UNIVERSITY OF MUMBAI

PRACTICAL REPORT
M.Sc. I.T (PART 2) 2023-2024

PAPER 1: APPLIED ARTIFICIAL INTELLIGENCE

PAPER 2: MACHINE LEARNING

PAPER 3: ROBOTIC PROCESS AUTOMATION

SUBMITTED BY:
Snehal Nagesh Bachewar

PROF. SUMIYA MADOO
{TEACHER INCHARGE}

PROF. SRAVANTI GAJJELLI
{TEACHER INCHARGE}

PROF. SIMRAN SHAIKH
{TEACHER INCHARGE}

Padmashri Annasaheb Jadhav Bhartiya Samaj Unnati Mandal's
B.N.N COLLEGE
(Arts, Science, Commerce & Self-Funded Course)
(Affiliated to University of Mumbai)
DEPARTMENT OF INFORMATION TECHNOLOGY
BHIWANDI, MAHARASHTRA-421302

Padmashri Annasaheb Jadhav Bhartiya Samaj Unnati Mandal's

B.N.N College of Arts, Science and Commerce, Bhiwandi.

(Self-Funded Course)

(Department of Information Technology)

CERTIFICATE

This is to certify that Mr./Miss. **Snehal Nagesh Bachewar .**
PRN No. **2016016402206486** Class: **MSc-IT** Exam Seat No. **2294989 .**
has satisfactorily completed practical in **Applied Artificial Intelligence .**
As laid down in the regulation of University of Mumbai for the purpose of **Semester- III Practical** Examination 2023-2024.

Date:
Place: **Bhiwandi**

In-Charge Professor **Signature of External Examiner**

Signature of HOD

Padmashri Annasaheb Jadhav Bhartiya Samaj Unnati Mandal's

B.N.N College of Arts, Science and Commerce, Bhiwandi.

(Self-Funded Course)

(Department of Information Technology)

CERTIFICATE

This is to certify that Mr./Miss. **Snehal Nagesh Bachewar.**

PRN No. **2016016402206486** Class: **MSc-IT** Exam Seat No. **2294989 .**

has satisfactorily completed practical in **Machine Learning .**

As laid down in the regulation of University of Mumbai for the purpose of **Semester- III Practical** Examination 2023-2024.

Date:

Place: **Bhiwandi**

In-Charge Professor **Signature of External Examiner**

Signature of HOD

Padmashri Annasaheb Jadhav Bhartiya Samaj Unnati Mandal's

B.N.N College of Arts, Science and Commerce, Bhiwandi.

(Self-Funded Course)

(Department of Information Technology)

CERTIFICATE

This is to certify that Mr./Miss. **Snehal Nagesh Bachewar**.

PRN No. **2016016402206486** Class: **MSc-IT** Exam Seat No. **2294989** .

has satisfactorily completed practical in **Robotic Process Automation** .

As laid down in the regulation of University of Mumbai for the purpose of **Semester- III Practical** Examination 2023-2024.

Date:

Place: **Bhiwandi**

In-Charge Professor **Signature of External Examiner**

Signature of HOD

APPLIED ARTIFICIAL INTELLIGENCE

INDEX

PRATICAL 1) Design a bot using AIML

CODE: We need to create following three files.

1) std-startup.xml

```
<aiml version="1.0.1" encoding="UTF-8">
<!-- std-startup.xml -->
<category>
        <pattern>LOAD</pattern>
        <template>
                <learn>basic_chat.aiml</learn>
        </template>
 </category>
</aiml>
```

