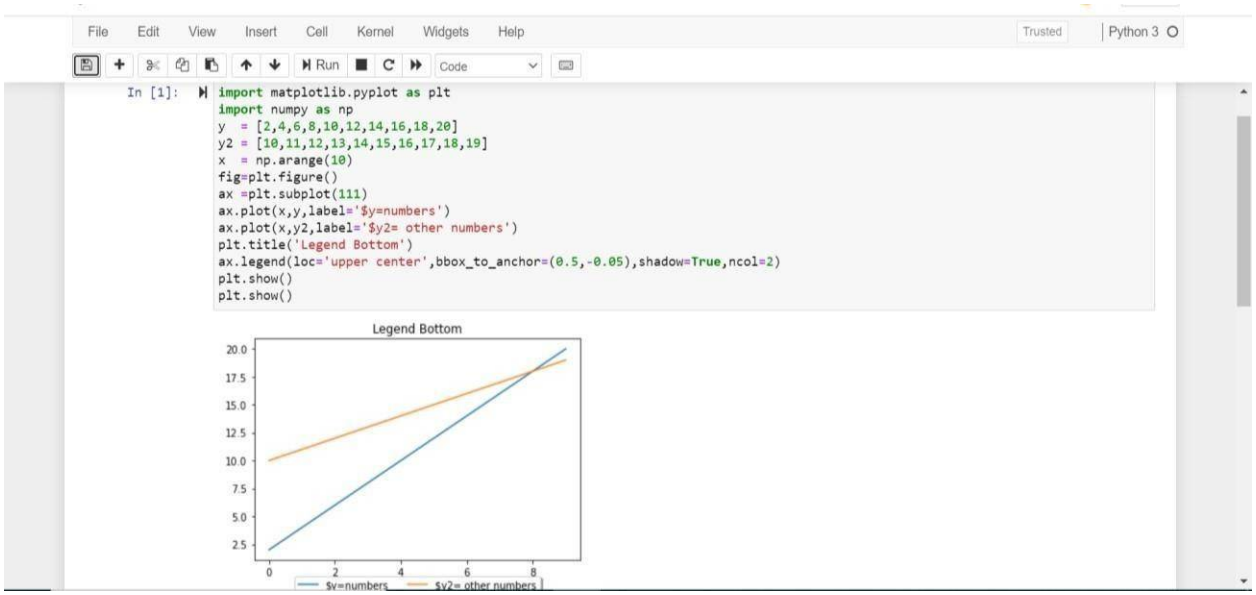
Discussion (PRE-LAB Session and POST- LAB Session) PRELAB:

1. What is the use of numpy package in curve plotting?
2. List the operation of command linspace with an example.
3. How to animate a function in python environment.
4. What is the use of arrange() function?
5. How to plot multiple functions in a single plot?

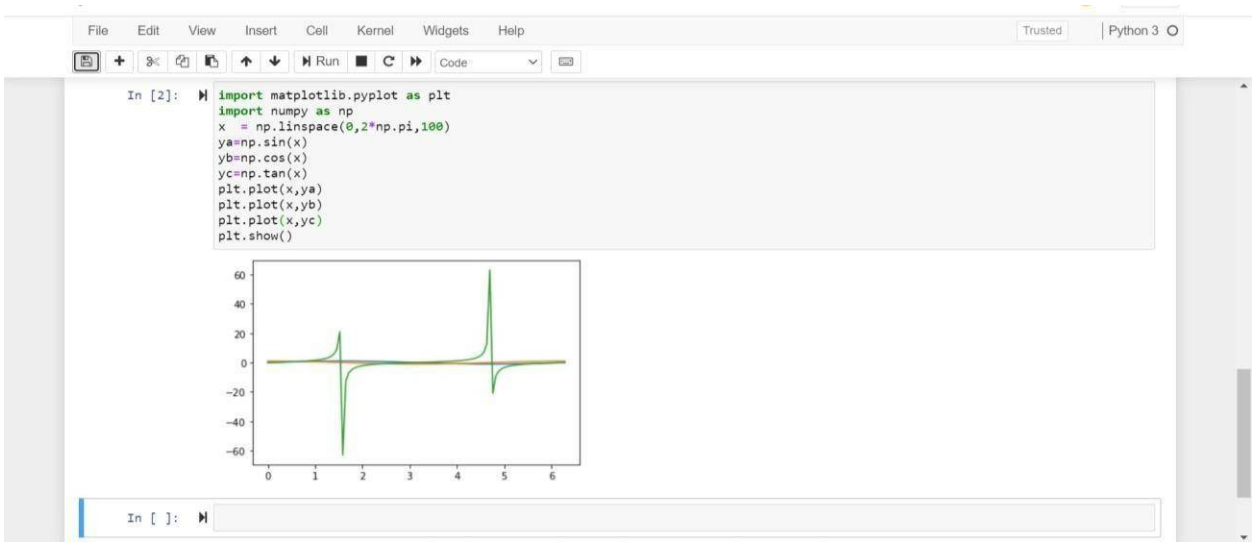
POSTLAB:

1. Give the command to place the legend inside, outside and bottom of the Curve.
2. What is the use of subplot command?
3. How to plot more than 2 functions in a graph

LEGENDBOTTOM



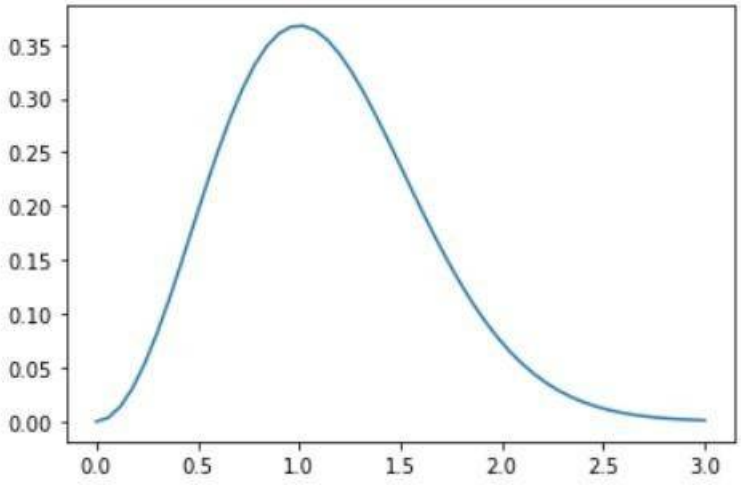
3) MORE THAN2PLOTS :



Results and Conclusion:

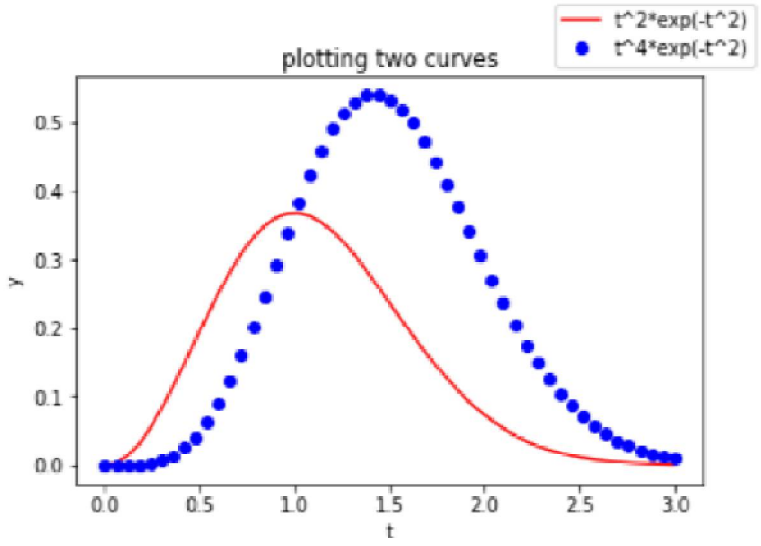
PLOTTING SINGLECURVE

In [10]:
```
#RA1811004010099
import numpy as np
import matplotlib.pyplot as plt
def f(t):
    return t**2*np.exp(-t**2)
t=np.linspace(0,3,51)
y=np.zeros(len(t))
for i in range(len(t)):
    y[i]=f(t[i])
plt.plot(t,y)
plt.show()
```



PLOTTING MULTIPLECURVE

```
In [24]: #RA1811004010099
def f1(t):
    return t**2*np.exp(-t**2)
def f2(t):
    return t**2*f1(t)
t=np.linspace(0,3,51)
y1=f1(t)
y2=f2(t)
plt.plot(t,y1,"r-")
plt.plot(t,y2,"bo")
plt.xlabel("t")
plt.ylabel("y")
plt.legend(["t^2*exp(-t^2)","t^4*exp(-t^2)"],bbox_to_anchor=(1.1, 1.2))
plt.title("plotting two curves")
plt.show()
```



RESULT:

Hence, the single □ = t**2*exp(-t**2) and multiple curves, f1(t) = t**2*np.exp(-t**2) and f2(t) = t**2*f1(t) were plotted by using the plot function from the module matplotlib.pyplot and the computations were done using the numpy package.

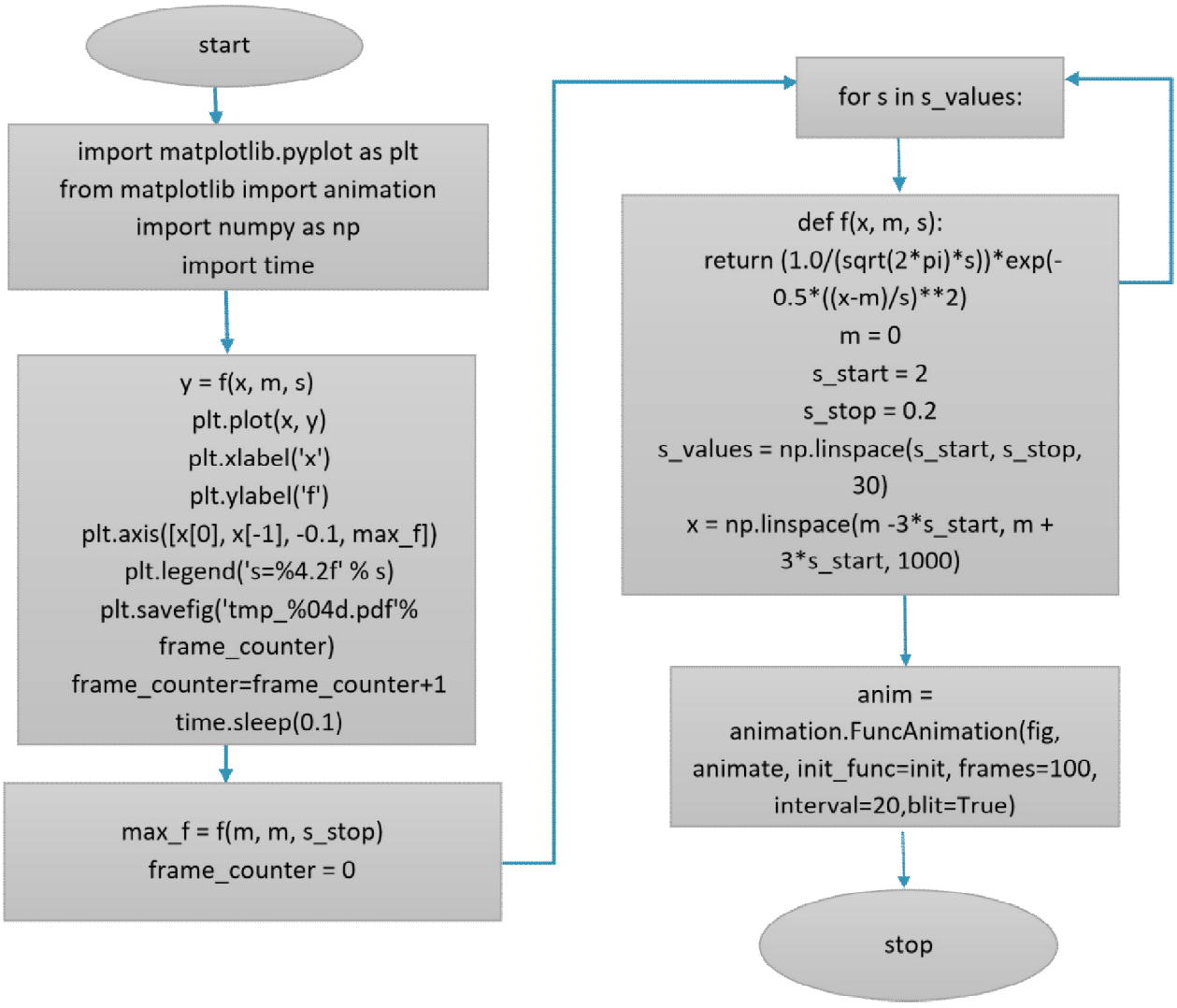
5.ANIMATING A FUNCTION – TEMPERATURE ON EARTH

Aim(s):

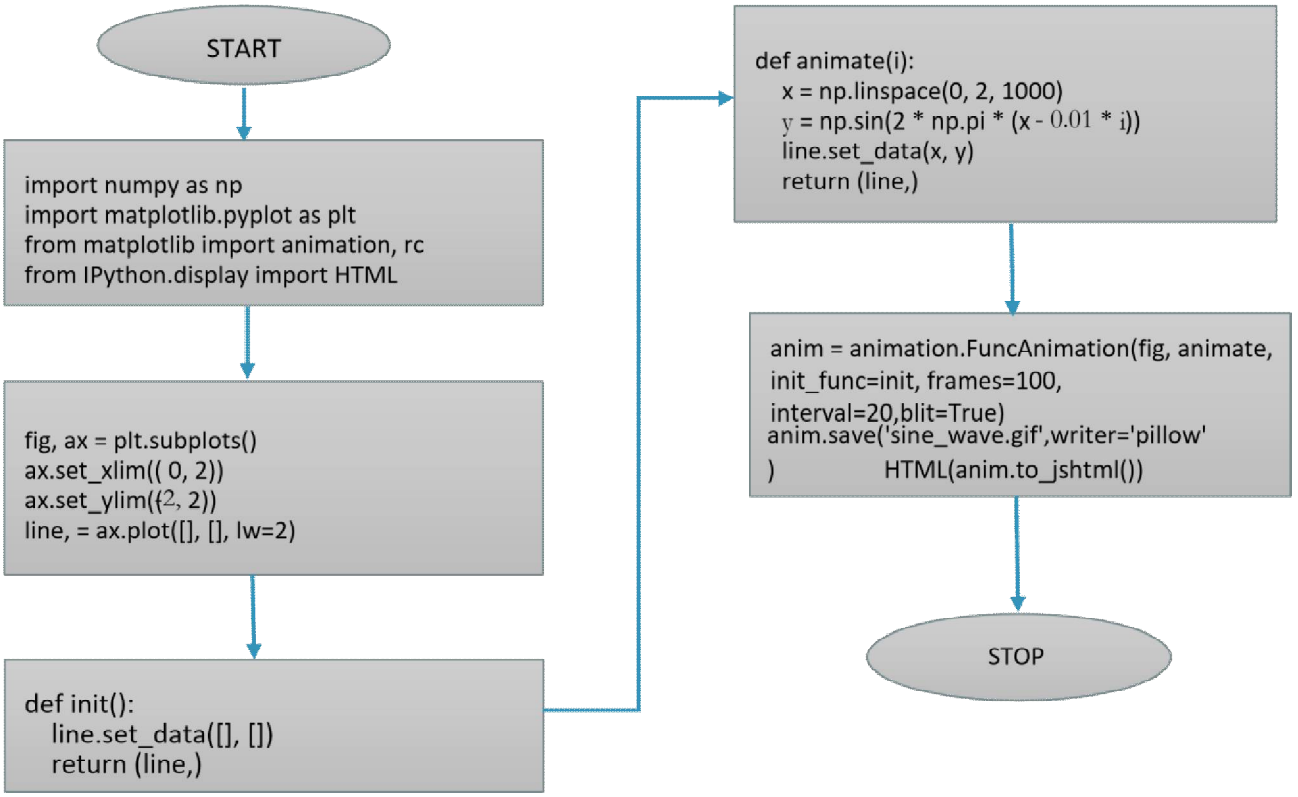
To generate an animate function using python for calculating the temperature on earth.

Flowchart/Algorithm

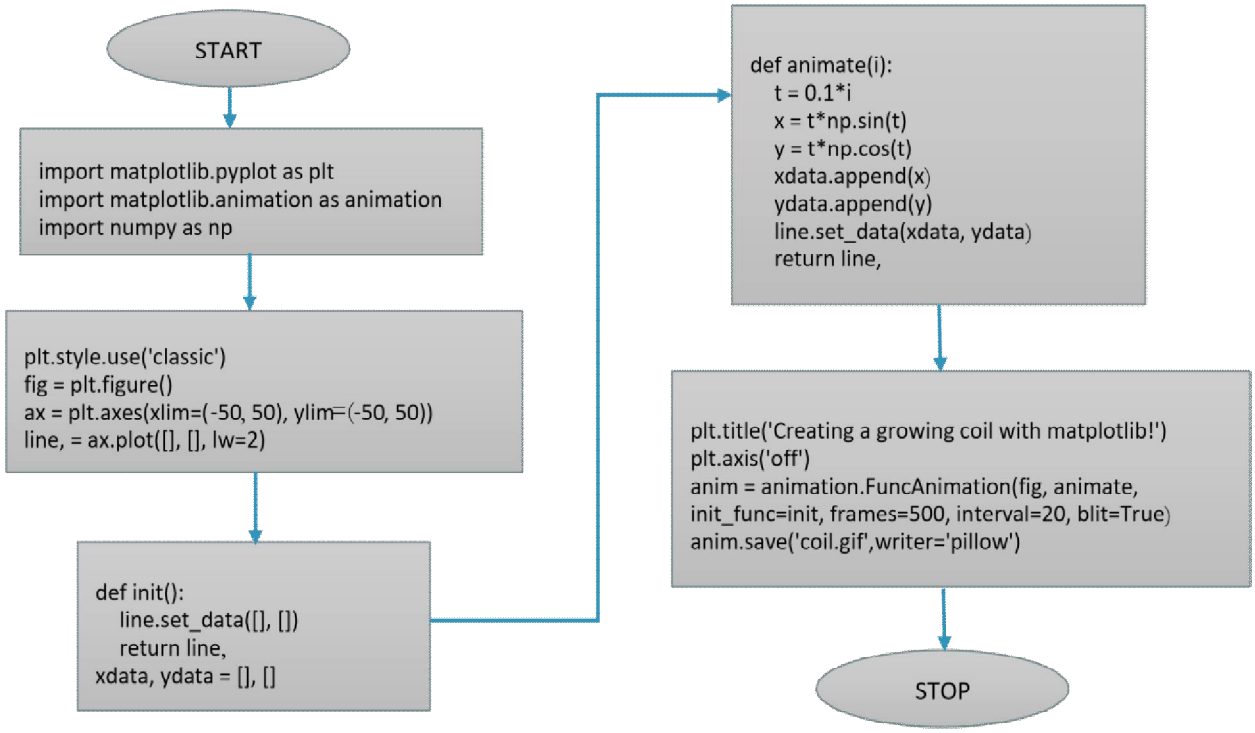
5.1 GENERATING FRAMES FOR ANIMATION:



CREATING A SINE WAVEANIMATION:



CREATING A GROWING COILANIMATION:



Program and Input Data

GENERATING FRAMES FORANIMATION:

Procedure:

1) Open the Jupyter notebook from the Anaconda Navigator.
2) Firstly from matplotlib import pyplot , animation and numpy packages, check that the modules are installed in the command line ,check these by running the commands.
3) Define functions x,m,s. By using gaussian bell function F(x)=(1.0/(sqrt(2*pi)*s))*exp(-0.5*((x-m)/s)**2).
4) Now we have our data m=0,start value=2,stopvalue=0.2
5) Create figure, axis and its properties for one frame, inside this give number of frames,interval.
6) Import time. Create animation object and synchronize animation object with plottedfigure.
7) Save theanimation.
8) Run the program to get the desired output.