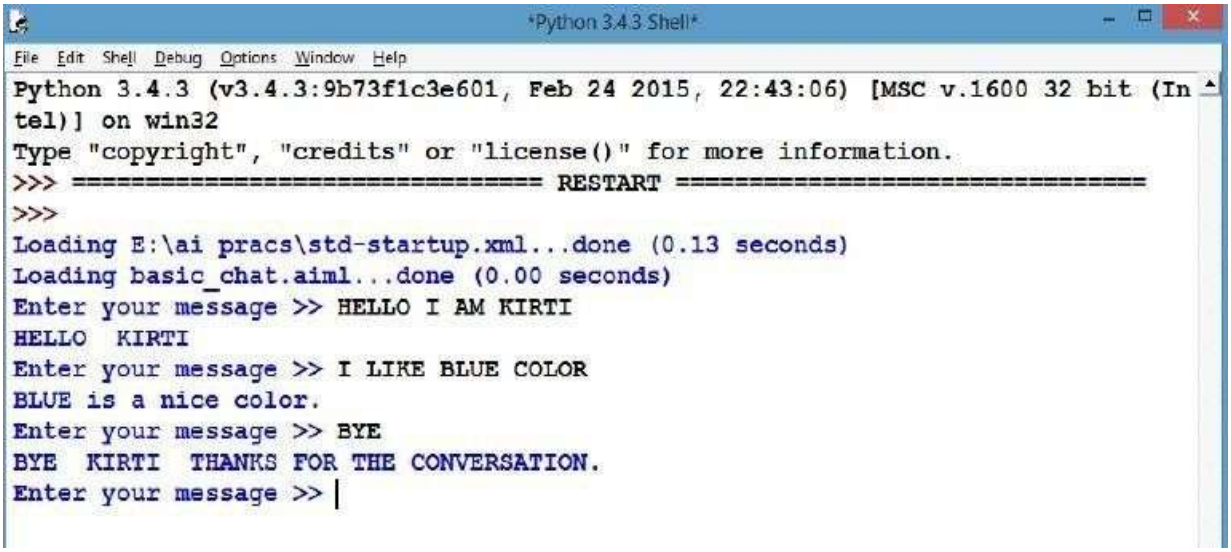
2) basic_chat.aiml

```
<aiml version="1.0.1" encoding="UTF-8">
<!-- basic_chat.aiml -->
 <category>
        <pattern>HELLO I AM *</pattern>
        <template> HELLO <set name="username"> <star/> </set> </template>
 </category>
 <category>
        <pattern>I LIKE * COLOR</pattern>
        <template><star index="1"/> is a nice color.</template>
 </category>
 <category>
        <pattern>BYE</pattern>
        <template> BYE <get name="username"/> THANKS FOR THECONVERSATION.
 </template>
 </category>
</aiml>
```

3) testbot.py

```
import aiml
kernel = aiml.Kernel()
kernel.learn("F:\\aaipracts\\std-startup.xml")
kernel.learn("F:\\aaipracts\\basic_chat.aiml")
kernel.respond("LOAD")
# Press CTRL-C to break this loopwhile
True:
        print( kernel.respond(input("Enter your message >> ")))
```

Output:

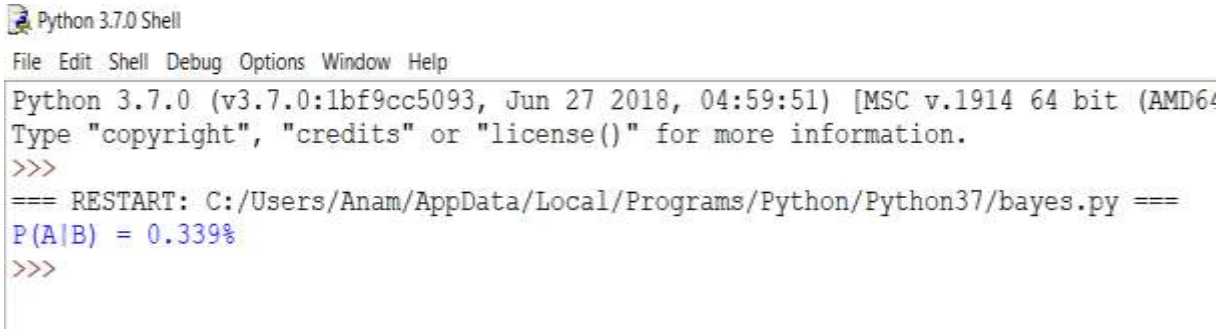

Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ========== RESTART ==========
>>>
Loading E:\ai pracs\std-startup.xml...done (0.13 seconds)
Loading basic_chat.aiml...done (0.00 seconds)
Enter your message >> HELLO I AM KIRTI
HELLO KIRTI
Enter your message >> I LIKE BLUE COLOR
BLUE is a nice color.
Enter your message >> BYE
BYE KIRTI THANKS FOR THE CONVERSATION.
Enter your message >>

PRATICAL 2) Implement Bayes Theorem using Python

```
# calculate the probability of cancer patient and diagnostic test
# calculate P(A|B) given P(A), P(B|A), P(B|not A)
 def bayes_theorem(p_a, p_b_given_a, p_b_given_not_a):
# calculate P(not A)
        not_a = 1 - p_a #
        calculate P(B)
        p_b = p_b_given_a * p_a + p_b_given_not_a * not_a #
        calculate P(A|B)
        p_a_given_b = (p_b_given_a * p_a) / p_breturn
        p_a_given_b

# P(A)
p_a = 0.0002
# P(B|A)
p_b_given_a = 0.85
# P(B|not A)
p_b_given_not_a = 0.05 #
calculate P(A|B)
result = bayes_theorem(p_a, p_b_given_a, p_b_given_not_a)
# summarize
print('P(A|B) = %.3f%%' % (result * 100))
```

Running this program calculates the probability that a patient has cancer.