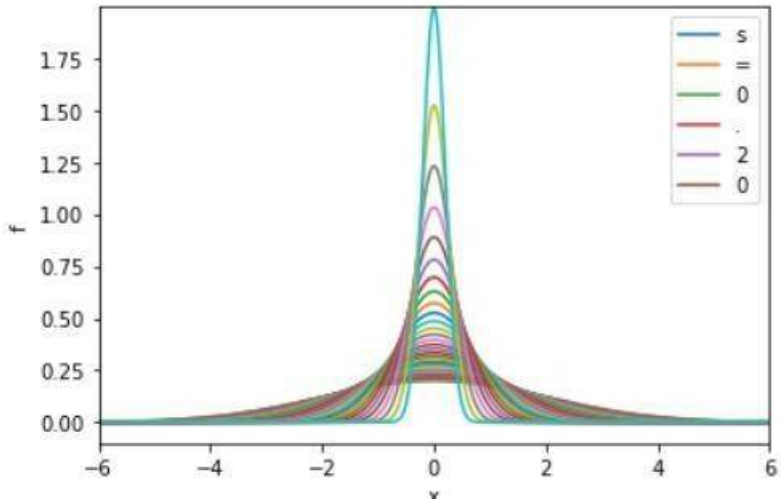
Code:

```
#RA1811004010099
import matplotlib.pyplot as plt
from matplotlib import animation
import numpy as np
import time
from numpy import *
from matplotlib.pyplot import *
def f(x, m, s):
    return (1.0/(sqrt(2*pi)*s))*exp(-0.5*((x-m)/s)**2)
m = 0
s_start = 2
s_stop = 0.2
s_values = np.linspace(s_start, s_stop, 30)
x = np.linspace(m -3*s_start, m + 3*s_start, 1000)
# f is max for x-m (smaller s gives larger max value)
max_f = f(m, m, s_stop)
import time
frame_counter = 0
for s in s_values:
    y = f(x, m, s)
    plt.plot(x, y)
    plt.xlabel('x')
    plt.ylabel('f')
    plt.axis([x[0], x[-1], -0.1, max_f])
    plt.legend('s=%1.2f' % s)
    plt.savefig('tmp_%04d.pdf'% frame_counter)
    frame_counter=frame_counter+1
    time.sleep(0.1)
anim = animation.FuncAnimation(fig, animate, init_func=init, frames=100, interval=20,blit=True)
anim.save('bell_curve.gif',writer='pillow')
```

```
import matplotlib.pyplot as plt from matplotlib

import animation

import numpy as np

import time from numpy

import * from matplotlib.pyplot

import * def f(x,m,s):

return (1.0/(sqrt(2*pi)*s))*exp(-0.5*((x-m)/s)**2) m = 0 s_start = 2 s_stop = 0.2

s_values = np.linspace(s_start, s_stop, 30) x = np.linspace(m -3*s_start, m + 3*s_start,1000)

# f is max for x=m (smaller s gives larger max value) max_f = f(m, m, s_stop)

 import time frame_counter = 0 for sins_values:
y = f(x, m,s)
plt.plot(x,y)
plt.xlabel('x')
plt.ylabel('f')
plt.axis([x[0], x[-1], -0.1, max_f])
plt.legend('s=%4.2f' % s)
plt.savefig('tmp_%04d.pdf'%
frame_counter)
frame_counter=frame_counter+1

  time.sleep(0.1)

anim = animation.FuncAnimation(fig, animate, init_func=init,
frames=100, interval=20,blit=True)

anim.save('bell_curve.gif',writer='pillow')
```

Variables	Input_Data
m	0
s_start	2
s_stop	0.2
frame_counter	0

Output:



CREATING A SINE WAVEANIMATION:

Procedure:

1)Open the Jupyter notebook from the Anaconda Navigator.

2) Firstly from matplotlib import pyplot , animation and numpy packages and IPython.display import HTML packages, check that the modules are installed in the command line ,check these by running the commands.

3)Let F(t)= sin2*pi*t. Define init function that will make the animation happen .Intializes the data and also sets the axis limits.

4)Create figure,axis and its properties for one frame.

5)Define animation object and synchronize animation object with plotted figure, inside this give number of frames, interval.

6)Save the animation.

7)Run the program to get the desired output.

```
#RA1811004010099
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import animation, rc
from IPython.display import HTML

fig, ax = plt.subplots()
ax.set_xlim(( 0, 2))
ax.set_ylim((-2, 2))
line, = ax.plot([], [], lw=2)

def init():
    line.set_data([], [])
    return (line,)

def animate(i):
    x = np.linspace(0, 2, 1000)
    y = np.sin(2 * np.pi * (x - 0.01 * i))
    line.set_data(x, y)
    return (line,)

anim = animation.FuncAnimation(fig, animate, init_func=init, frames=100, interval=20,blit=True)
anim.save('sine_wave.gif',writer='pillow')
HTML(anim.to_jshtml())
```

Code:

Import numpy as np
import matplotlib.pyplot as plt from matplotlib
import animation, rc from IPython.display

import HTML

fig, ax = plt.subplots() ax.set_xlim((0,2))
ax.set_ylim((-2,2))

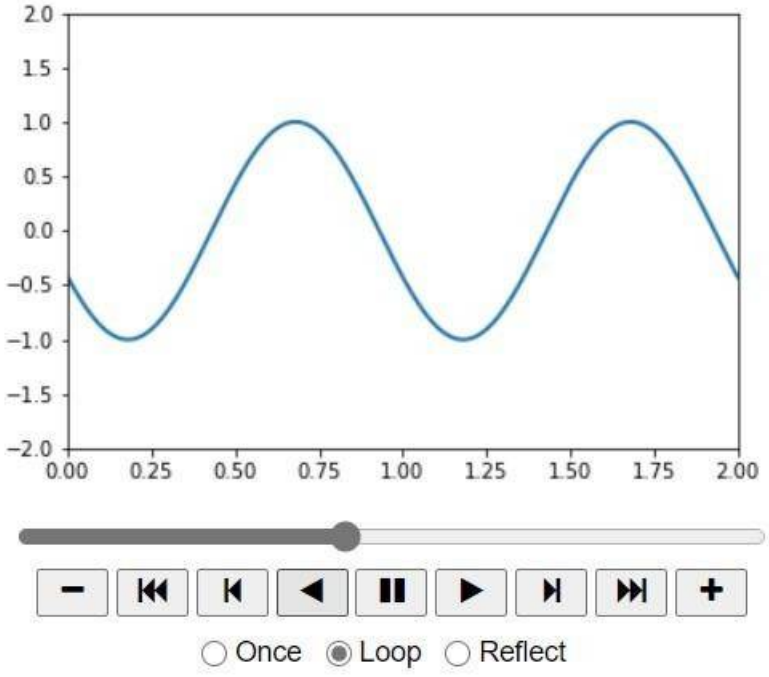
line, = ax.plot([], [], lw=2)

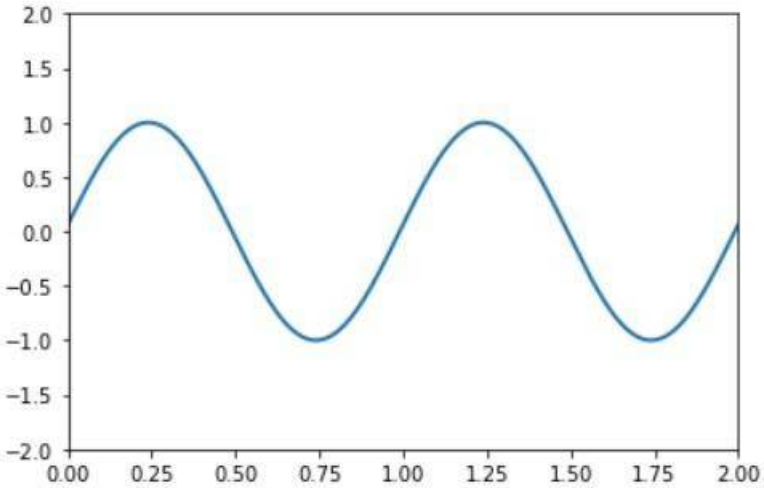
def init(): line.set_data([], []) return(line,)

def animate(i):x = np.linspace(0,2,1000)
y= np.sin(2 * np.pi * (x - 0.01 * i)) line.set_data(x, y)

return (line,)

anim = animation.FuncAnimation(fig, animate, init_func=init, frames=100, interval=20,blit=True) anim.save('sine_wave.gif',writer='pillow') HTML(anim.to_jshtml()) **Output:**





CREATING A GROWING COILANIMATION:

Procedure:

1) Open the Jupyter notebook from the Anaconda Navigator.

2) Firstly from matplotlib import pyplot , animation and numpy packages, check that the modules are installed in the command line ,check these by running the commands.

3) Initialize the function. Define init function that will make the animation happen .Initializes the data and also sets the axis limits.

4) Creating an empty plot or frame.

5) Create the lists to store x and y axis points.

6) Define animation object and synchronize animation object with plotted figure, inside this give number of frames,interval.

7) Appending new points to x, y axes points list.

8) Setting a title for the plot.

9) Hiding the axis details.

10) Call the animator. 11) Save the animation.

12) Run the program to get the desired outpu.t

Code:

```
#RA1811004010099
import matplotlib.pyplot as plt
import matplotlib.animation as animation
import numpy as np
plt.style.use('classic')
fig = plt.figure()
ax = plt.axes(xlim=(-50, 50), ylim=(-50, 50))
line, = ax.plot([], [], lw=2)
# initialization function
def init():
    # creating an empty plot/frame
    line.set_data([], [])
    return line,
# lists to store x and y axis points
xdata, ydata = [], []
# animation function
def animate(i):
    # t is a parameter
    t = 0.1*i
    # x, y values to be plotted
    x = t*np.sin(t)
    y = t*np.cos(t)
    # appending new points to x, y axes points list
    xdata.append(x)
    ydata.append(y)
    line.set_data(xdata, ydata)
    return line,
# setting a title for the plot
plt.title('Creating a growing coil with matplotlib!')
# hiding the axis details
plt.axis('off')
# call the animator
anim = animation.FuncAnimation(fig, animate, init_func=init, frames=500, interval=20, blit=True)
# save the animation as mp4 video file
anim.save('coil.gif',writer='pillow')
```

import matplotlib.pyplot as plt
import matplotlib.animation as

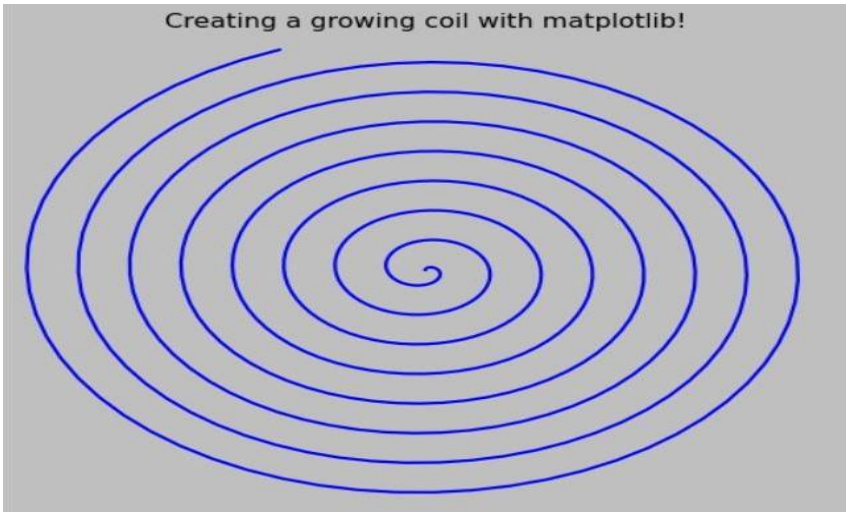
animation import numpy as np

plt.style.use('classic ') fig = plt.figure()

ax = plt.axes(xlim=(-50, 50), ylim=(-50, 50))

```
line, = ax.plot([], [], lw=2) # initialization function def init():
   # creating an empty plot/frame
   line.set_data([], [])
return line,

# lists to store x and y
axis points xdata, ydata
= [], [] # animation
function def animate(i):
                  # t is a
parameter
   t =0.1*i
   # x, y values to
be plotted    x =
t*np.sin(t)    y =
t*np.cos(t)
   # appending new points to x, y
axes pointslist      xdata.append(x)
ydata.append(y)
            line.set_data(xdata, ydata)
            returnline,

# setting a title for the plot

plt.title('Creating a growing coil with

matplotlib!') # hiding the axis details

plt.axis('off')

# call the animator

anim = animation.FuncAnimation(fig, animate, init_func=init, frames=500, interval=20,

blit=True) # save the animation as mp4 video file
anim.save('coil.gif',writer='pillow')
```

Output:



Discussion (PRE-LAB Session and POST-LAB Session)

(5

PRELAB:

1. What is the use of animate function in curve plotting?
2. List the operation of command linspace() with an example.
3. How to animate a function in python environment.
4. What is the use of arrange()function?
5. How to plot multiple functions in a single plot?
6. List the significance of numpy package.

POSTLAB:

1. Give the command to place the legend inside, outside and bottom of the Curve.
2. What is the use of subplot command?
3. How to plot more than 2 functions in a graph?

Result:

Therefore, generation of frames, creating a sine wave, creating a growing coil were plotted and executed successfully.

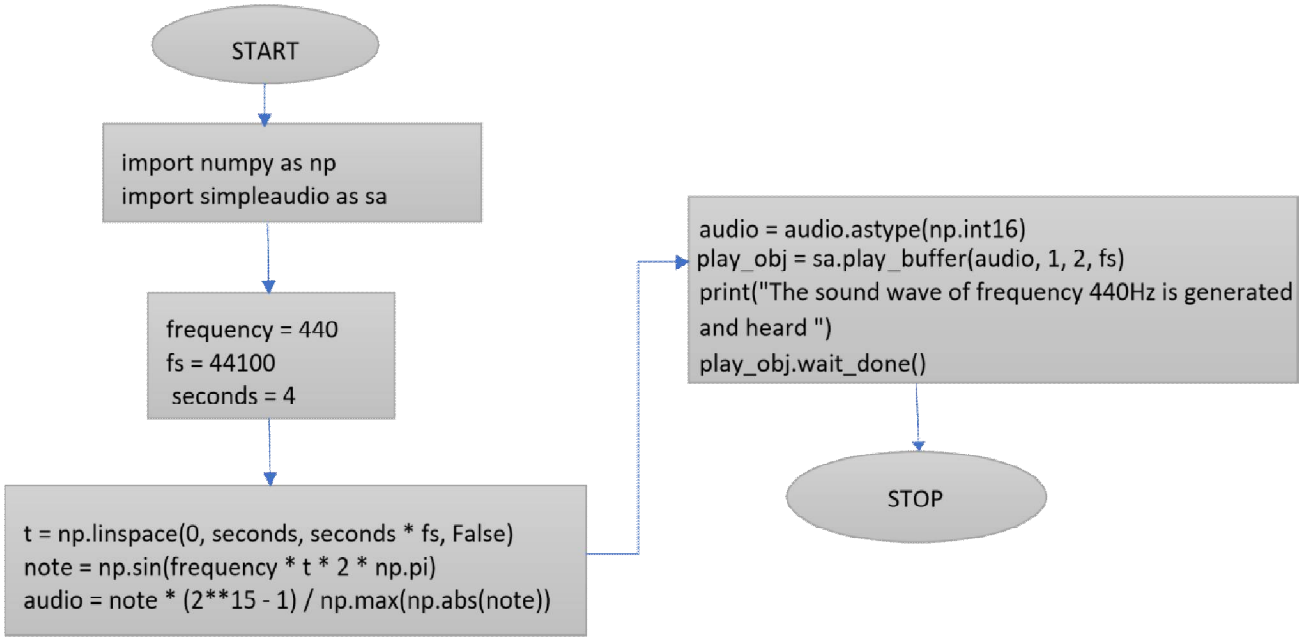
6.Sound generated by formula

Aim:
In this problem statement we will write a code that can generate sound of desired frequency (440 Hz in this case).

Software used:
- AnacondaNavigator
- Jupyter Notebook

Procedure:
1. Open Anaconda Navigator and open jupyterNotebook.
2. Import simpleaudio and numpypackages.
3. Give the input frequency as 440Hz , sampling frequency as 44100 Hzand seconds as 4 (which counts upto 3only).
4. Create array with seconds*sample_rate steps, ranging between 0 and seconds.
5. Generate a 440 Hz sinewave.
6. Ensure that highest value is in 16-bitrange.
7. Convert highest value to 16-bitdata.
8. Start playback and Wait for playback to finish beforeexiting.
9. Run the program and getoutput.

Flowchart:



INPUTS TABLE:

LIBRARIES USED	FREQUENCY OF SOUND WAVE(Hz)	NUMBER OF SAMPLES PER SECOND	GENERATION OF 440 HZ SINE WAVE	PLAYBACK FORMULA
numpy simpleaudio	frequency = 440	fs = 44100	note = np.sin(frequency * t * 2 * np.pi)	play_obj = sa.play_buffer(audio, 1, 2, fs)

Program and Output:

CODE:

```
import numpy as np import simple audio as sa

frequency = 440 # Our played note
will be 440 Hz
fs = 44100 # 44100 samples per
second seconds = 4 # Note duration
of 3 seconds # Generate array with
seconds*sample_rate steps, ranging
between 0 and seconds
t = np.linspace(0, seconds, seconds
* fs, False)
# Generate a 440 Hz sine wave
note = np.sin(frequency * t * 2 *
np.pi) # Ensure that highest value
is in 16-bit range
audio = note * (2**15 - 1)
/ np.max(np.abs(note))
# Convert to 16-bit data
audio =
audio.astype(np.int16) #
Start playback
play_obj = sa.play_buffer(audio, 1, 2,
fs) print("The sound wave of frequency
440Hz is generated and heard ")
# Wait for playback to finish before
exiting play_obj.wait_done()
```

```python
#RA1811004010099
import numpy as np
import simpleaudio as sa

frequency = 440 # Our played note will be 440 Hz
fs = 44100  # 44100 samples per second
seconds = 4  # Note duration of 3 seconds

# Generate array with seconds*sample_rate steps, ranging between 0 and seconds
t = np.linspace(0, seconds, seconds * fs, False)

# Generate a 440 Hz sine wave
note = np.sin(frequency * t * 2 * np.pi)

# Ensure that highest value is in 16-bit range
audio = note * (2**15 - 1) / np.max(np.abs(note))
# Convert to 16-bit data
audio = audio.astype(np.int16)

# Start playback
play_obj = sa.play_buffer(audio, 1, 2, fs)
print("The sound wave of frequency 440Hz is generated and heard ")

# Wait for playback to finish before exiting
play_obj.wait_done()
```

OUTPUT:

```
The sound wave of frequency 440Hz is generated and heard
```

Discussion (PRE-LAB Session and POST-LAB Session):

Pre-lab:

1. write the syntax for generating a sine wave of 440Hz.
2. Draw the flowchart for sound wave generation technique inpython.

Post-lab:

1. command for starting the playback

```
# Start: pLaybacb
p1ayobj      = sa.play buffer(aud1o, 1, 2,fs)
pr1nt ("The  soundNaveof          frequency 446Hz 1s generated and heard")

#waitfor        pLaybacb to fi n?sh before exiting
play obj.wait done()
```

Results and Conclusion:

Generation of sound using formula and difference equation is performed, executed properly and it is heard.

7. Reading student marks from a textfile and storing into dictionary data and assigning grades based on marks.
Reading student marks file into a dictionary data with the student name as key and computing the average grades.

Aim:

Reading student marks from a text file and storing into dictionary data and assigning grades based on marks.
Reading student marks file into a dictionary data with the student name as key and computing the average grades.

Reading student marks from a text file and storing into dictionary data and assigning grades based on marks.

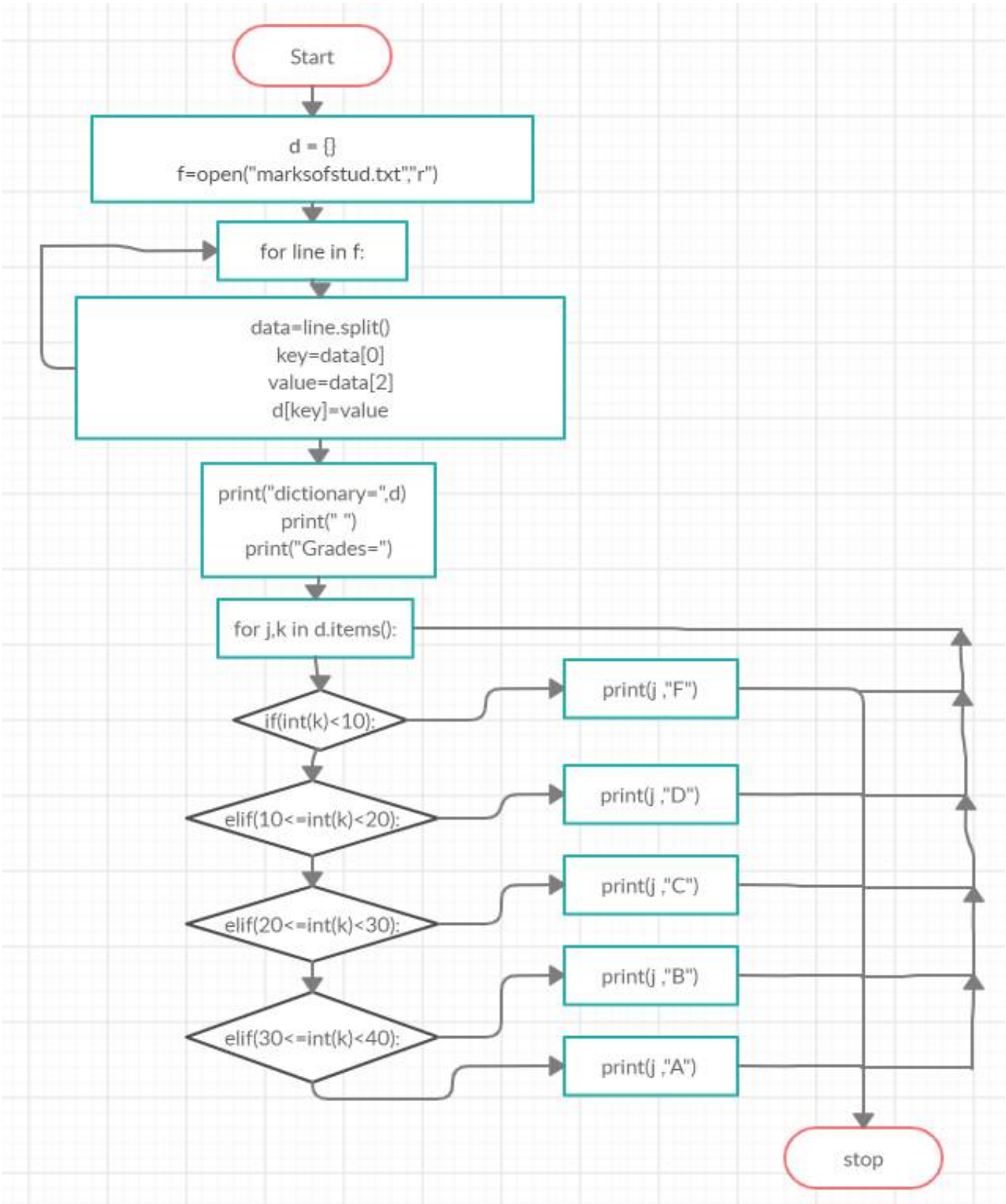
Software used:

1. Anaconda navigator
2. Jupyter Notebook

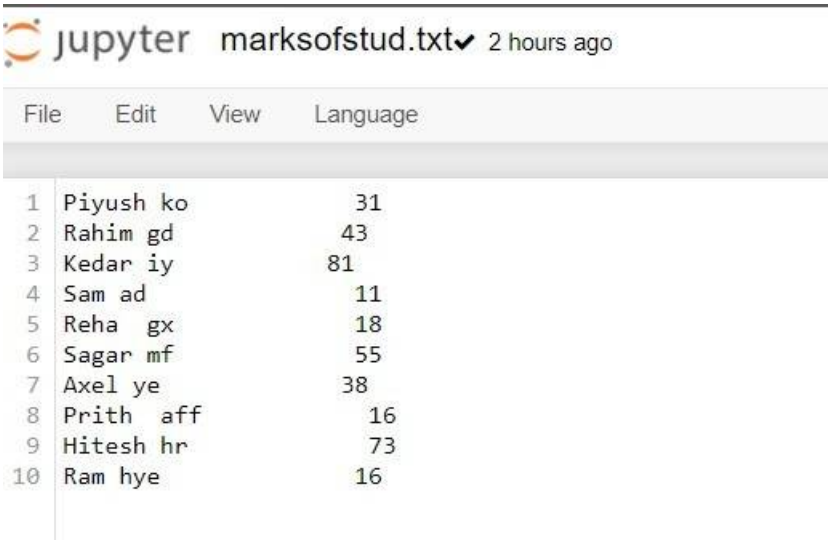
Procedure:

1. Create a file with the given data below in a text editor and save it as marksofstud.txt
2. Open Anaconda Navigator and launch Jupyter Notebook.
3. Upload the file marksofstud.txt in the Jupyter Notebook.
4. Open a new Python 3file.
5. Create an empty dictionary.
6. Open the file marksofstud.txt in read mode and read the lines in the file.
7. Create a dictionary with the values in the file with student name as key and mark as value.
8. Display the dictionary.
9. Find the grade of each and every student.
10. Display the grade of all the students.
11. Run the program to get the result.

Flowchart:


Start
d = {}
f=open("marksofstud.txt","r")
for line in f:
data=line.split()
key=data[0]
value=data[2]
d[key]=value
print("dictionary=",d)
print(" ")
print("Grades=")
for j,k in d.items():
if(int(k)<10):
print(j ,"F")
elif(10<=int(k)<20):
print(j ,"D")
elif(20<=int(k)<30):
print(j ,"C")
elif(30<=int(k)<40):
print(j ,"B")
print(j ,"A")
stop

Text file input:



Code:

```
d = {}
f=open("marksofstud.txt","r
") for line in f:
   data=line.split() key=data[0]
   value=data[2] d[key]=value

print("dictionary=",d)
print(" ") print("Grades=")

for j,k in
   d.items():
   if(int(k)<10):
      print(j,"F")
   elif(10<=int(k)<20):
      print(j,"D")
   elif(20<=int(k)<30):
      print(j,"C")
   elif(30<=int(k)<40):
      print(j,"B")
   else:
      print(j,"A")
```

Output:

```
#RA1811004010099
d = {}
f=open("marksofstud.txt","r")
for line in f:
    data=line.split()
    key=data[0]
    value=data[2]
    d[key]=value

print("dictionary=",d)
print(" ")
print("Grades=")

for j,k in d.items():
    if(int(k)<10):
        print(j,"F")
    elif(10<=int(k)<20):
        print(j,"D")
    elif(20<=int(k)<30):
        print(j,"C")
    elif(30<=int(k)<40):
        print(j,"B")
    else:
        print(j,"A")
```

```
dictionary= {'Piyush': '31', 'Rahim': '43', 'Kedar': '81', 'Sam': '11', 'Reha': '18', 'Sagar': '55', 'Axel': '38', 'Prith': '1
6', 'Hitesh': '73', 'Ram': '16'}

Grades=
Piyush B
Rahim A
Kedar A
Sam D
Reha D
Sagar A
Axel B
Prith D
Hitesh A
Ram D
```

Reading the student marks file into a dictionary data with student name as key and computing the average grades:

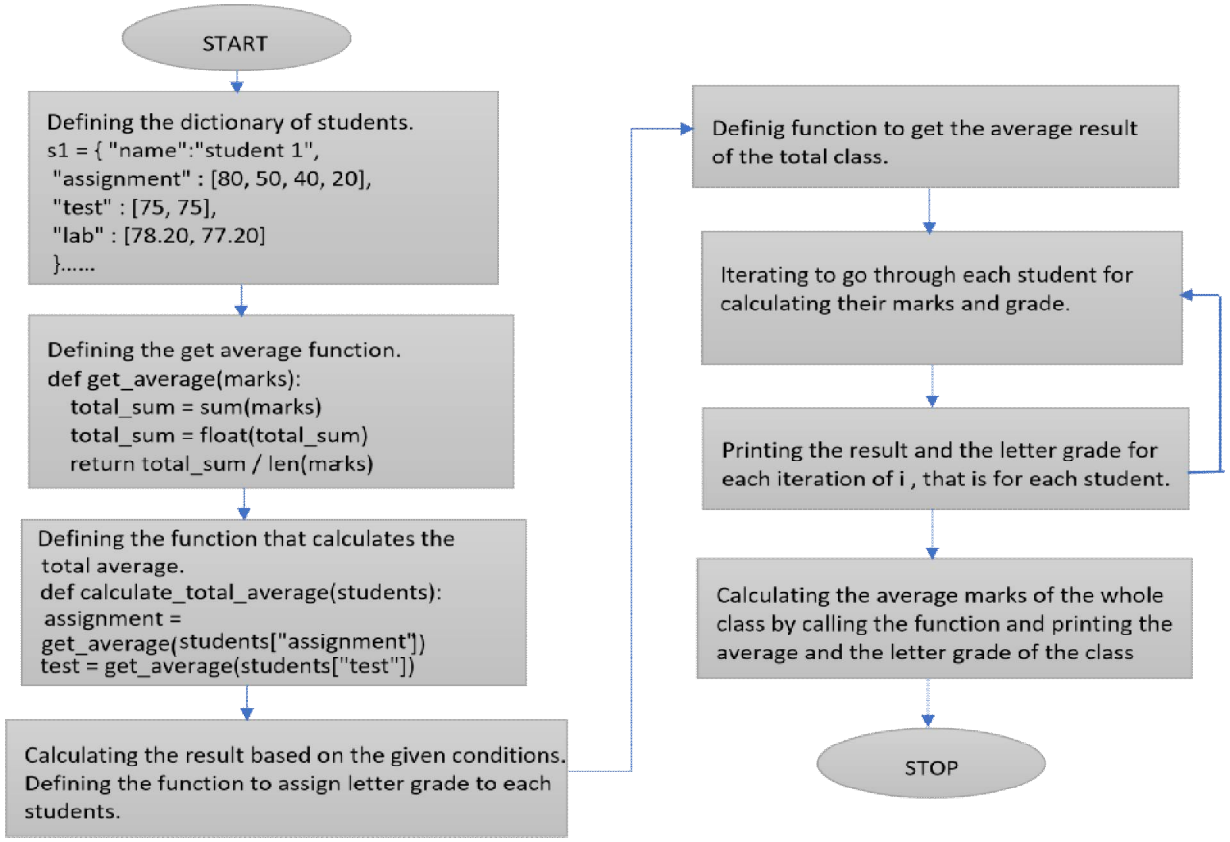
Software used:

Anaconda navigator
Jupyter Notebook

Procedure:

1. Open Anaconda Navigator and launch Jupyter Notebook.
2. Give the dictionary input for 6students.
3. Define a function that calculate the average as sum/number of elements
4. Define a function that calculates the total average as 0.1*assignment+0.7*test+0.2*practicals 5. Define a function that calculates letter grade of each student according to the marks obtained
6. Define a function that calculates the total average of the entire class.
7. Create a list of dictionary of all the 6students.
8. Calculate the total average and find the average grade of each student in the list by calling and printing the function.
9. Calculate the average of whole class and print it.
10. Run the program to get the result.

Flowchart:



CODE:

```
# 1. stud1 dictionary
s1 = { "name":"student 1",
       "assignment" : [80, 50, 40, 20],
       "test" : [75, 75],
       "lab" : [78.20, 77.20]
     }

# 2. stud2 dictionary
s2 = { "name":"student 2",
       "assignment" : [82, 56, 44, 30],
       "test" : [80, 80],
       "lab" : [67.90, 78.72]
     }

# 3. stud3 dictionary
s3 = { "name" : "student 3",
       "assignment" : [77, 82, 23, 39],
       "test" : [78, 77],
       "lab" : [80, 80]
     }

# 4. stud4 dictionary
s4 = { "name" : "student 4",
       "assignment" : [67, 55, 77, 21],
       "test" : [40, 50],
       "lab" : [69, 44.56]
     }
```

```
# 5. stud5 dictionary
s5 = { "name" : "student 5",
       "assignment" : [29, 89, 60, 56],
       "test" : [65,56],
       "lab" : [50,40.6]
     }
# 6. stud5dictionary
s6 = { "name" : "student 6",
       "assignment" : [45, 54, 37, 84],
       "test" : [78, 45],
       "lab" : [69, 87]
      }
# 7. stud5 dictionary
s7 = { "name" : "student 7",
       "assignment" : [78, 45, 66, 89],
       "test" : [71, 59],
       "lab" : [87, 64]
      }

# Function calculates
average def
get_average(marks):
   total_sum = sum(marks)
   total_sum =
   float(total_sum) return
   total_sum / len(marks)

# Function calculates total average
defcalculate_total_average(students):
   assignment =
```

```
    get_average(students["assignment"]) test
    =get_average(students["test"])
    lab = get_average(students["lab"])
    # Return the result based on weightage
    supplied # 10 % from assignments
    # 70 % from test
    # 20 % from lab-works
    return (0.1 * assignment + 0.7 * test + 0.2 * lab)

 # Calculate letter grade of each
student def
assign_letter_grade(score):
    if score >= 90: return "A"
    elif score >= 80: return
    "B" elif score >= 70:
    return "C" elif score >=
    60: return "D" else :
    return "E"

 # Function to calculate the total average marks of the whole
class def class_average_is(student_list):
    result_list = []

    for student in student_list:
        stud_avg
        =calculate_total_average(student)
        result_list.append(stud_avg)
    returnget_average(result_list)

 # Student list consisting
the # dictionary of all
```

```
students

students = [s1,s2,s3,s4,s5,s6,s7]


# Iterate through the students list and calculate their respective
average marks and letter grade

for i in students :

   print(i["name"]

   ) print("----- ")

   print("Average marks of %s is : %s "
%(i["name"], calculate_total_average(i)))


   print("Letter Grade of %s is : %s"
%(i["name"],
assign_letter_grade(calculate_total_average
(i))))


   print()



# Calculate the average of whole

class S class_av =

class_average_is(students)


print( "Class Average is %s" %(class_av))

print("Letter Grade of the class is %s " %(assign_letter_grade(class_av)))
```

CODE OUTPUT:

```
# 1. stud1 dictionary
s1 = { "name":"student 1",
        "assignment" : [80, 50, 40, 20],
        "test" : [75, 75],
        "lab" : [78.20, 77.20]
      }

# 2. stud2  dictionary
s2 = { "name":"student 2",
         "assignment" : [82, 56, 44, 30],
         "test" : [80, 80],
         "lab" : [67.90, 78.72]
       }

# 3. stud3  dictionary
s3 = { "name" : "student 3",
         "assignment" : [77, 82, 23, 39],
         "test" : [78, 77],
         "lab" : [80, 80]
       }

# 4. stud4  dictionary
s4 = { "name" : "student 4",
         "assignment" : [67, 55, 77, 21],
         "test" : [40, 50],
         "lab" : [69, 44.56]
       }

# 5. stud5  dictionary
s5 = { "name" : "student 5",
        "assignment" : [29, 89, 60, 56],
        "test" : [65, 56],
        "lab" : [50, 40.6]
      }
# 6. stud5  dictionary
s6 = { "name" : "student 6",
        "assignment" : [45, 54, 37, 84],
        "test" : [78, 45],
        "lab" : [69, 87]
       }
# 7. stud5  dictionary
s7 = { "name" : "student 7",
        "assignment" : [78, 45, 66, 89],
        "test" : [71, 59],
        "lab" : [87, 64]
       }
```

```
del- ge1_aver age (mar ks ) :
     tota1_sum - sum (ma rka )
     total_sum = float(total_sum)
     r'e1ur n tot a1_sum / 1en {ma rks }


def calculate total average(students):
     assignment = get_average(students["assignment"j)
     test = get_average(students{"test"j}
     lab = get_average(students§"lab"j)

     # Return the          based  on weightage supplied

             from

     r'e6urn    from  '* ass1gnmenI  -+        B . 7 " I eslz -+ B . 2 ^ 1 ab}



def assign_letter_grade(score):
     nf score >= 9B: return "A"
     elif score >= 8B: return "B"
     elif score >= 70: return "C"
     elif score >= 6B: return "D"
     else : return "E"

 # Function to calculate the total average marks of the whole class
def  c1ass_aver•age_1s{s1 udent_1i  st)      :
     result_list = [ ]

     for student in student list:
          stud_avg = calculate_total_average(student)
          result list.append(stud avg)
     weturn get_average(result_list)

 # Student List consisting the

s tud ent s = [ s1, s 2, s 3, s4, s,5        s,6   s 7 ]
# /terote through the students List and caLcuLate their respective average aarks and Letter yrode
TOP 1 IN StMg£'FIt S :
     print(i["name"])
     pr1nt("          "
     pr1nt("Average marks of gs 1s : is " g(i [ "naive" ], calculate total average(1) ) )

     pr1nt("Letter' Grade oT is is : Ss" g(1["name " ] , ass1gn letter rade (calculate total average(i) )) )

     pr1nt()


# CaLcuLate the average of uhoLe cLass S
class av = class averageis(students)

pr1nt( "Class Average 1s gs" g(class av))
pr1nt(" Letter Grade url the class is is " g(a3sign letter grade(class av)))
```

OUTPUT:

```
student 1
--------
Average marks of student 1 is : 72.79
Letter Grade of student 1 is : C

student 2
--------
Average marks of student 2 is : 75.962
Letter Grade of student 2 is : C

student 3
--------
Average marks of student 3 is : 75.775
Letter Grade of student 3 is : C

student 4
--------
Average marks of student 4 is : 48.356
Letter Grade of student 4 is : E

student 5
--------
Average marks of student 5 is : 57.26
Letter Grade of student 5 is : E

student 6
--------
Average marks of student 6 is : 64.15
Letter Grade of student 6 is : D

student 7
--------
Average marks of student 7 is : 67.55000000000001
Letter Grade of student 7 is : D

Class Average is 65.97757142857144
Letter Grade of the class is D
```

Discussion (PRE-LAB Session and POST-LAB Session):

Pre - Lab:

1. How to compute the average of numbers in python.
2. Sample for dictionary creation.
3. Display function for dictionary data.

Post- Lab:

1. Write a sample to append the data in the existing dictionary.
2. How to modify the value/Key value in the dictionary.

LAB OUPUTS:1.

```
#RA1811004910099
def avg(l):
    return sum(l)/len(l)
l=[10,8,6,5,4,3]
average_no=avg(l)
print("Average of the no.'s in the list=%0.2f"%round(average_no,2))
```

```
Average of the no.'s in the list=6.00
```

2.

```
#RA1811004010099
dictionary={'a':45,'b':78,'c':34}
print(dictionary)
```

```
{'a': 45, 'b': 78, 'c': 34}
```

3.

```
#RA1811004010099
dic={1:'apple',2:'banana',3:'grapes',4:'cars'}
def display(dic):
    for i,j in dic.items():
        print(i,":",j)
display(dic)
```

```
1 : apple
2 : banana
3 : grapes
4 : cars
```

Results and Conclusion:

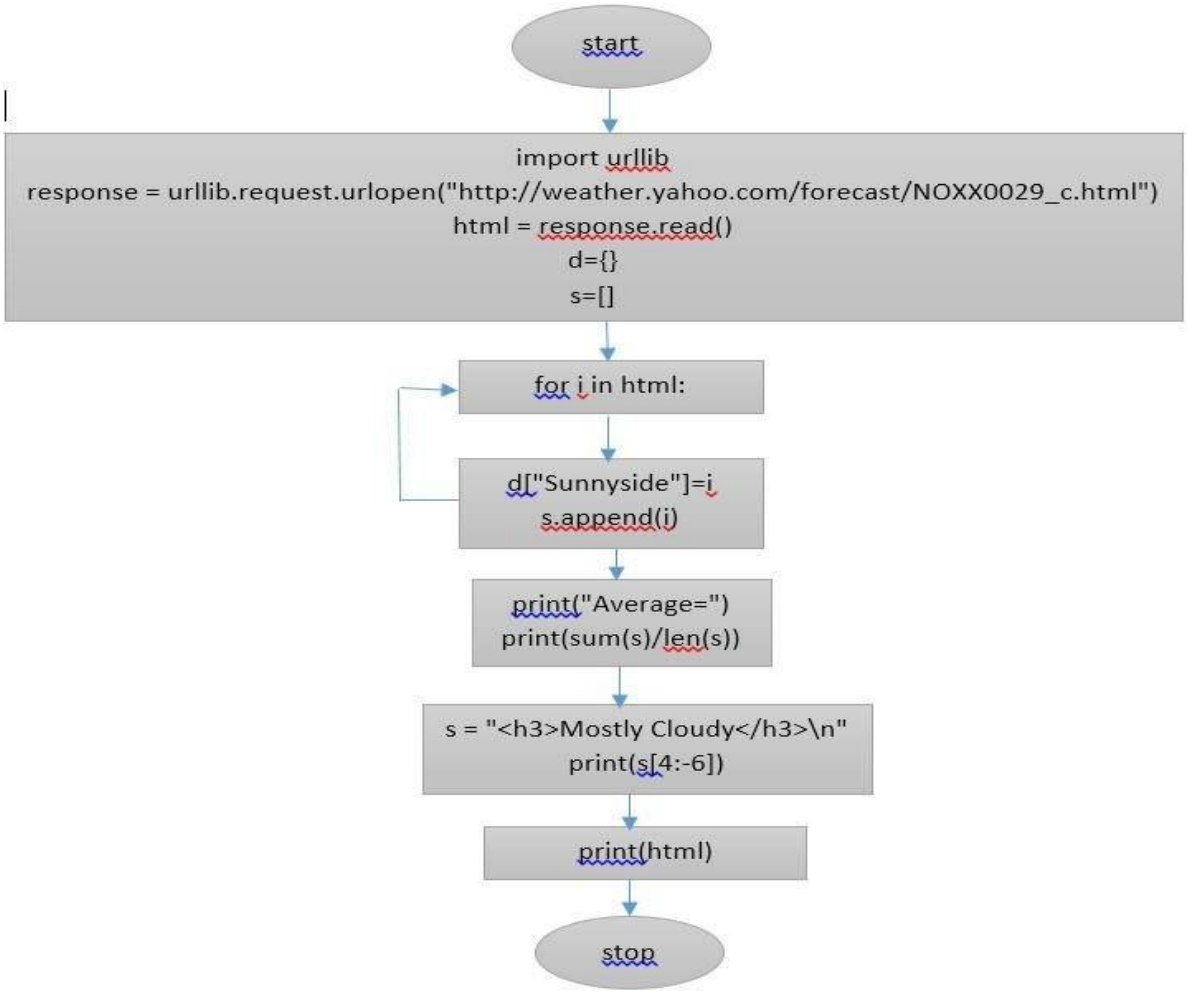
The above programs are performed and executed properly and the desired output is obtained.