Output:

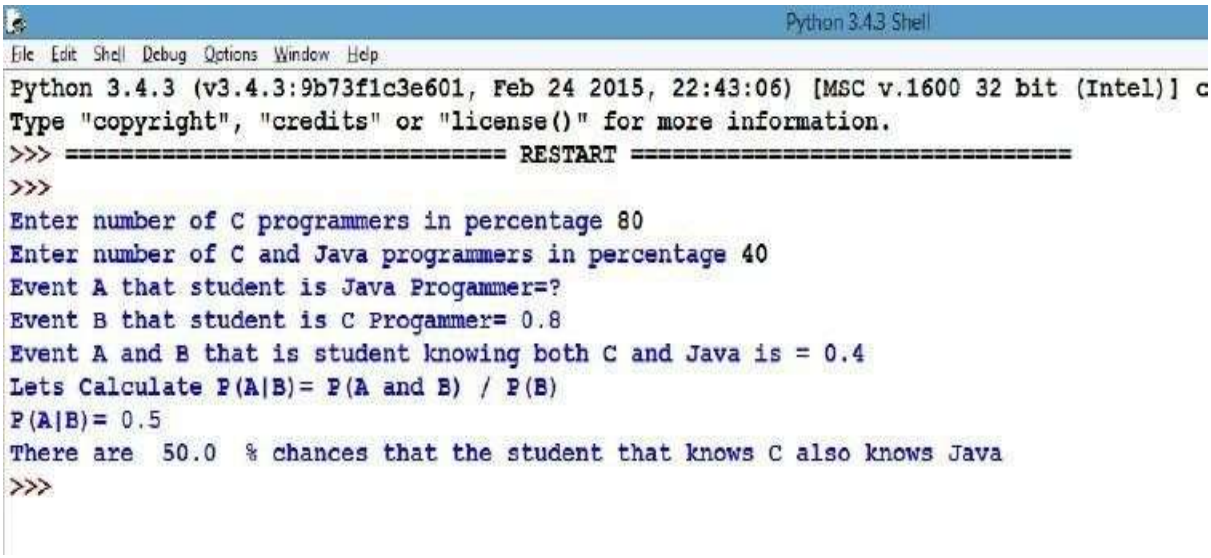


PRATICAL 3) Implement Conditional Probability and joint probability using Python.

A) Code for conditional probability:

```
pofB=float(input("Enter number of C programmers in percentage "))
pofAandB=float(input("Enter number of C and Java programmers in percentage "))
pofB=pofB/100
pofAandB=pofAandB/100
print("Event A that student is Java Progammer=?")
print("Event B that student is C Progammer=",pofB)
print("Event A and B that is student knowing both C and Java is =",pofAandB)
print("Lets Calculate P(A|B)= P(A and B) / P(B)")
pAgivenB=pofAandB / pofB
print("P(A|B)=",pAgivenB)
print("There are ",pAgivenB *100," % chances that the student that knows C also knowsJava")
```