8. READING THE WEB TEMPERATURE TEXT FILE INTO DICTIONARY AND COMPUTING AVERAGE TEMPERATURE

Aim

TO READ THE WEB TEMPERATURE TEXT FILE INTO DICTIONARY AND COMPUTING AVERAGE TEMPERATURE

Flowchart:



INPUT TABLE:

LIBRARIES USED	VARIABLE TO STORE DATA OF WEB PAGE	VARIABLE TO READ WEB PAGE DATA
urllib	response = urllib.request.urlopen("http://weather.yahoo.com/forecast/NOXX0029_c.html")	html = response.read()

Software used:

- Anacondanavigator

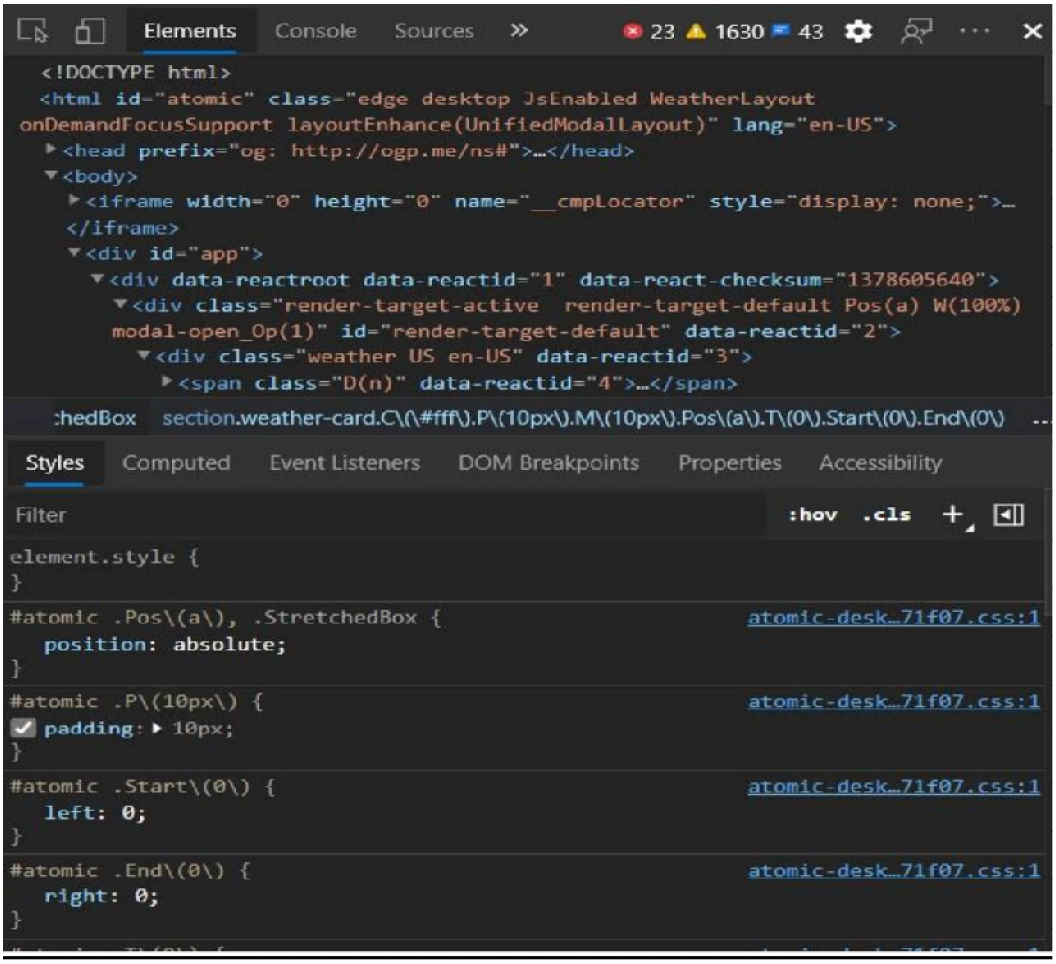
- Jupyter Notebook

PROCEDURE:

1. Open the Jupyter notebook from the Anaconda Navigator.
2. Import urllib.request to read the data from the URL and printit.
3. Open a connection to a URL using urllib and store the data in textfile.
4. Define the main function and declare thevariable.
5. Then call the URLopen function on the URL liblibrary.
6. Read the response from the page and store in html.
7. Create an empty dictionaryd.
8. Create an empty list s to store thevalues.
9. Print the averagetemperature.
10. Print the HTML of thepage.
11. Run the program to get the desiredoutput.

Program and Input Data

Input Web Page Data:



CODE:

Part 1:

```
import urllib
response=urllib.request.urlopen("http://weather.yahoo.com/forecast/NOXX0029_c.html")
html = response.read()
d={}
s=[]
for i in html:
    d["Sunnyside"]=i
    s.append(i)
print("Average=")
print(sum(s)/len(s))
```

Part 2:

s = "<h3>Mostly
Cloudy</h3>\n" print(s[4:-6])

Part 3:

import urllib
response=urllib.request.urlopen("http://weather.yahoo.com/forecast/NOXX0029_c.html")
html = response.read()
print(html)

Code and Output :

Part 1:

```
#RA181004010099
import urllib
response = urllib.request.urlopen("http://weather.yahoo.com/forecast/NOXX0029_c.html")
html = response.read()
d={}
s=[]
for i in html:
    d["Sunnyside"]=i
    s.append(i)
print("Average=")
print(sum(s)/len(s))
```

```
Average=
83.418204603470004
```

Part2:

```
#RA181004010099
s = "<h3>Mostly Cloudy</h3>\n"
print(s[4:-6])
```

```
Mostly Cloudy
```

Part 3:

```
#RA181004010099
import urllib
response = urllib.request.urlopen("http://weather.yahoo.com/forecast/NOXX0029_c.html")
html = response.read()
print(html)
```

```
b'<!DOCTYPE html><html id="atomic" class="Nojs desktop" lang="en-US"><head prefix="og: http://ogp.me/ns#"><script>window.perf
ormance && window.performance.mark && window.performance.mark(\'PageStart\');</script><meta charset="utf-8"/><meta http-equiv
="x-dns-prefetch-control" content="on"/><meta name="apple-itunes-app" content="app-id=628677149"/><meta name="render-optimize
-policy" content="first-screen-advance;enable"/><meta name="referrer" content="unsafe-url"/><meta name="oath:guce:consent-hos
t" content="guce.yahoo.com"/><meta name="twitter:description" content="View the latest weather forecasts, maps, news and aler
ts on Yahoo Weather. Find local weather forecasts for Sunnyvale, United States throughout the world"/><meta name="twitter:tit
le" content="Sunnyvale, United States - Weather Forecasts | Maps | News - Yahoo Weather"/><meta name="description" content="V
iew the latest weather forecasts, maps, news and alerts on Yahoo Weather. Find local weather forecasts for Sunnyvale, United
States throughout the world"/><meta name="keywords" content="weather forecasts, weather reports, weather maps, Sunnyvale, Uni
ted States weather forecast, worldwide weather forecast, local weather forecast, 5 day forecasts, 10 day forecasts, news, wea
ther alerts, yahoo weather, yahoo"/><meta property="al:android:package" content="com.yahoo.mobile.client.android.weather"/><m
eta property="og:description" content="View the latest weather forecasts, maps, news and alerts on Yahoo Weather. Find local
weather forecasts for Sunnyvale, United States throughout the world"/><meta property="og:title" content="Sunnyvale, United St
ates - Weather Forecasts | Maps | News - Yahoo Weather"/><link rel="dns-prefetch" href="//s.yimg.com"/><link rel="dns-prefetc
h" href="//mbp.yimg.com"/><link rel="dns-prefetch" href="//geo.query.yahoo.com"/><link rel="dns-prefetch" href="//consent.cm
p.oath.com"/><link rel="dns-prefetch" href="//csc.beap.bc.yahoo.com"/><link rel="dns-prefetch" href="//beap.gemini.yahoo.co
m"/><link rel="dns-prefetch" href="//yep.video.yahoo.com"/><link rel="dns-prefetch" href="//video-api.yql.yahoo.com"/><link r
el="dns-prefetch" href="//yrlas.blrll.com"/><link rel="dns-prefetch" href="//shim.blrll.com"/><link rel="preconnect" href="//
s.yimg.com"/><link rel="preconnect" href="//mbp.yimg.com"/><link rel="preconnect" href="//geo.query.yahoo.com"/><link rel="pr
econnect" href="//consent.cmp.oath.com"/><link rel="preconnect" href="//csc.beap.bc.yahoo.com"/><link rel="preconnect" href
```

Discussion (PRE-LAB Session and POST-LAB Session) PRELAB:

1. What is the significance of urllib package.
2. Syntax for importing the web page data in to python program.

POSTLAB:

1. How to get the response of the average temperature in form of html. given an sample.

RESULT:

Reading the web temperature text file into dictionary and computing average temperature is performed and the desired output is fetched and executed.

9. Programming on class

Aim:

9.1: Defining a class for creating bank account profile for a person.

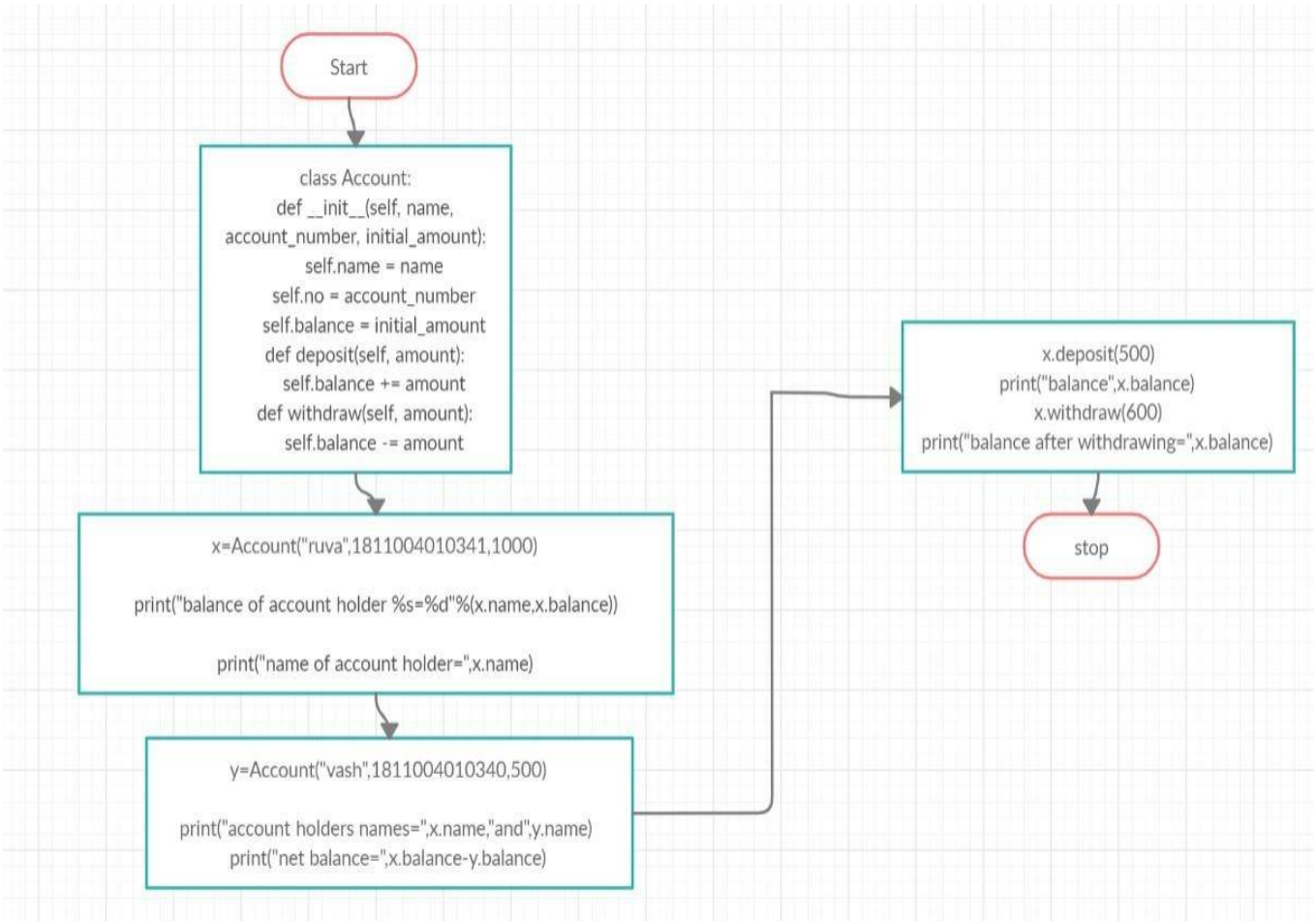
9.2: Defining a class for creating personal details for a person.

Defining a class for creating bank account profile for a person:

Software used:

- Anacondanavigator
- Jupyternotebook

FLOWCHART:



Procedure:

1. Open Anaconda Navigator and launch Jupyter Notebook.
2. Create a Class Account.
3. Define a function _init_ that helps in getting values: name, account number and initial amount for the class.
4. Define a function deposit that calculates the balance as balance + deposit amount
5. Define a function withdraw that calculates the balance as balance – withdrawal amount.
6. Call the function _init_ using object of class by passing parameters name, account number and initial amount through print function.
7. Call the function deposit using object of class by passing parameter amount through print function.
8. Call the function withdraw using object of a class by passing parameter amount through print function.
9. Calculate the final balance and print it.
10. Run the program to get the result.

CODE:

```
class Account:

   definit(self, name, account_number,
initial_amount):
      self.name = name
      self.no = account_number
      self.balance =
      initial_amount
   def deposit(self,
      amount): self.balance
      += amount
   def withdraw(self,
      amount): self.balance -
      = amount
x=Account("ruva",1811004010341,1000)
print("balance of account holder %s=%d"%(x.name,x.balance))
print("name of account holder=",x.name)
y=Account("vash",1811004010340,500)
print("account holders
names=",x.name,"and",y.name) print("net
balance=",x.balance-y.balance) x.deposit(500)
print("balance",x.balanc
e) x.withdraw(600)
print("balance after withdrawing=",x.balance))
```

OUTPUT:

```
#RA1811004010099
class Account:
    def __init__(self, name, account_number, initial_amount):
        self.name = name
        self.no = account_number
        self.balance = initial_amount
    def deposit(self, amount):
        self.balance += amount
    def withdraw(self, amount):
        self.balance -= amount


x=Account("ruva",1811004010341,1000)

print("balance of account holder %s=%d"%(x.name,x.balance))

print("name of account holder=",x.name)

y=Account("vash",1811004010340,500)

print("account holders names=",x.name,"and",y.name)
print("net balance=",x.balance-y.balance)
x.deposit(500)
print("balance",x.balance)
x.withdraw(600)
print("balance after withdrawing=",x.balance)
```

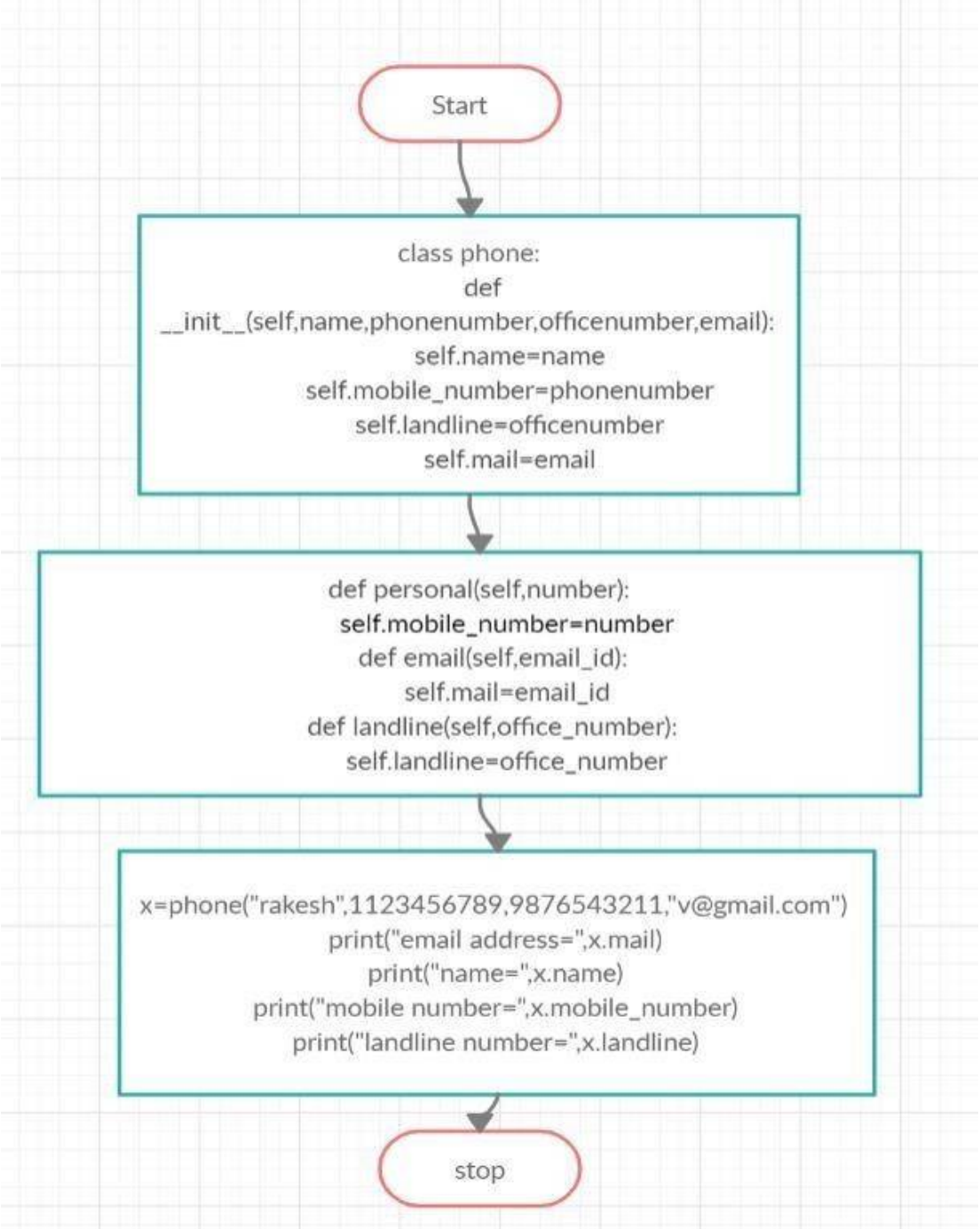
```
balance of account holder ruva=1000
name of account holder= ruva
account holders names= ruva and vash
net balance= 500
balance 1500
balance after withdrawing= 900
```

Defining a class for creating personal details for a person.

Software used:

- Anaconda navigator
- Jupyter notebook

FLOWCHART:



PROCEDURE:

1. Open Anaconda Navigator and launch JupyterNotebook.
2. Create a ClassAccount.
3. Define a function _init_ that helps in getting values: name, phone number, office number and email for theclass.
4. Define a function personal that stores the mobilenumber
5. Define a function email that stores the emailid
6. Define a function landline that stores the officenumber
7. Call the function _init_ using object of class by passing parameters name, phone number, office number andemail.
8. Pass arguments for the class phone through a variable x to the _init_function.
9. Call the functions personal, email, and landline through a classobject.
10. Print all the information using printfunction.
11. Run the program to get theresult.

Program CODE:

```
class phone:
  definit(self,name,phonenumber,officenumber,email):
       self.name=name
       self.mobile_number=phonenumb
       er self.landline=officenumber
       self.mail=email
  def personal(self,number):
    self.mobile_number=numb
    er
  def
    email(self,email_id):
    self.mail=email_id
  def
    landline(self,office_number)
    :
    self.landline=office_number
x=phone("rakesh",1123456789,9876543211,"v@gmail.c
om") print("email address=",x.mail)
print("name=",x.name)
print("mobile
number=",x.mobile_number)
print("landline number=",x.landline)
```

OUTPUT:

```
#RA1811004010099
class phone:
    def __init__(self,name,phonenumber,officenumber,email):
            self.name=name
            self.mobile_number=phonenumber
            self.landline=officenumber
            self.mail=email
    def personal(self,number):
        self.mobile_number=number
    def email(self,email_id):
        self.mail=email_id
    def landline(self,office_number):
        self.landline=office_number
x=phone("rakesh",1123456789,9876543211,"v@gmail.com")
print("email address=",x.mail)
print("name=",x.name)
print("mobile number=",x.mobile_number)
print("landline number=",x.landline)
```

```
email address= v@gmail.com
name= rakesh
mobile number= 1123456789
landline number= 9876543211
```

Discussion (PRE-LAB Session and POST-LAB Session):

Pre-Lab:

1. what are functions and variables called inpython?
2. Define a class in python with anexample?
3. what is the significant __init__ in classprogram?

Post-Lab:

1. Draw the UML diagram for the program done. 2. How constructor is useful in class program.

Results and Conclusion:

The program for defining a class for creating bank account profile for a person and defining a class for creating personal details for a person is executed and performed successfully and the desired output is achieved.

10. REAL CARD GAME : RANDOM NUMBER GENERATION

Aim:

MAKE A DECK OFCARDS.
DEAL AHAND.
DEAL HANDS FOR A SET OFPLAYERS.
FINDING THE NO.OF SAME KINDS OF CARDS FOR A SINGLEHAND.
FINDINGTHENO.OFPAIRSANDNO.OFSAMEKINDANDCARDSOFSAMESUITEOF HAND.

SOFTWARE USED:

- Anaconda Navigator.
- Jupyter Notebook.

PROCEDURE (For 10.1, 10.2, 10.3):

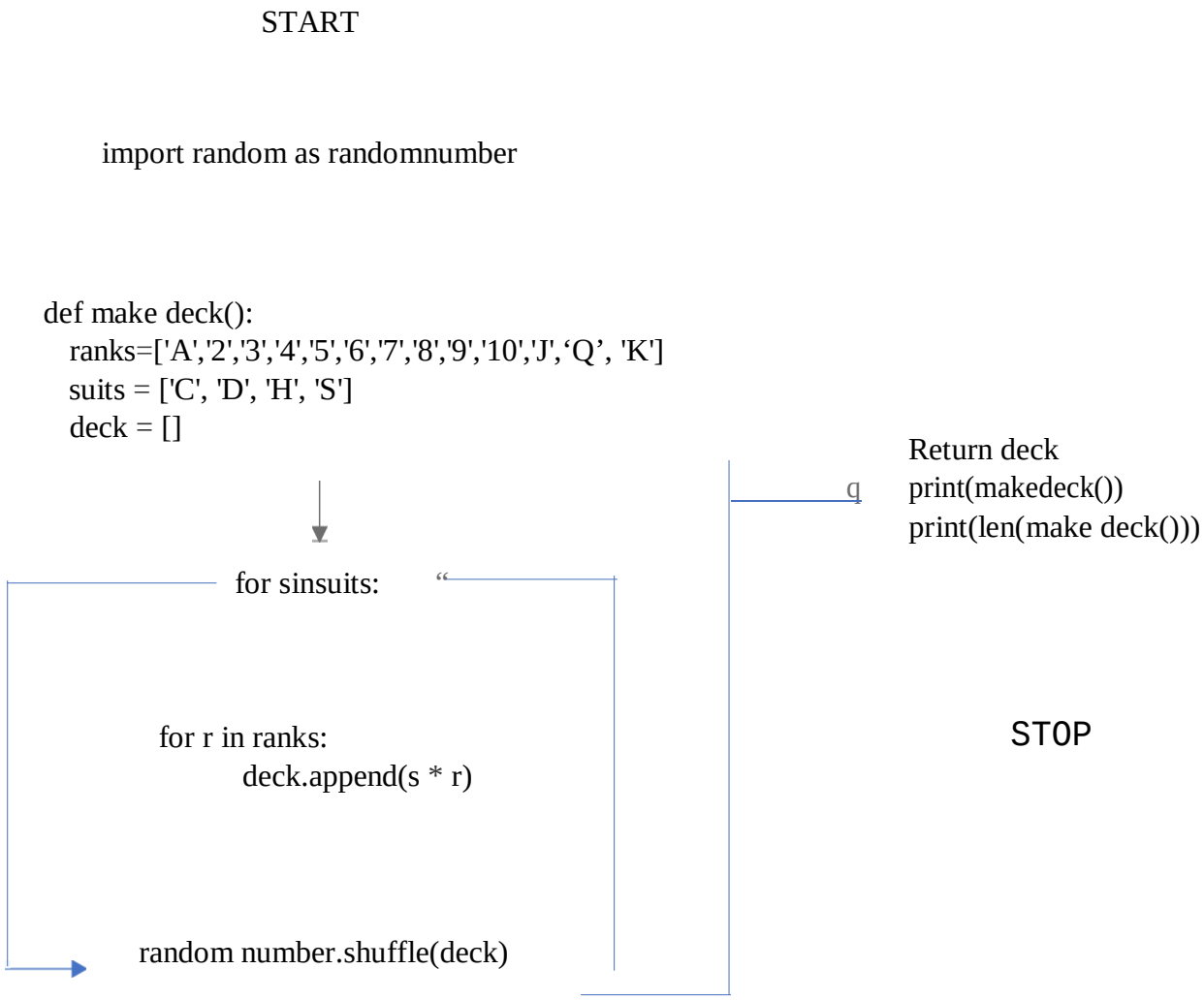
1. Open Anaconda Navigator and launch JupyterNotebook.
2. Define a function make_deck to generate all possible combinations ofcards.
3. Return the value of the function after using the shuffle to shuffle thecards.
4. Print the values thus obtained after shuffling by functioncall.
5. Initialise a variable deck=make_deck to store result of functionmake_deck.
6. Define a function deal_hand to pick a deal of hand full ofcards.
7. Return the value of the function to display hand in cards anddeck.
8. Print the values thus obtained after function call.
9. Define a function deal to show the cards received by eachperson.
10. Return the value of the function to display the card he owns and card left on thedeck.
11. Repeat till all the players get the desired number ofcards.
12. Print the values obtained after function call passing arguments cards per playerand number ofplayers.
13. Run the programs to get theresult.

PROCEDURE (For 10.4, 10.5):

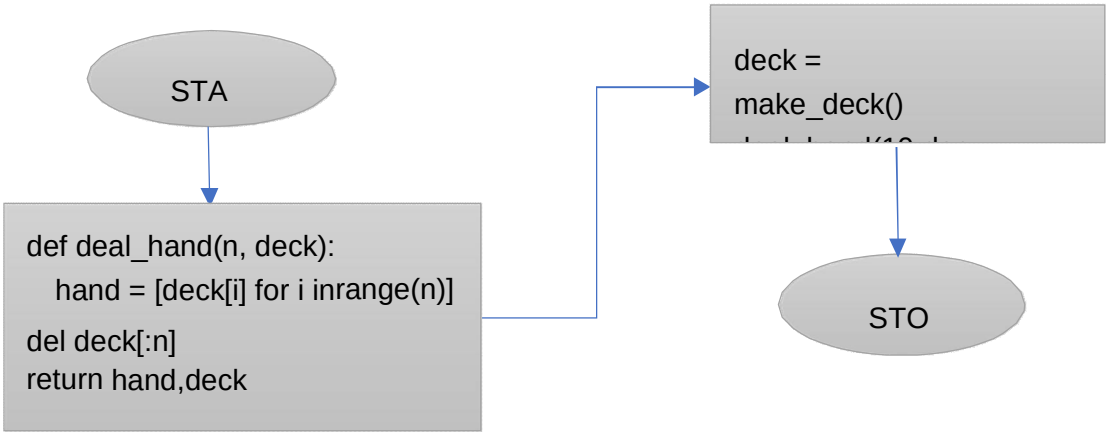
1. Open Anaconda Navigator and launch Jupyter Notebook.
2. Define function same_rank to check if cards belong to same rank.
3. Initialise the counter value andalready_counted.
4. For every rank check if they are not already counted. If not compare it with the arguments passed.
5. If condition is satisfied then increment the counter value.
6. Return the counter value.
7. Call the function using same_rank(hand,2) (by pasing parameters) and print it.
8. Define function same_Suit to check the number of combinations of same suit.
9. Initialize the suits and counter values.
10. For every suit value, check if count value is greater than 1. If it is greater than one then assign it to counter[suit].
11. For every hand in players, use print function and find the combination of suits this obtained.
12. Run the programs to get the result.

FLOWCHARTS:

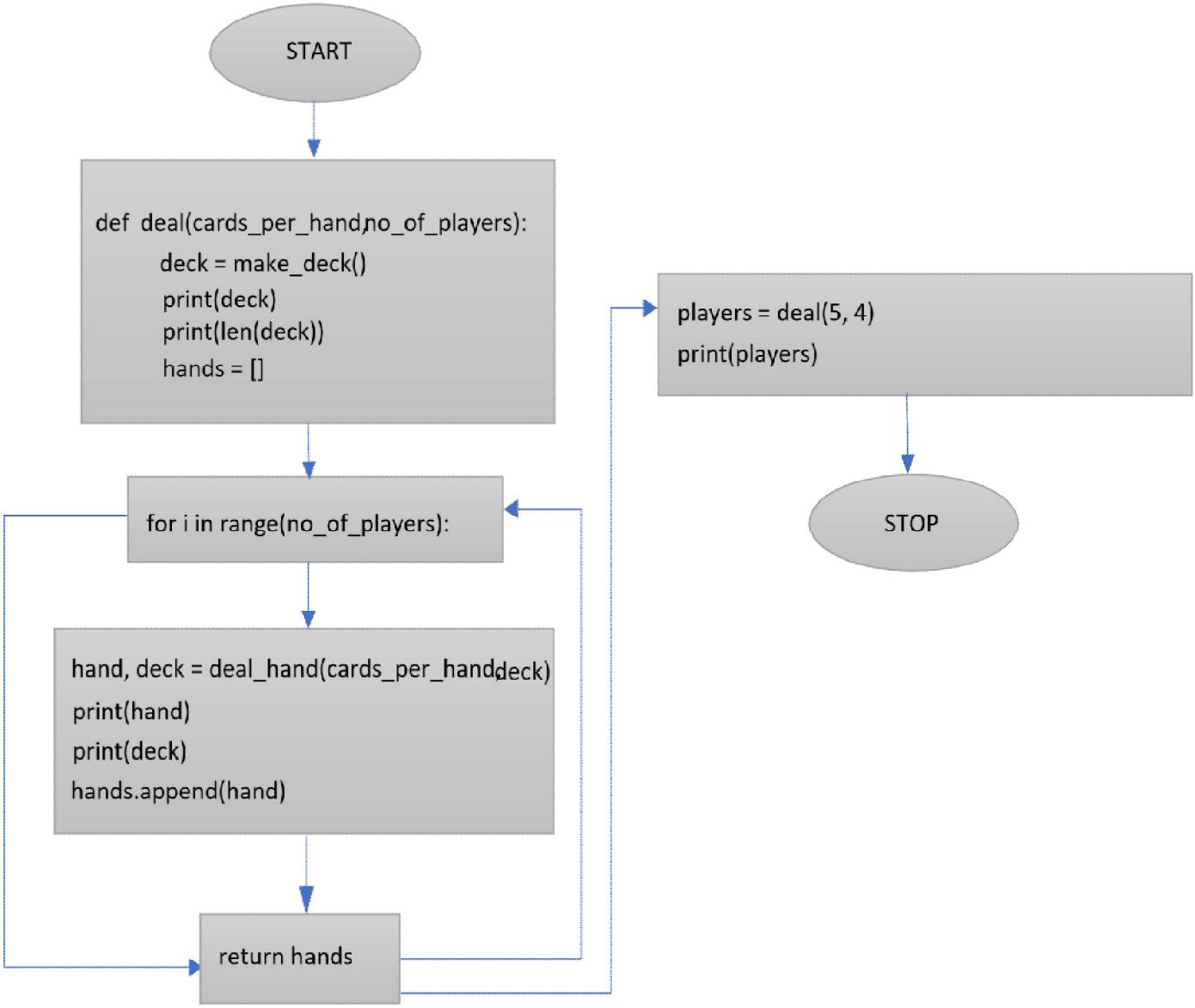
MAKE A DECK OFCARDS



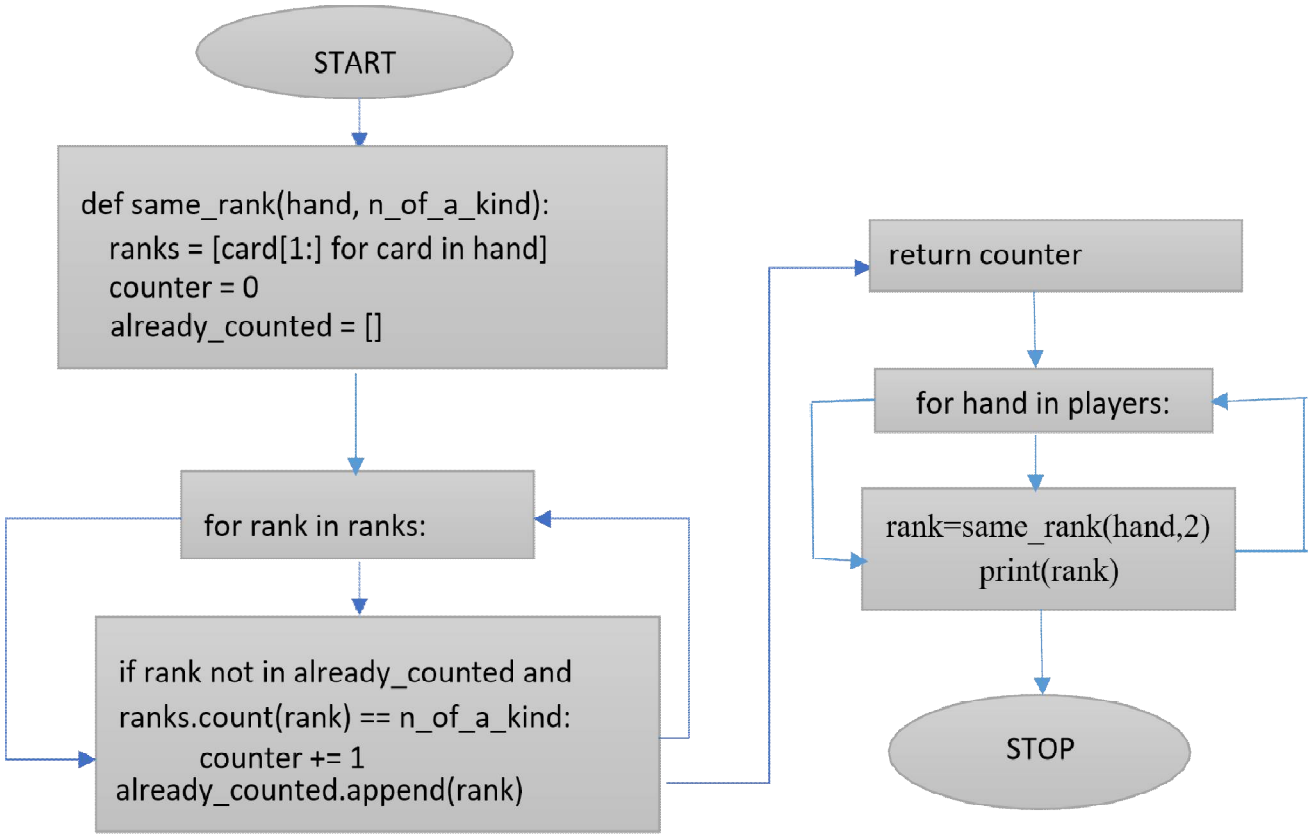
DEAL A HAND



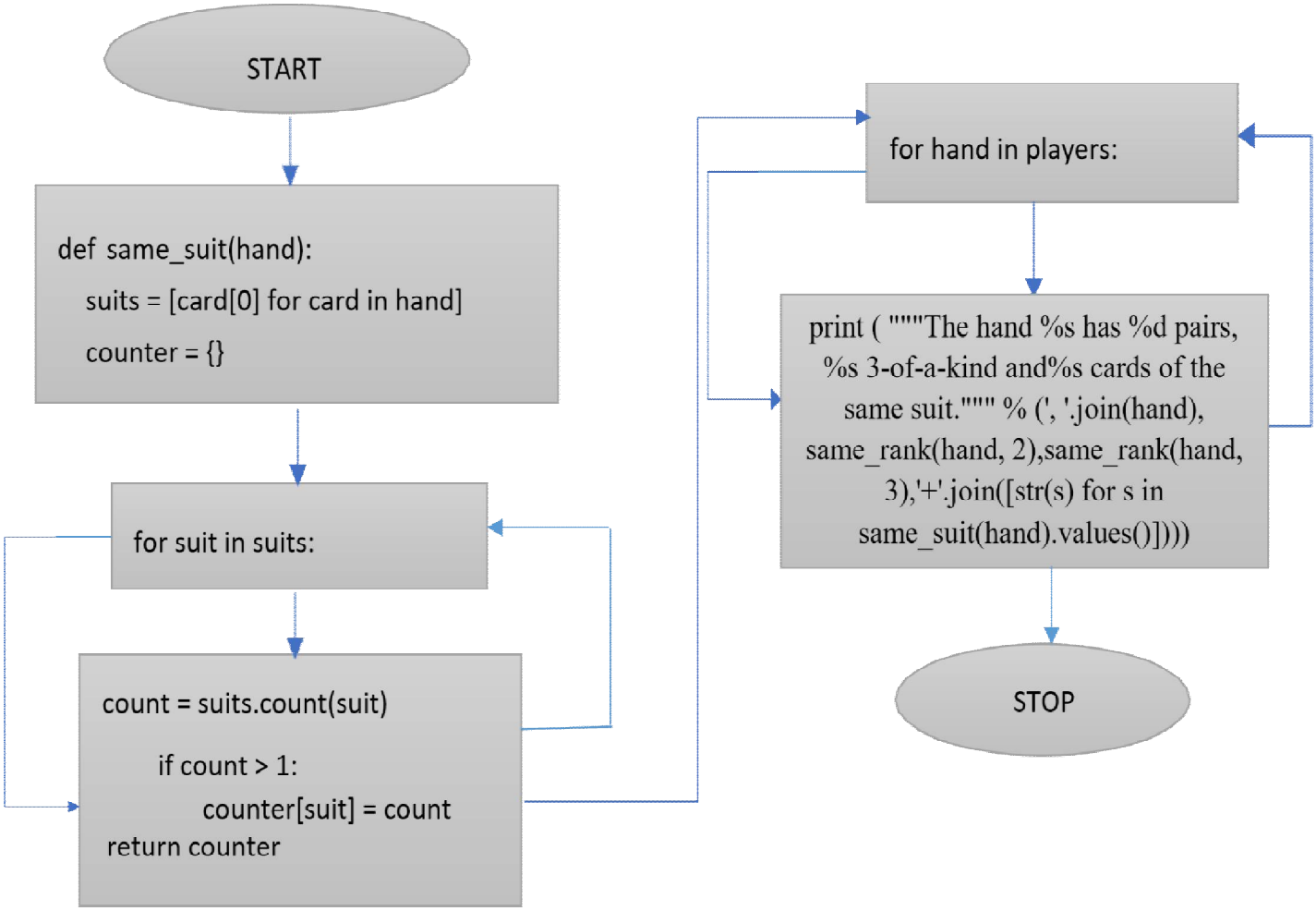
DEAL HANDS FOR A SET OFPLAYERS



FINDING THE NO.OF SAME KINDS OF CARDS FOR A SINGLE HAND



FINDING THE NO.OF PAIRS AND NO.OF SAME KIND AND CARDS OF SAME SUITE OFHAND



INPUTS TABLE:

LIBRARY USED	FUNCTIONS USED
import random as random_number	make_deck , deal_hand , deal , same_rank , same_suit

Program and Input Data:

10.1 MAKEADECKOFCARDS CODE:

```
import random as
random_number def
make_deck():
   ranks = ['A', '2', '3', '4', '5', '6', '7', '8', '9', '10', 'J', 'Q',
   'K'] suits = ['C', 'D', 'H', 'S']
   deck = []
   for s in suits: # s0, s1, s2, s3 - each have 13 cards - shapes are clubs, spade,
      diamond, heart for r in ranks:
         deck.append(s + r)
      random_number.shuffle(dec
      k)
   return deck
print(make_deck())

print(len(make_deck()))
print(len(make_deck()))
```

10.2 DEAL A HAND CODE:

```
def deal_hand(n, deck):
   hand = [deck[i] for i in
   range(n)] del deck[:n]
   return
hand,deck
deck=make_de()
deal_hand(10,deck)
```

DEAL HANDS FOR A SET OFPLAYERS

CODE:

```
def deal(cards_per_hand,
    no_of_players): deck = make_deck()
    #52 cards in deck print(deck)
   print(len(deck))
    hands = [] #empty array--> count=0; consists of nothing
    for iinrange(no_of_players):     # 4players(i=0,1,2,3 --->
       4times hand, deck =deal_hand(cards_per_hand,deck)
                                        #5,52 print(hand)
      print(deck)
       hands.append(hand)   #adding to the empty array-- store hand
values to hands array
   return hands
players =deal(5,4)            #hand contains 5 cards, 4
players print(players)
```

FINDING THE NO. OF SAME KINDS OF CARDS FOR A SINGLE HAND CODE:

```
Defsame_rank(hand,
   n_of_a_kind): ranks = [card[1:]
   for card in hand] counter = 0
   already_counted = []
   for rank in ranks:
```

```
        if rank not in already_counted and ranks.count(rank) == n_of_a_kind: counter+= 1
        already_counted.append(rank)
    return counter
for hand in players:
    rank=same_rank(hand,2)
    print(rank)
```

FINDING THE NO. OF PAIRS AND NO. OF SAME KIND AND CARDS OF SAME SUITE OF HAND

CODE:

```
def same_suit(hand):
    suits = [card[0] for card in hand]
    counter = {} # counter[suit] = how many cards of
    suit for suit in suits:
        count =
        suits.count(suit) if
        count > 1:
            counter[suit] = count
        returncounter
for hand inplayers:
    print ( """The hand %s has %d pairs, %s 3-of-a-kind and%s cards of the same suit."""" % (',
'.join(hand), same_rank(hand, 2),same_rank(hand, 3),'+'.join([str(s) for s in
same_suit(hand).values()])))
```

Discussion (PRE-LAB Session and POST-LAB Session):

PRE LAB:

1. Write the correct function from the following list to get the random integer between 99 to 200, which is divisible by 3?