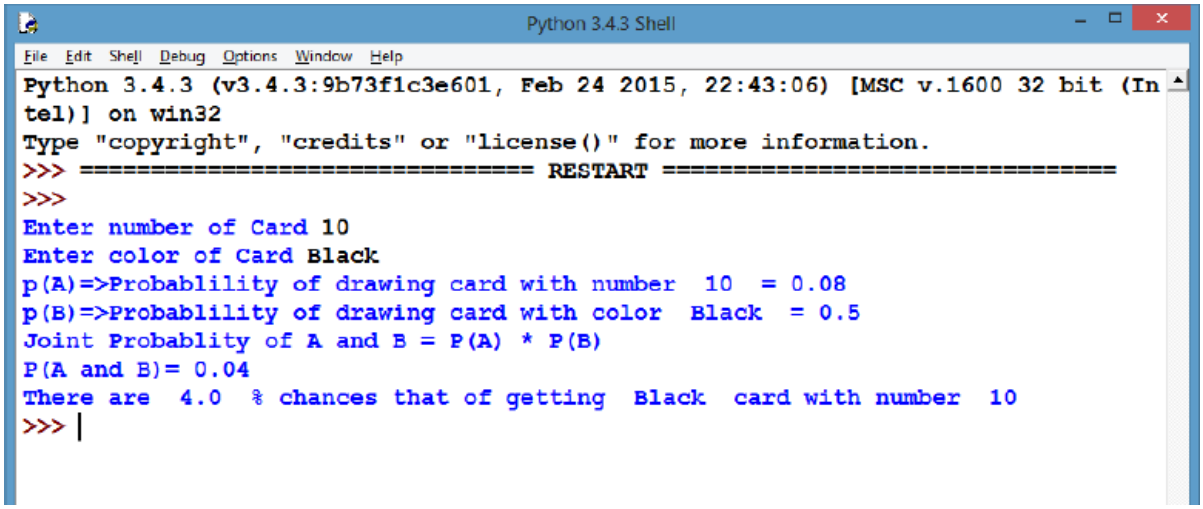
Output:



B)Code for Joint Probability:

```
cardnumber=input("Enter number of Card")
cardcolor=input("Enter color of Card")
pofA=4/52
pofB=26/52
print("p(A)=>Probablility of drawing card
with number
",cardnumber,"=",round(pofA,2))
print("p(B)=>Probablility of drawing card with color ",cardcolor," =",round(pofB,2))
print("Joint Probablity of A and B=P(A) * P(B)")
pAandB=round(pofA pofB,2)
print("P(A and B)=",pAandB)
print("There are ",pAandB *100," % chances that of getting ",cardcolor, " card with number",
cardnumber)
```

Output:



PRATICAL 4) Design a Fuzzy based application using Python / R .