POST LAB:

1. To generate a random float number between 20.5 to 50.5, which function of a random module I need to use ?

OUTPUT:

MAKE A DECK OFCARDS

```
#RA1811004010099
import random as random_number
def make_deck():
    ranks = ['A', '2', '3', '4', '5', '6', '7', '8', '9', '10', 'J', 'Q', 'K']
    suits = ['C', 'D', 'H', 'S']
    deck = []
    for s in suits: # s0, s1, s2, s3 - each have 13 cards - shapes are clubs, spade, diamond, heart
        for r in ranks:
            deck.append(s + r)
        random_number.shuffle(deck)
    return deck
print(make_deck())

print(len(make_deck()))
```

```
['H8', 'C6', 'S2', 'S9', 'C9', 'C2', 'C8', 'D3', 'C3', 'C4', 'H5', 'C10', 'HA', 'DQ', 'SK', 'H10', 'S7', 'S5', 'H2', 'HK', 'D
6', 'H9', 'SA', 'CQ', 'D4', 'D8', 'DA', 'S6', 'DJ', 'H6', 'C7', 'S4', 'S3', 'CJ', 'H4', 'SQ', 'H7', 'SJ', 'H3', 'D10', 'D2', 'H
J', 'D7', 'CK', 'D9', 'DK', 'D5', 'C5', 'S8', 'CA', 'HQ', 'S10']
52
```

DEAL AHAND

```
#RA1811004010099
def deal_hand(n, deck):
    hand = [deck[i] for i in range(n)]
    del deck[:n]
    return hand,deck
deck=make_deck()
deal_hand(10,deck)
```

```
(['CJ', 'C7', 'SA', 'DQ', 'C8', 'C9', 'CA', 'D9', 'C2', 'S4'],
 ['S3',
  'C5',
  'D4',
  'C6',
  'DK',
  'C4',
  'H2',
  'S2',
  'D6',
  'S9',
  'H6',
  'H5',
  'DJ',
  'SQ',
  'H10',
  'H4',
  'CQ',
  'D8',
  'HJ',
  'SJ',
  'D5',
  'CK',
  'S7',
  'H7',
  'C10'
```

DEAL HANDS FOR A SET OF PLAYERS

```
#RA1811004010099
def deal(cards_per_hand, no_of_players):
    deck = make_deck()  #52 cards in deck
    print(deck)
    print(len(deck))
    hands = [] #empty array--> count=0; consists of nothing
    for i in range(no_of_players):        # 4players(i=0,1,2,3 ---> 4times
        hand, deck = deal_hand(cards_per_hand, deck)     #5,52
        print(hand)
        print(deck)
        hands.append(hand)       #adding to the empty array-- store hand values to hands array
    return hands
players = deal(5, 4)                  #hand contains 5 cards, 4 players
print(players)
```

```
['D4', 'S4', 'D8', 'S8', 'DK', 'C4', 'S9', 'H7', 'S6', 'DA', 'CJ', 'C8', 'H6', 'HK', 'H10', 'S3', 'SK', 'C2', 'D2', 'H5', 'SJ',
'H2', 'D5', 'D10', 'CA', 'D3', 'D9', 'DJ', 'CQ', 'HQ', 'DQ', 'S2', 'D6', 'H8', 'HJ', 'S10', 'CK', 'S7', 'C3', 'C10', 'SQ', 'S
5', 'HA', 'H4', 'C6', 'H9', 'C7', 'D7', 'C9', 'C5', 'SA', 'H3']
52
['D4', 'S4', 'D8', 'S8', 'DK']
['C4', 'S9', 'H7', 'S6', 'DA', 'CJ', 'C8', 'H6', 'HK', 'H10', 'S3', 'SK', 'C2', 'D2', 'H5', 'SJ', 'H2', 'D5', 'D10', 'CA', 'D
3', 'D9', 'DJ', 'CQ', 'HQ', 'DQ', 'S2', 'D6', 'H8', 'HJ', 'S10', 'CK', 'S7', 'C3', 'C10', 'SQ', 'S5', 'HA', 'H4', 'C6', 'H9',
'C7', 'D7', 'C9', 'C5', 'SA', 'H3']
['C4', 'S9', 'H7', 'S6', 'DA']
['CJ', 'C8', 'H6', 'HK', 'H10', 'S3', 'SK', 'C2', 'D2', 'H5', 'SJ', 'H2', 'D5', 'D10', 'CA', 'DJ', 'D9', 'DJ', 'CQ', 'HQ', 'D
Q', 'S2', 'D6', 'H8', 'HJ', 'S10', 'CK', 'S7', 'C3', 'C10', 'SQ', 'S5', 'HA', 'H4', 'C6', 'H9', 'C7', 'D7', 'C9', 'C5', 'SA',
'H3']
['CJ', 'C8', 'H6', 'HK', 'H10']
['S3', 'SK', 'C2', 'D2', 'H5', 'SJ', 'H2', 'D5', 'D10', 'CA', 'D3', 'D9', 'DJ', 'CQ', 'HQ', 'DQ', 'S2', 'D6', 'H8', 'HJ', 'S1
0', 'CK', 'S7', 'C3', 'C10', 'SQ', 'S5', 'HA', 'H4', 'C6', 'H9', 'C7', 'D7', 'C9', 'C5', 'SA', 'H3']
['S3', 'SK', 'C2', 'D2', 'H5']
['SJ', 'H2', 'D5', 'D10', 'CA', 'D3', 'D9', 'DJ', 'CQ', 'HQ', 'DQ', 'S2', 'D6', 'H8', 'HJ', 'S10', 'CK', 'S7', 'C3', 'C10', 'S
Q', 'S5', 'HA', 'H4', 'C6', 'H9', 'C7', 'D7', 'C9', 'C5', 'SA', 'H3']
[['D4', 'S4', 'D8', 'S8', 'DK'], ['C4', 'S9', 'H7', 'S6', 'DA'], ['CJ', 'C8', 'H6', 'HK', 'H10'], ['S3', 'SK', 'C2', 'D2', 'H
5']]
```

FINDING THE NO.OF SAME KINDS OF CARDS FOR A SINGLEHAND

```
#RA1811004010099
def same_rank(hand, n_of_a_kind):
    ranks = [card[1:] for card in hand]
    counter = 0
    already_counted = []
    for rank in ranks:
        if rank not in already_counted and ranks.count(rank) == n_of_a_kind:
            counter+= 1
        already_counted.append(rank)
    return counter
for hand in players:
    rank=same_rank(hand,2)
    print(rank)
```

```
2
0
0
1
```

FINDING THE NO. OF PAIRS AND NO. OF SAME KIND AND CARDS OF SAME SUITE OF HAND

```
#RA1811004010099
def same_suit(hand):
    suits = [card[0] for card in hand]
    counter = {} # counter[suit] = how many cards of suit
    for suit in suits:
        count = suits.count(suit)
        if count > 1:
            counter[suit] = count
        return counter
for hand in players:
    print ( """The hand %s has %d pairs, %s 3-of-a-kind and%s cards of the same suit.""" % (', '.join(hand), same_rank(hand, 2),s
```

```
The hand D4, S4, D8, S8, DK has 2 pairs, 0 3-of-a-kind and3 cards of the same suit.
The hand C4, S9, H7, S6, DA has 0 pairs, 0 3-of-a-kind and cards of the same suit.
The hand CJ, C8, H6, HK, H10 has 0 pairs, 0 3-of-a-kind and2 cards of the same suit.
The hand S3, SK, C2, D2, H5 has 1 pairs, 0 3-of-a-kind and2 cards of the same suit.
```

RESULT AND CONCLUSION:

The real card game : random number generation is executed and performed successfully and the desired output is printed.

12. Sample Games: Guessing a Number and Rolling Two Dice

Aim(s):

To Guess a Number and Rolling Two Dice and perform the program in Jupyter Notebook.

Program 1: Guessing a Number

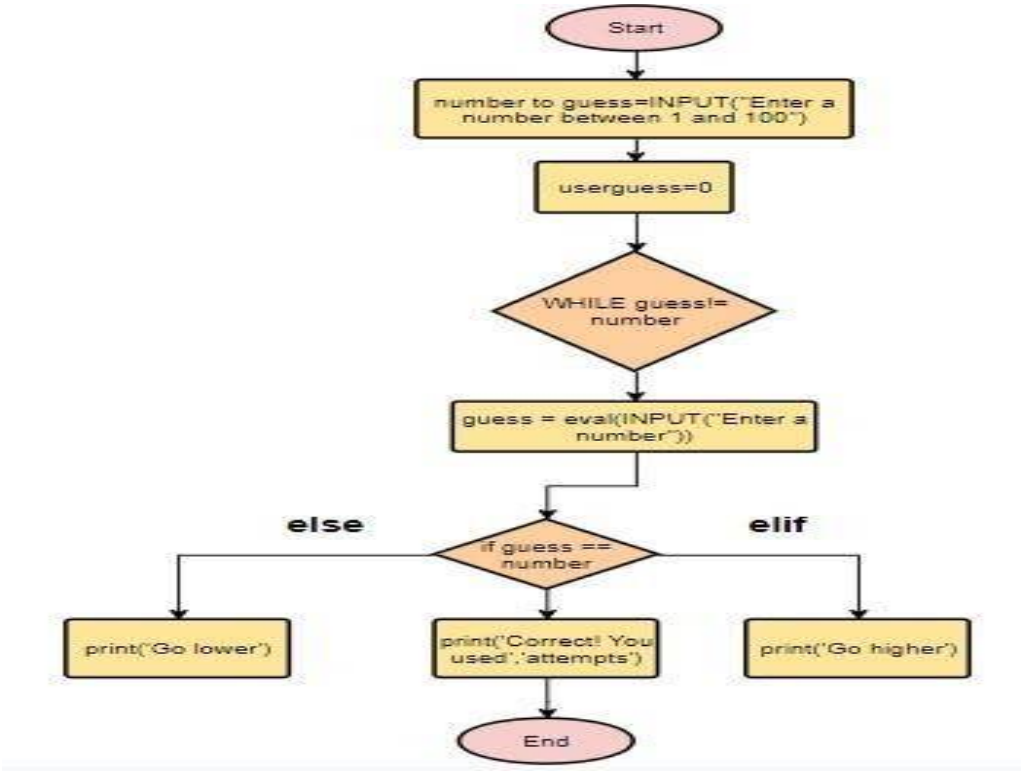
Program 2: Rolling Two Dice

Program 1: Guessing a Number

Introduction / Background:

In this problem statement we write the code to Guess a number and rolling two dice

Flowchart:



Materials

1. Anaconda Navigator
2. Jupyter Notebook

Procedure:

1. Open Anaconda Navigator and launch Jupyter Notebook,
2. Now import random as random_number
3. Define a function random_number.randint (1,100) to generate possible attempts.
4. Initialise count number of attempts to guess the number.
5. Use while loop to guess the number
6. Guess a number
7. if guess = = number. If the entered number is correct print (‘Correct !You used’,attempts,’attempts!’)
8. Break the loop elseif guess<number,print(‘Gohigher!’)
9. Else print (‘Golower!’)
10. Run the program to get the desired output.

PROGRAM AND INPUT DATA:

CODE:

```
import random as random_number
number = random_number.randint(1,100)
 attempts = 0 # count no of attempts to
guess the number guess = 0
 while guess != number:
   guess = eval(input('Guess a
number: ')) attempts += 1
    if guess == number:
  print('Correct!Youused',attempts,'atte
  mpts!') break
elif guess <
  number: print
  ('Go higher!')
else:
  print ('Go lower!')
```

OUTPUT:

```
#RA1811004010099
import random as random_number
number = random_number.randint(1, 100)
attempts = 0 # count no of attempts to guess the number
guess = 0
while guess != number:
    guess = eval(input('Guess a number: '))
    attempts += 1
    if guess == number:
        print ('Correct! You used', attempts, 'attempts!')
        break
    elif guess < number:
        print ('Go higher!')
    else:
        print ('Go lower!')
```

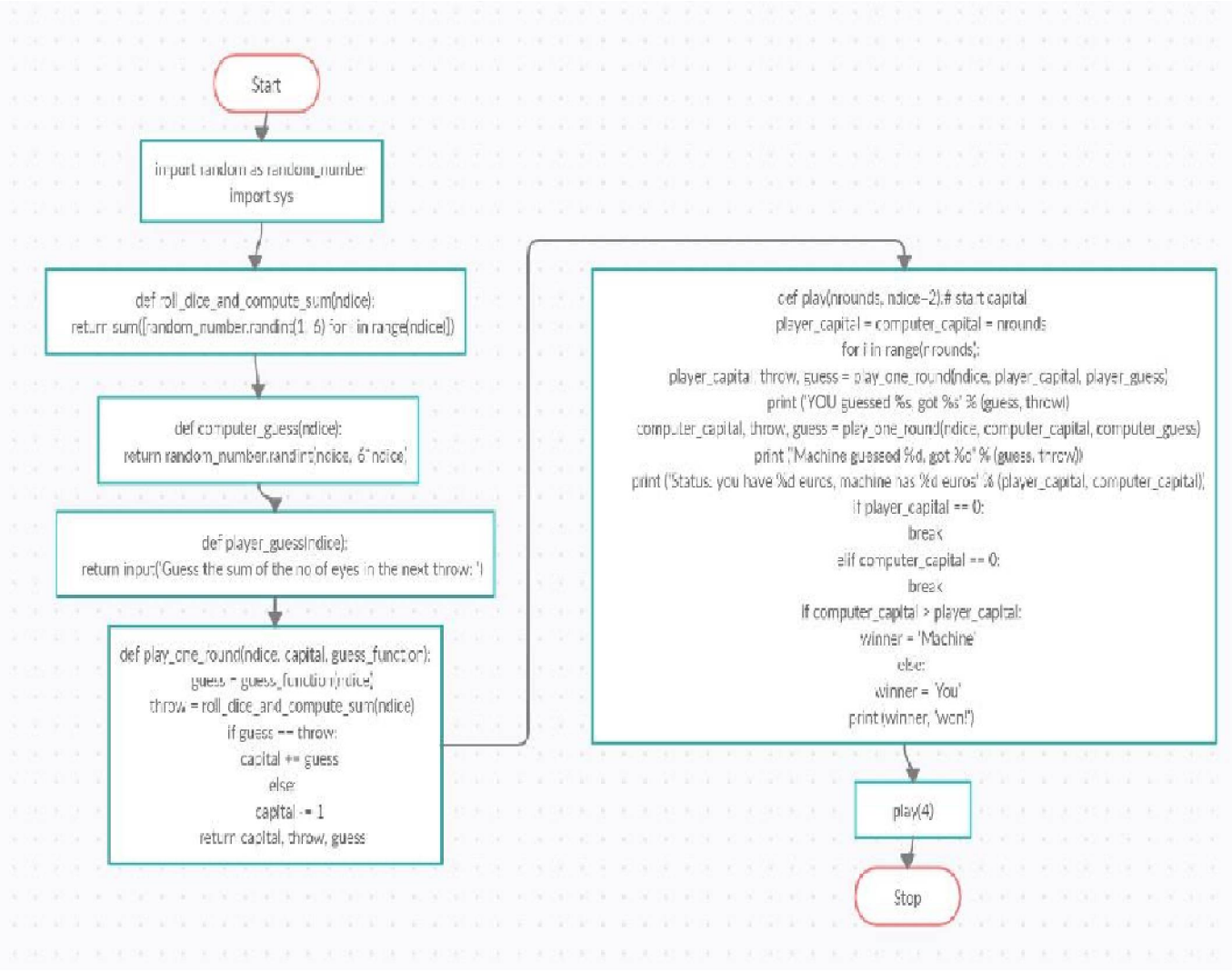
```
Guess a number: 15
Go lower!
Guess a number: 10
Go lower!
Guess a number: 8
Go higher!
Guess a number: 9
Correct! You used 4 attempts!
```

PROGRAM 2: ROLLING TWO DICE

PROCEDURE:

1. Open Anaconda Navigator and launch JupyterNotebook,
2. Now import random as random_number,importsys
3. There are three actions that we can naturally implement asfunctions:
4. (i) roll two dice and compute thesum
5. (ii) ask the player to guess the number ofeyes
6. (iii) draw the computer's guess of the number ofeyes.
7. We can now implement one round in the game for the player or thecomputer.
8. The round starts with a capital, a guess is performed by calling the right function for guessing, and the capital isupdated.
9. Here, guess_function is either computer_guess orplayer_guess.
10. With the play_one_round function we can run a number of rounds involving bothplayers
11. Run the program to get the desiredoutput.

Flowchart:



PROGRAM AND INPUT DATA:

CODE:

```
import random as

random_number import sys

def roll_dice_and_compute_sum(ndice):

   return sum([random_number.randint(1, 6) for i in

range(ndice)]) def computer_guess(ndice):
```

```
   return random_number.randint(ndice, 6*ndice)

defplayer_guess(ndice):                return input('Guess the sum
of the no of eyes in the next throw:')


def play_one_round(ndice, capital,
guess_function):  guess
=guess_function(ndice)
throw=roll_dice_and_compute_sum(ndice) if
guess ==throw:      capital += guess else:
capital -= 1

   return capital, throw, guess


def play(nrounds, ndice=2):# start capital:

   player_capital = computer_capital=nrounds for i in range(nrounds):
player_capital, throw, guess = play_one_round(ndice, player_capital,
player_guess) print ('YOU guessed %s, got %s' % (guess,throw))

      computer_capital, throw, guess = play_one_round(ndice,
computer_capital, computer_guess)

      print ('Machine guessed %d, got %d' % (guess, throw))

      print ('Status: you have %d euros, machine has %d euros' %
(player_capital, computer_capital))          if player_capital ==0:

         break      elif
computer_capital ==0:

         break          if
computer_capital>
player_capital:      winner =
'Machine'    else:

      winner = 'You'
print (winner,
'won!') play(4)
```

OUTPUT

```
#RA1811004010099
import random as random_number
import sys

def roll_dice_and_compute_sum(ndice):
    return sum([random_number.randint(1, 6) for i in range(ndice)])

def computer_guess(ndice):
    return random_number.randint(ndice, 6*ndice)

def player_guess(ndice):
    return input('Guess the sum of the no of eyes in the next throw: ')

def play_one_round(ndice, capital, guess_function):
    guess = guess_function(ndice)
    throw = roll_dice_and_compute_sum(ndice)
    if guess == throw:
        capital += guess
    else:
        capital -= 1
    return capital, throw, guess

def play(nrounds, ndice=2):# start capital:
    player_capital = computer_capital = nrounds
    for i in range(nrounds):
        player_capital, throw, guess = play_one_round(ndice, player_capital, player_guess)
        print ('YOU guessed %s, got %s' % (guess, throw))
        computer_capital, throw, guess = play_one_round(ndice, computer_capital, computer_guess)
        print ('Machine guessed %d, got %d' % (guess, throw))
        print ('Status: you have %d euros, machine has %d euros' % (player_capital, computer_capital))
        if player_capital == 0:
            break
        elif computer_capital == 0:
            break
    if computer_capital > player_capital:
        winner = 'Machine'
    else:
        winner = 'You'
    print (winner, 'won!')
```

```
play(4)
```

```
Guess the sum of the no of eyes in the next throw: 6
YOU guessed 6, got 10
Machine guessed 8, got 7
Status: you have 3 euros, machine has 3 euros
Guess the sum of the no of eyes in the next throw: 9
YOU guessed 9, got 7
Machine guessed 8, got 6
Status: you have 2 euros, machine has 2 euros
Guess the sum of the no of eyes in the next throw: 8
YOU guessed 8, got 12
Machine guessed 10, got 9
Status: you have 1 euros, machine has 1 euros
Guess the sum of the no of eyes in the next throw: 9
YOU guessed 9, got 6
Machine guessed 11, got 3
Status: you have 0 euros, machine has 0 euros
You won!
```

PRE-LAB :

1) random.seed() method is used to initialize the pseudorandom number generator. The random module uses the seed value as a base to generate a random number. If seed value is not present, it takes the system's current time. Statement is True or False ?