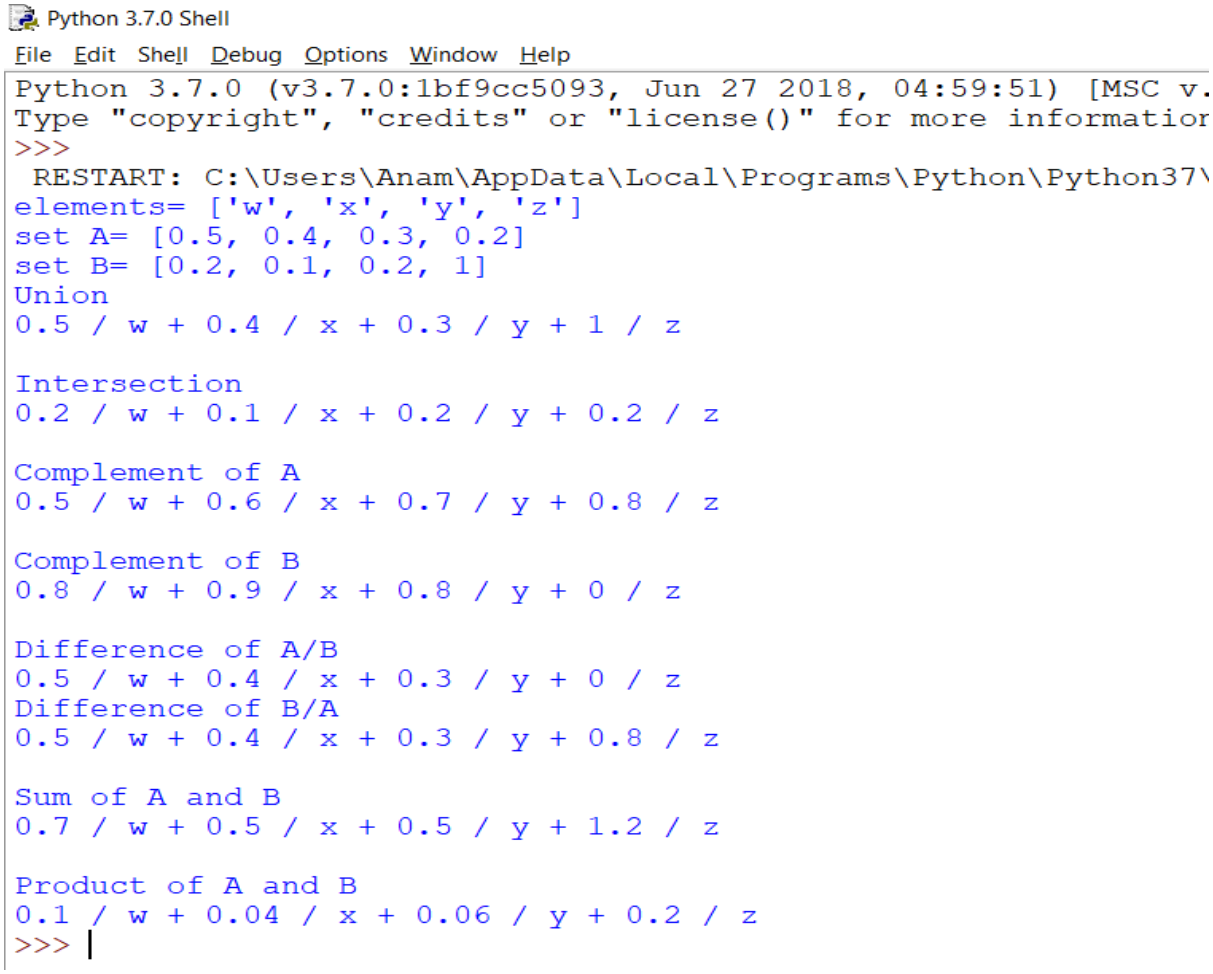
Code:

```
elt=['w','x','y','z']
A=[0.5,0.4,0.3,0.2]
B=[0.2,0.1,0.2,1]
U=[]
print("elements=",elt)
print("set A=",A)
  print("set B=",B)
 for i in range(0,4):
      if A[i]>B[i]:
           U.append(A[i])
       else:
          U.append(B[i])
print("Union")
for i in range(0,3):
    print(U[i] ,"/",elt[i],end=' + ')
    for i in range(3,4):
    print(U[i] ,"/",elt[i],end=' ')
print()
I=[]
  for i in range(0,4):if
       A[i]<B[i]:
           I.append(A[i])
       else:
          I.append(B[i])
print()
print("Intersection")
         for i in range(0,3):
    print(I[i] ,"/",elt[i],end=' + ')
    for i in range(3,4):
      print(I[i] ,"/",elt[i],end=' ')
print()
J=[]
K=[]
C=[1,1,1,1]
print()
 print("Complement of A")
for i in range(0,4):
J.append(C[i]-A[i])
output=round(J[i],2)
for i in range(0,3):
print(J[i] ,"/",elt[i],end=' + ')
for i in range(3,4):
print(J[i] ,"/",elt[i],end=' ')
print()
print()
print("Complement of B")
for i in range(0,4):
K.append(C[i]-B[i])
```

```
   for i in range(0,3):
    print(K[i] ,"/",elt[i],end=' + ')
    for i in range(3,4):
    print(K[i] ,"/",elt[i],end=' ')
    L=[]
M=[]
print()
 for i in range(0,4):if
     A[i]<K[i]:
L.append(A[i])
         else:
        L.append(K[i])
print()
print("Difference of A/B")
for i in range(0,3):
    print(L[i] ,"/",elt[i],end=' + ')
    for i in range(3,4):
    print(L[i] ,"/",elt[i],end=' ')
    for i in range(0,4):
     if B[i]<J[i]:
         M.append(A[i])
     else:
        M.append(J[i])
print()
print("Difference of B/A")
for i in range(0,3):
    print(M[i] ,"/",elt[i],end=' + ')
    for i in range(3,4):
    print(M[i] ,"/",elt[i],end=' ')
    print()
Sum=[]
Sum1=[]
print()
print("Sum of A and B")
for i in range(0,4):
     Sum.append(A[i]+B[i])
     output=round(Sum[i],2)
     Sum1.append(output)
for i in range(0,3):
    print(Sum1[i] ,"/",elt[i],end=' + ')
    for i in range(3,4):
    print(Sum1[i] ,"/",elt[i],end=' ')
    print()
Prod=[]
Prod1=[]
print()
print("Product of A and B")
for i in range(0,4):
     Prod.append(A[i]*B[i])
```

```
    output=round(Prod[i],2)
    Prod1.append(output)
for i in range(0,3):
  print(Prod1[i] ,"/",elt[i],end=' + ')
  for i in range(3,4):
    print(Prod1[i] ,"/",elt[i],end=' ')
```