POST LAB :

1) Which method should I use to get 4 elements from the following list randomly
   Sample list = [20, 40, 60, 80, 100, 120, 140, 160, 180, 200]?

Result:

We successfully executed the code to Guess a number and rolling two dice and got the desired output.

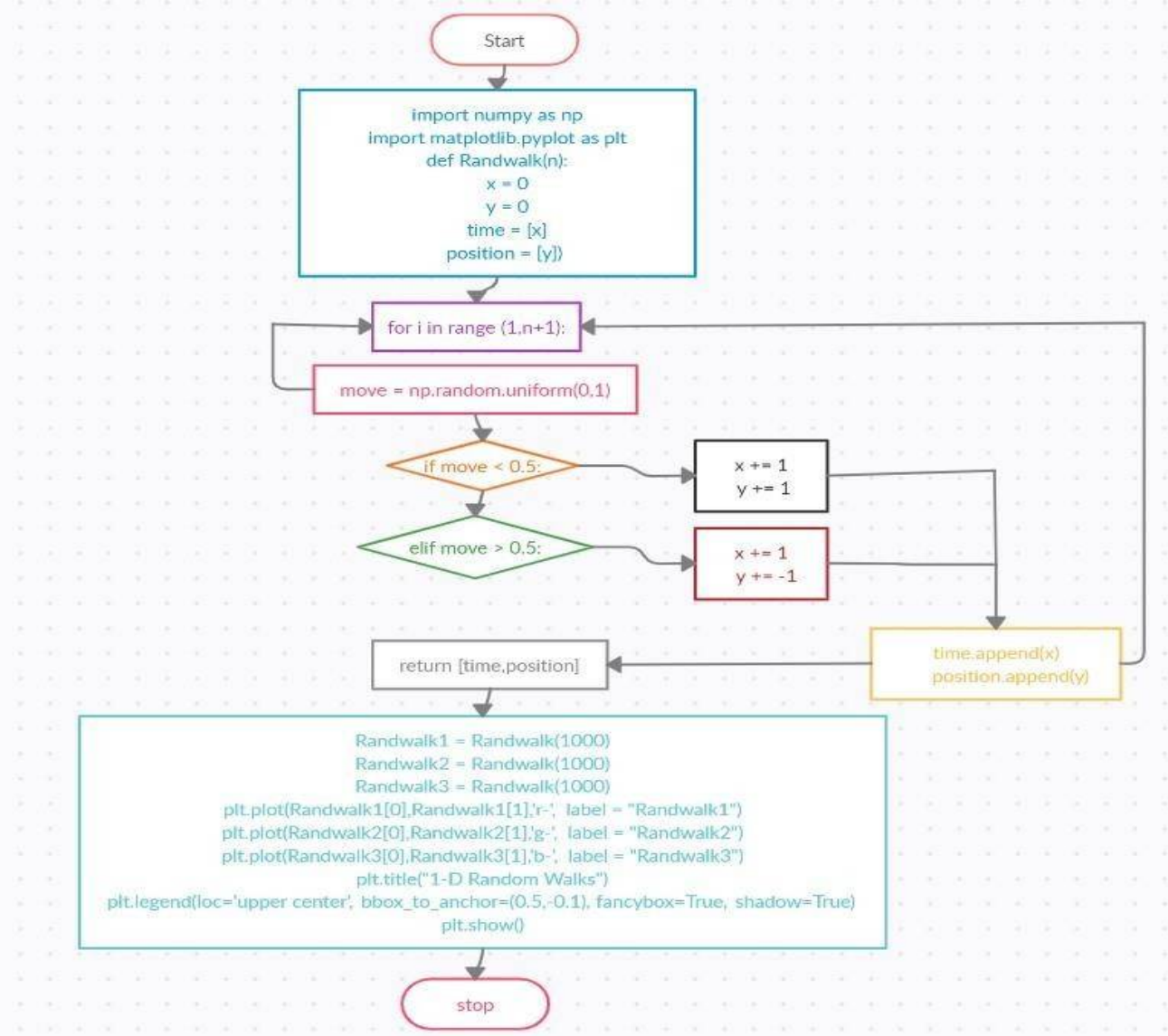
12. Random walk in one space dimensions

Aim: Random walk in one space dimensions

Introduction / Background:

In this problem statement we write the code for a random walk in one space dimensions

Flowchart:



SOFTWARE USED:

1. Anaconda Navigator
2. Jupyter Notebook

PROCEDURE:

1. Open Anaconda Navigator and launch Jupyter Notebook,
2. Now import random asrandom_number
3. import matplotlib.pyplot asplt
4. Now create a function Randwalk(n)
5. Assign 2 variables x,y for time and position and also assign MOVE with conclusions to append time and positions.
6. Give inputs like
   Randwalk1 = Randwalk(1000) Randwalk2

   = Randwalk(1000) Randwalk3 =

   Randwalk(1000)

7. Now plot those functions usingplt
8. A plot will be displayed depicting theouput.

PROGRAM AND INPUT DATA:

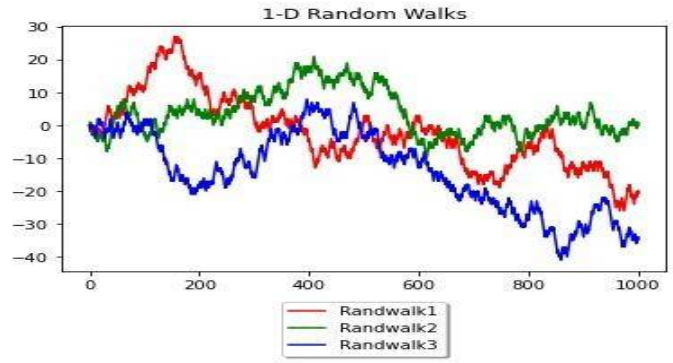
CODE:

```
import numpy as np
import matplotlib.pyplot
as plt
 def Randwalk(n):
   x = 0
    y = 0
    time =[x]
position =[y]

    for i in range (1,n+1):
move = np.random.uniform(0,1)
if move < 0.5:
        x += 1

        y   +=   1

elif move > 0.5:
```

```
        x +=1
y += -1
      time.append(x)
       position.append(y)
   return[time,position]
Randwalk1 = Randwalk(1000)
Randwalk2 = Randwalk(1000) Randwalk3 =
Randwalk(1000) plt.plot(Randwalk1[0],Randwalk1[1],'r-',
label = "Randwalk1")
plt.plot(Randwalk2[0],Randwalk2[1],'g-', label =
"Randwalk2") plt.plot(Randwalk3[0],Randwalk3[1],'b-',
label = "Randwalk3")
 plt.title("1-D Random Walks")

 plt.legend(loc='upper center', bbox_to_anchor=(0.5,-0.1), fancybox=True, shadow=True)
 plt.show()
```

OUTPUT

```
import numpy as np
import matplotlib.pyplot as plt
def Randwalk(n):
    x = 0
    y = 0
    time = [x]
    position = [y]
    for i in range (1,n+1):
        move = np.random.uniform(0,1)
        if move < 0.5:
            x += 1
            y += 1
        elif move > 0.5:
            x += 1
            y += -1
        time.append(x)
        position.append(y)
    return [time,position]
Randwalk1 = Randwalk(1000)
Randwalk2 = Randwalk(1000)
Randwalk3 = Randwalk(1000)
plt.plot(Randwalk1[0],Randwalk1[1],'r-', label = "Randwalk1")
plt.plot(Randwalk2[0],Randwalk2[1],'g-', label = "Randwalk2")
plt.plot(Randwalk3[0],Randwalk3[1],'b-', label = "Randwalk3")
plt.title("1-D Random Walks")
plt.legend(loc='upper center', bbox_to_anchor=(0.5,-0.1), fancybox=True, shadow=True)
plt.show()
```



Result:

Thus Random walk in one space dimensions has been successfully executed using jupyter notebook.

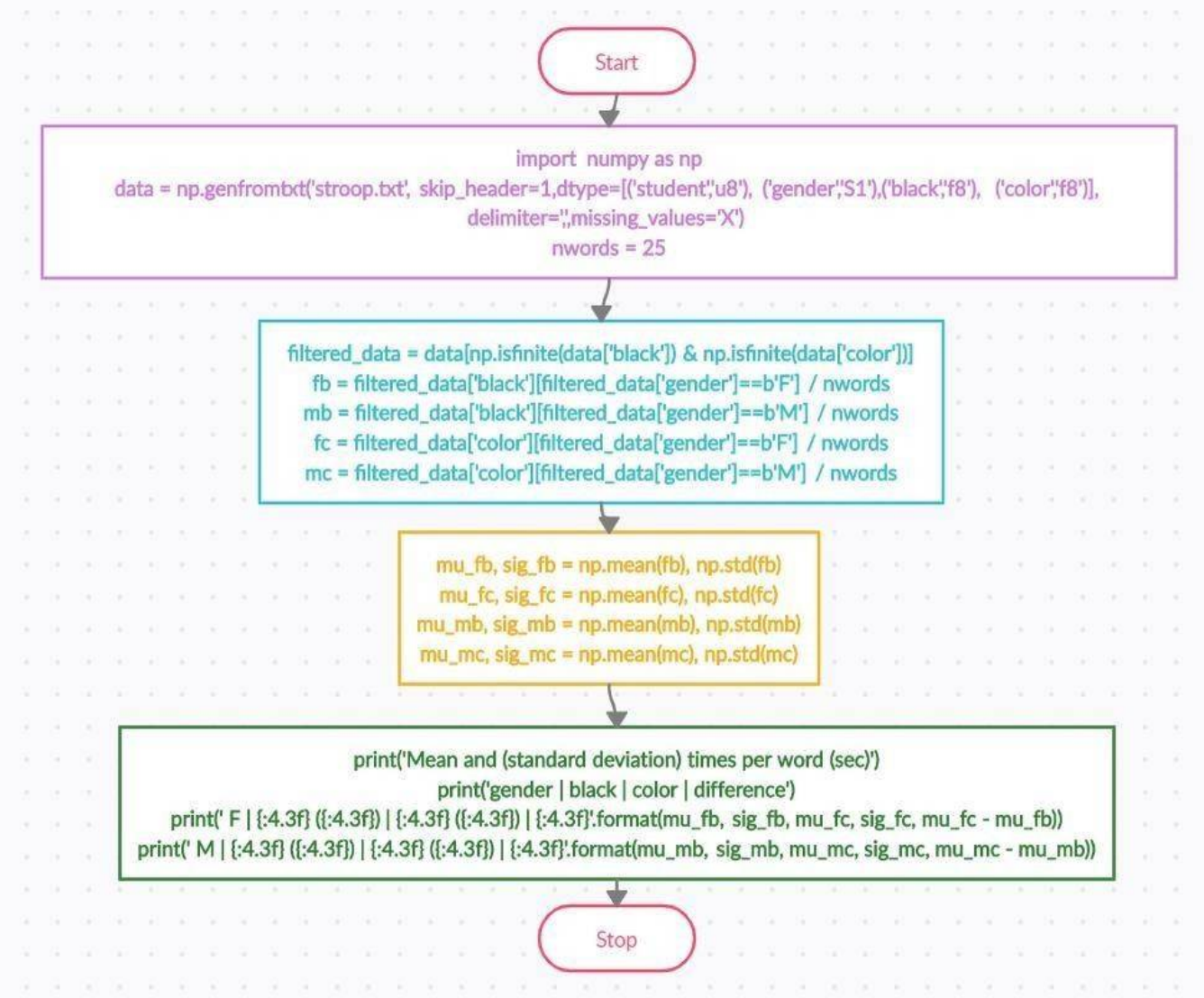
13. Numpy file reading and data analysis

Aim: Numpy file reading and data analysis

Introduction /Background

In this problem statement we write the code for numpy file reading and data analysis

Flowchart/Algorithm:



Materials/Equipment

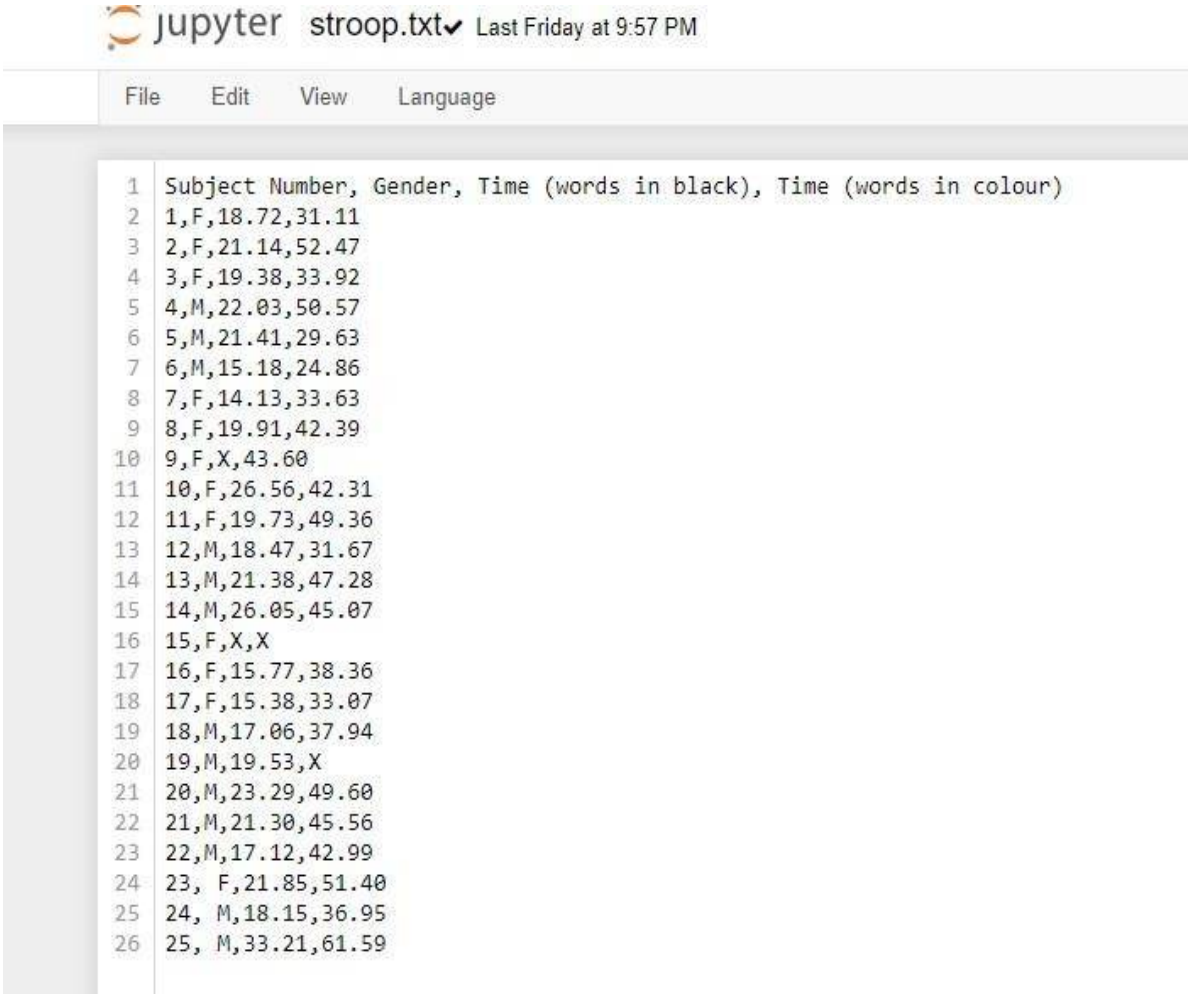
1.AnacondaNavigator
2.Jupyter Notebook

PROCEDURE:

1. Open Anaconda Navigator and launch Jupyter Notebook,
2. Preparing a text file namedstroop.txt
3. Importing pandas as pd
4. Getting the information from the text file into the data variable
5. Removing the invalid rows from the dataset
6. Extract rows by gender (M/F) and word color (black/color) and normalize to the time taken per word
7. Produce statistics mean and standard deviation by gender and word color
8. Run and print the code to get desired output result

PROGRAM AND INPUT DATA:

Text file input:



CODE:

```
import numpy as np
# Read in the data from stroop.txt, identifying missing values and replacing them
with NaN
data = np.genfromtxt('stroop.txt', skip_header=1,dtype=[('student','u8'),
('gender','S1'),('black','f8'), ('color','f8')], delimiter=',',missing_values='X')
nwords = 25
# Remove invalid rows from data set
filtered_data = data[np.isfinite(data['black']) &
np.isfinite(data['color'])] # Extract rows by gender (M/F) and
word color (black/color) and normalize
# to time taken per word fb =
filtered_data['black'][filtered_data['gender']==b'F'] / nwords
mb = filtered_data['black'][filtered_data['gender']==b'M'] /
nwords fc =
filtered_data['color'][filtered_data['gender']==b'F'] / nwords
mc = filtered_data['color'][filtered_data['gender']==b'M'] /
nwords # Produce statistics: mean and standard deviation by
gender and word color mu_fb, sig_fb = np.mean(fb),
np.std(fb) mu_fc, sig_fc = np.mean(fc), np.std(fc) mu_mb,
sig_mb = np.mean(mb), np.std(mb) mu_mc, sig_mc =
np.mean(mc), np.std(mc)
print('Mean and (standard deviation) times per word
(sec)') print('gender | black | color | difference')
print(' F | {:4.3f} ({:4.3f}) | {:4.3f} ({:4.3f}) | {:4.3f}'.format(mu_fb, sig_fb,
mu_fc, sig_fc,
mu_fc - mu_fb))
print(' M | {:4.3f} ({:4.3f}) | {:4.3f} ({:4.3f}) | {:4.3f}'.format(mu_mb, sig_mb,
mu_mc,
sig_mc, mu_mc - mu_mb))
```

OUTPUT:

```
import  numpy as np
# Read in the data from stroop.txt, identifying missing values and  replacing them with NaN
data = np.genfromtxt('stroop.txt', skip_header=1,dtype=[('student','u8'), ('gender','S1'),('black','f8'), ('color','f8')],
delimiter=',',missing_values='X')
nwords = 25
# Remove invalid rows from data set
filtered_data = data[np.isfinite(data['black']) & np.isfinite(data['color'])]
# Extract rows by gender (M/F) and word color (black/color) and normalize
# to time taken per word
fb = filtered_data['black'][filtered_data['gender']==b'F'] / nwords
mb = filtered_data['black'][filtered_data['gender']==b'M'] / nwords
fc = filtered_data['color'][filtered_data['gender']==b'F'] / nwords
mc = filtered_data['color'][filtered_data['gender']==b'M'] / nwords
# Produce statistics: mean and standard deviation by gender and word color
mu_fb, sig_fb = np.mean(fb), np.std(fb)
mu_fc, sig_fc = np.mean(fc), np.std(fc)
mu_mb, sig_mb = np.mean(mb), np.std(mb)
mu_mc, sig_mc = np.mean(mc), np.std(mc)
print('Mean and (standard deviation) times per word (sec)')
print('gender | black | color | difference')
print(' F | {:4.3f} ({:4.3f}) | {:4.3f} ({:4.3f}) | {:4.3f}'.format(mu_fb, sig_fb, mu_fc, sig_fc, mu_fc - mu_fb))
print(' M | {:4.3f} ({:4.3f}) | {:4.3f} ({:4.3f}) | {:4.3f}'.format(mu_mb, sig_mb, mu_mc, sig_mc, mu_mc - mu_mb))
```

```
Mean and (standard deviation) times per word (sec)
gender | black | color | difference
 F | 0.760 (0.149) | 1.628 (0.275) | 0.868
 M | 0.813 (0.126) | 1.621 (0.342) | 0.808
```

Prelab Questions:

1)How would you create identity matrix inpython?
2) What is the significance ofnp.genfromtxt?
3) What does np.std do?

Post lab Questions:

1)Find the output of mu_mb, sig_mb = np.mean(mb), np.std(mb)
from aboveprogram?
2) In [x]: print(np.cov(X)) X=[[0.115 , 0.0575,-1.2325],
[0.0575, 3.757 , -0.8775], [-1.2325, -0.8775, 14.525]] ,output?

Result:

Thus numpy file reading and data analysis has been successfully executed using jupyter notebook.

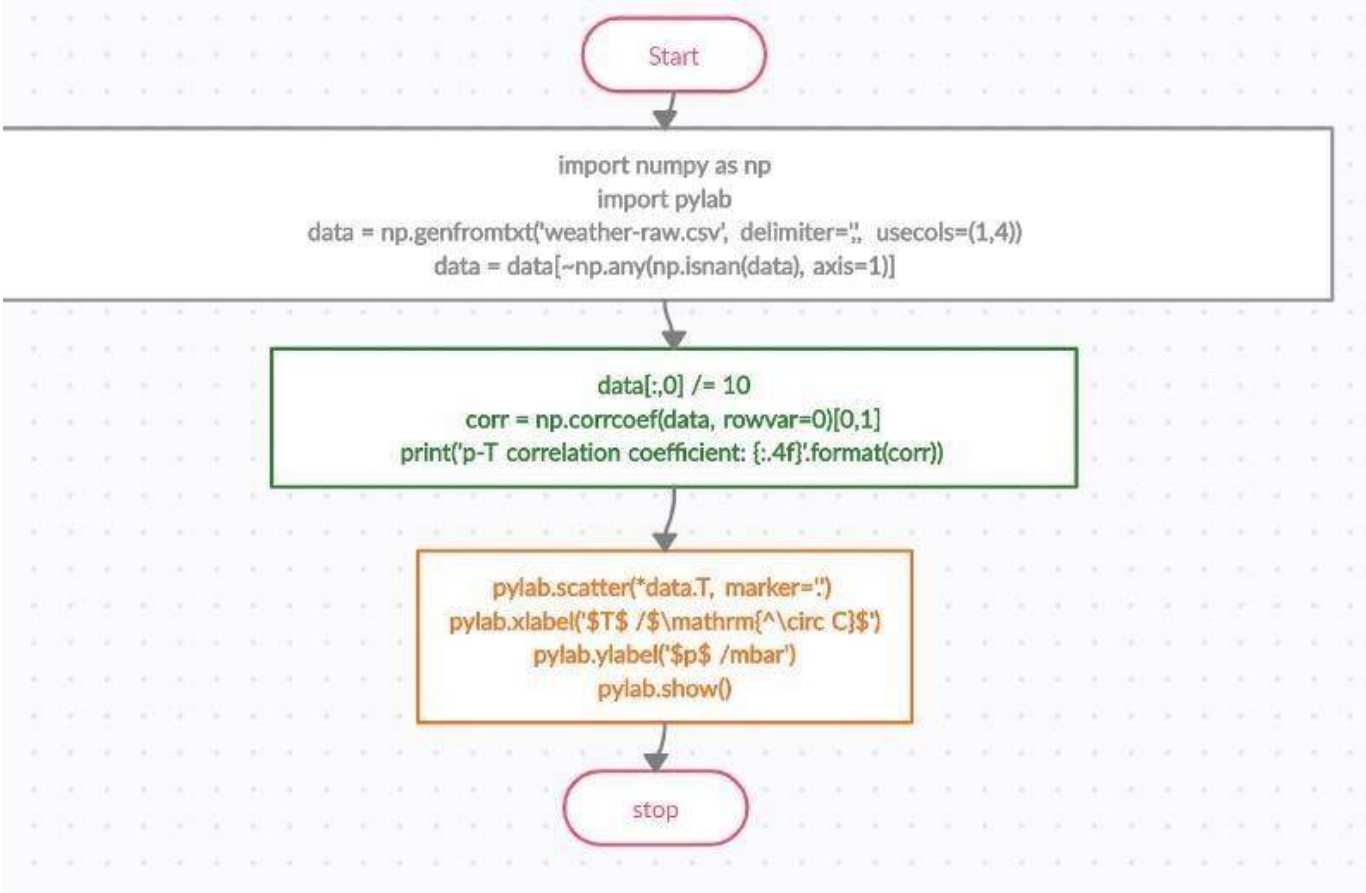
14. Calculating the correlation coefficient between air temperature and pressure

Aim: Calculating the correlation coefficient between air temperature and pressure

Introduction / Background

In this problem statement we write the code Calculating the correlation coefficient between air temperature and pressure**.**

Flowchart/Algorithm:



SOFTWARE USED:

1.AnacondaNavigator

2.JupyterNotebook

PROCEDURE:

1. Open Anaconda Navigator and launch JupyterNotebook.
2. Now import numpy asnp
3. import pylab
4. Assign a variable named DATA with the information in CSV file of thewebsite.
5. Using CORR calculate the correlation coefficient
   as: corr = np.corrcoef(data,rowvar=0)[0,1]
6. Now plot ithe data using pylab.scatter(*data.T,marker='.')
7. Using 'XLABEL' , 'YLABEL' specify display names for x and yfields.
8. A plot will be displayed depicting the ouput as well as displaying the correlation coefficient.

PROGRAM AND INPUT DATA:

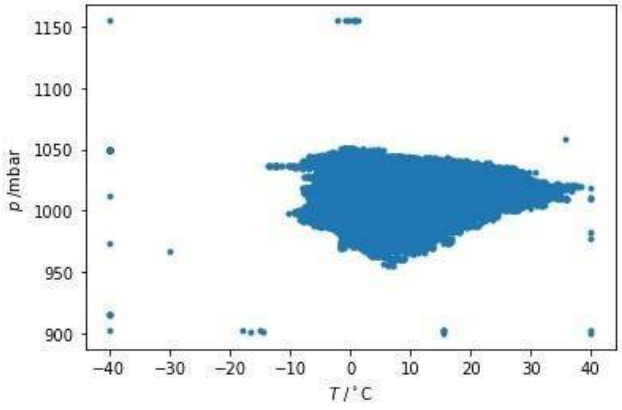
CODE:

```
import numpy as np
import pylab
data = np.genfromtxt('weather-raw.csv', delimiter=',', usecols=(1,4)) #
Remove any rows with either missing T or missing
p data = data[~np.any(np.isnan(data), axis=1)]
# Temperatures are reported after multiplication by a factor of 10 so
remove # this factor
data[:,0]
/= 10
# Get the correlation coefficient
corr = np.corrcoef(data, rowvar=0)[0,1]
print('p-T correlation coefficient: {:.4f}'.format(corr)) #
Plot the data on a scatterplot: T on x-axis, p on y-axis.
pylab.scatter(*data.T, marker='.')
pylab.xlabel('$T$ /$\mathrm{^\circ
C}$') pylab.ylabel('$p$ /mbar')
pylab.show()
```

OUTPUT:

In [3]:
```
import numpy as np
import pylab
data = np.genfromtxt('weather-raw.csv', delimiter=',', usecols=(1,4))
# Remove any rows with either missing T or missing p
data = data[~np.any(np.isnan(data), axis=1)]
# Temperatures are reported after multiplication by a factor of 10 so remove
# this factor
data[:,0] /= 10
# Get the correlation coefficients
corr = np.corrcoef(data, rowvar=0)[0,1]
print('p-T correlation coefficient: {:.4f}'.format(corr))
# Plot the data on a scatterplot: T on x-axis, p on y-axis.
pylab.scatter(*data.T, marker='.')
pylab.xlabel('$T$ /$\mathrm{^\circ C}$')
pylab.ylabel('$p$ /mbar')
pylab.show()
```

```
p-T correlation coefficient: 0.0276
```



Pre lab Questions:

1) Write about np.isnan?
2) What is the significance of pylab.scatter?
3) What does np.corrcoefdo?

Post lab Questions:

1) Find the output of x = np.array([1., 2., 3., 4., 5.]) y = np.array([0.08, 0.31, 0.41, 0.48, 0.62]) print(np.corrcoef(x,y))?

2) marks = np.array([45, 68, 56, 23, 60, 87, 75, 59, 63, 72]) bins = [20, 40, 60, 80, 100]. Draw a histogram for this data?

Result:

Thus the correlation coefficient between air temperature and pressure has been successfully calculated and displayed using jupyter notebook.

15	Numpy signal processing-Blurring an image with a Gaussian filter

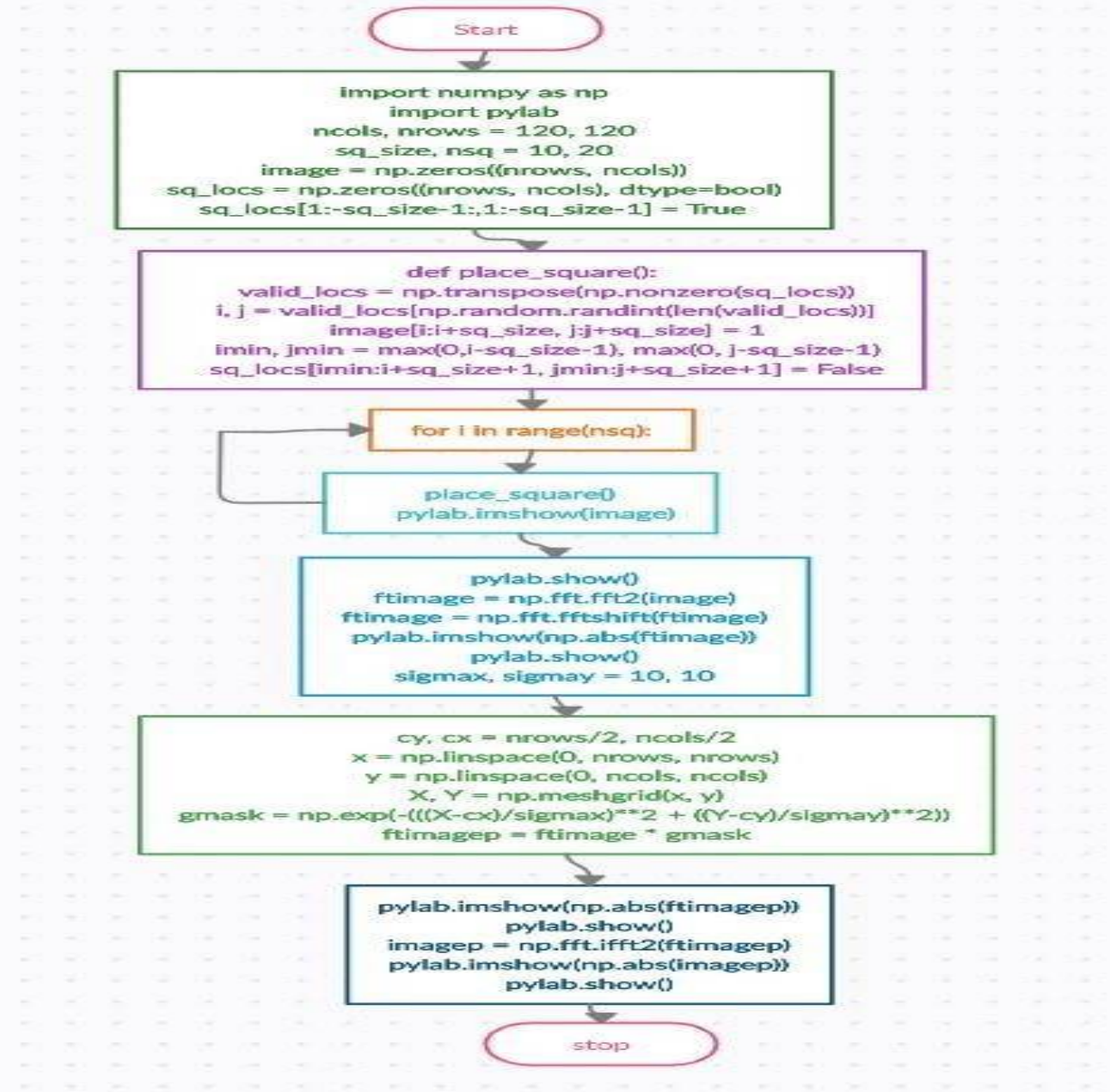
Aim:

To find the Numpy signal processing-Blurring an image using a Gaussian filter.

Introduction / Background:

In this problem statement we write the code for the Numpy signal processing-Blurring an image with a Gaussian filter

Flowchart/Algorithm:



Materials/Equipment

1.AnacondaNavigator
2.Jupyter Notebook

PROCEDURE:

1. Open Anaconda Navigator and launch JupyterNotebook.
2. Set the image size, square side length, number ofsquares
3. The image array (0=background, 1=square) and boolean array of allowed places to add a square so that it doesn't touch another or the imagesides
4. valid_locs is an array of the indexes of True entries insq_locs
5. pick one such entry at random, and add the square so its topleft
6. Then updatesq_locs
7. Add the required number of squares to theimage
8. Finally, take the inverse transform and show the blurredimage
9. Verify theoutput

PROGRAM AND INPUT DATA:

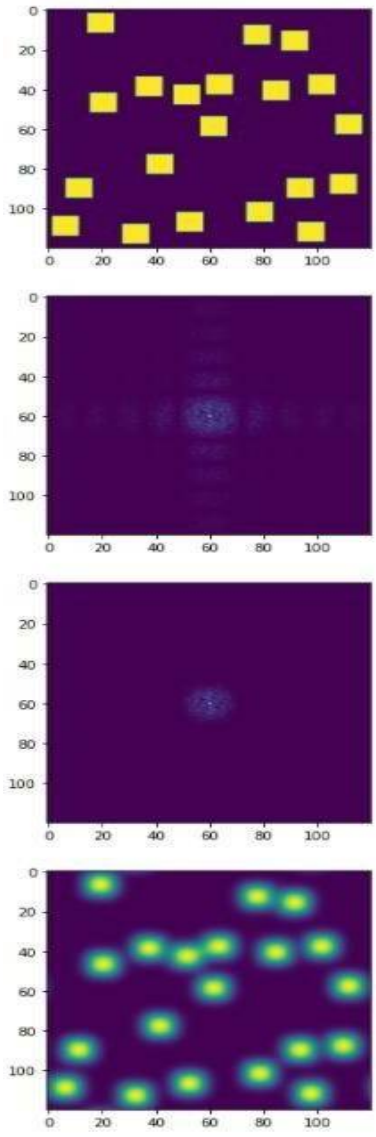
CODE:

```
import numpy as
np import pylab
# image size, square side length, number of
squares ncols, nrows = 120, 120 sq_size, nsq
= 10, 20
# The image array (0=background, 1=square) and boolean array of
allowed places # to add a square so that it doesn't touch another or the
image sides
image = np.zeros((nrows, ncols)) sq_locs = np.zeros((nrows,
ncols), dtype=bool) sq_locs[1:-sq_size-1:,1:-sq_size-1] =
True def place_square():
  """ Place a square at random on the image and update
sq_locs. """ # valid_locs is an array of the indexes of True
entries in sq_locs valid_locs =
np.transpose(np.nonzero(sq_locs))
  # pick one such entry at random, and add the square so its top left
```

```
    # corner is there; then update sq_locs
    i, j = valid_locs[np.random.randint(len(valid_locs))]
image[i:i+sq_size, j:j+sq_size] =1

    imin, jmin = max(0,i-sq_size-1), max(0, j-sq_size-1) sq_locs[imin:i+sq_size+1,
jmin:j+sq_size+1] = False # Add the required number of squares to the image

for i in range(nsq):
place_square()
pylab.imshow(imag
e)
pylab.show()
# Take the two-dimensional DFT and center the
frequencies ftimage = np.fft.fft2(image) ftimage =
np.fft.fftshift(ftimage)     pylab.imshow(np.abs(ftimage))
pylab.show()

# Build and apply a Gaussian filter.
sigmax, sigmay = 10, 10 cy, cx = nrows/2, ncols/2 x =
np.linspace(0, nrows, nrows) y = np.linspace(0, ncols,
ncols) X, Y = np.meshgrid(x, y) gmask = np.exp(-(((X-
cx)/sigmax)**2 + ((Y-cy)/sigmay)**2)) ftimagep =
ftimage * gmask pylab.imshow(np.abs(ftimagep))
pylab.show()

# Finally, take the inverse transform and show the blurred
image imagep = np.fft.ifft2(ftimagep)
pylab.imshow(np.abs(imagep)) pylab.show()
```

OUTPUT:

```
import numpy as np
import pylab
# image size, square side length, number of squares
ncols, nrows = 120, 120
sq_size, nsq = 10, 20
# The image array (0=background, 1=square) and boolean array of allowed places
# to add a square so that it doesn't touch another or the image sides
image = np.zeros((nrows, ncols))
sq_locs = np.zeros((nrows, ncols), dtype=bool)
sq_locs[1:-sq_size-1:,1:-sq_size-1] = True
def place_square():
    """ Place a square at random on the image and update sq_locs. """
    # valid_locs is an array of the indexes of True entries in sq_locs
    valid_locs = np.transpose(np.nonzero(sq_locs))
    # pick one such entry at random, and add the square so its top left
    # corner is there; then update sq_locs
    i, j = valid_locs[np.random.randint(len(valid_locs))]
    image[i:i+sq_size, j:j+sq_size] = 1
    imin, jmin = max(0,i-sq_size-1), max(0, j-sq_size-1)
    sq_locs[imin:i+sq_size+1, jmin:j+sq_size+1] = False
# Add the required number of squares to the image
for i in range(nsq):
    place_square()
    pylab.imshow(image)

pylab.show()
# Take the two-dimensional DFT and center the frequencies
ftimage = np.fft.fft2(image)
ftimage = np.fft.fftshift(ftimage)
pylab.imshow(np.abs(ftimage))
pylab.show()
# Build and apply a Gaussian filter.
sigmax, sigmay = 10, 10
cy, cx = nrows/2, ncols/2
x = np.linspace(0, nrows, nrows)
y = np.linspace(0, ncols, ncols)
X, Y = np.meshgrid(x, y)
gmask = np.exp(-(((X-cx)/sigmax)**2 + ((Y-cy)/sigmay)**2))
ftimagep = ftimage * gmask
pylab.imshow(np.abs(ftimagep))
pylab.show()
# Finally, take the inverse transform and show the blurred image
imagep = np.fft.ifft2(ftimagep)
pylab.imshow(np.abs(imagep))
pylab.show()
```


0
20
40
60
80
100
0
20
40
60
80
100

Pre lab Questions:

1)What type of Gaussian filter is used in above program?

2) What is the significance ofnp.fft.fft?

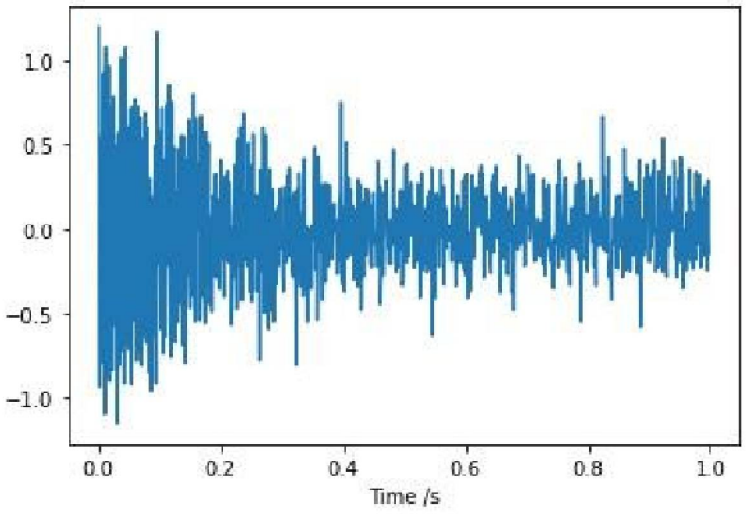
3) Explain place_square() function in above program?

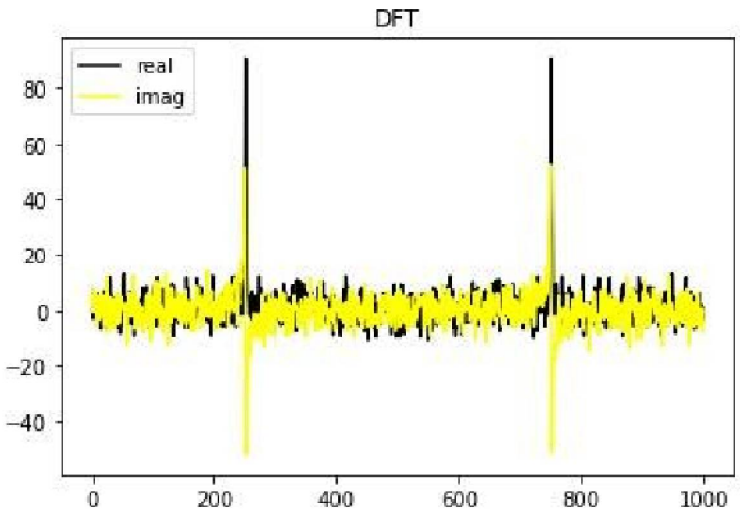
Post lab Questions:

1) Consider a signal in the time domain defined by the function f (t) = cos(2πνt)e−t/τ , with frequency ν =250 Hz decaying exponentially with a lifetime τ = 0.2 s. Plot the function, sampled at 1,000 Hz, and its discrete Fourier transform against frequency?

OUTPUT:

```
#RA1811004010099s
tou=0.2
v=250
fsamp = 1000
t = np.arange(0, 1, 1/fsamp)
n = len(t)
f = np.cos(2*np.pi*v*t)*np.exp(-t/tou)
f += 0.2 * np.random.randn(n)
pylab.plot(t, f)
pylab.xlabel('Time /s')
```





Result:

Thus the Numpy signal processing-Blurring an image with a Gaussian filter has been successfully calculated using jupyter notebook.