Output:



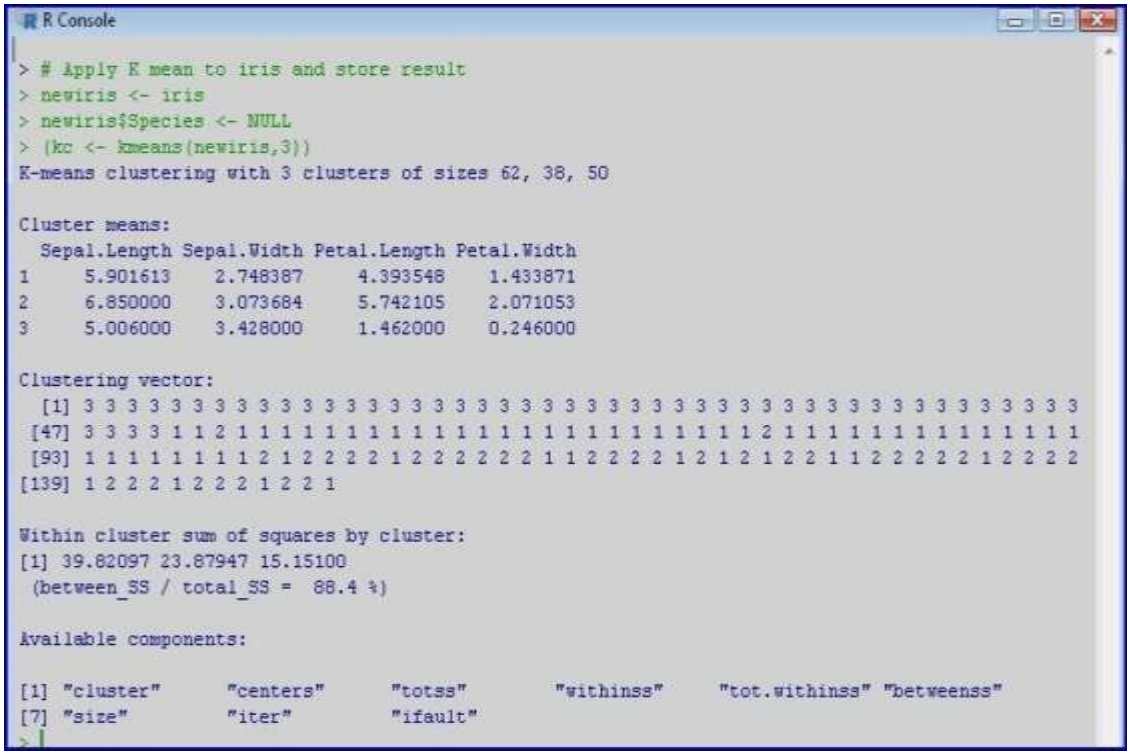
PRATICAL 5) Write an application to implement clustering algorithm.

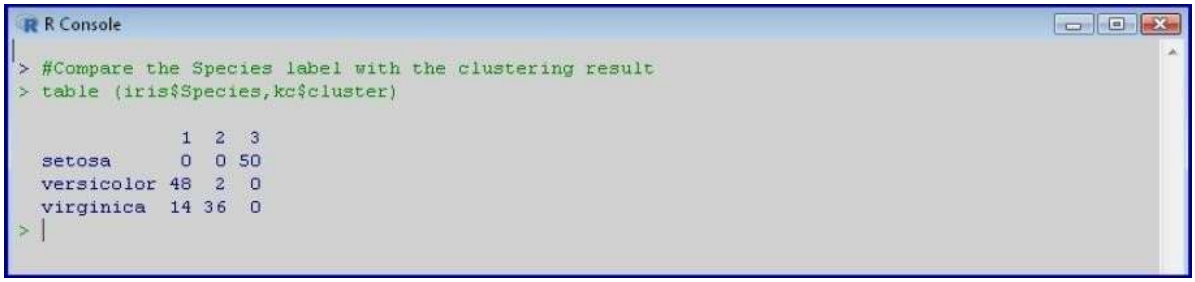
K-Means Clustering:

Code:

```
newiris <- iris
newiris$Species <- NULL
(kc <- kmeans(newiris,3))
print(kc)
# Compare the Species label with the clustering result.table
(iris$Species,kc$cluster)
plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
points(kc$centers[,c("Sepal.Length","Sepal.Width")],col=1:3,pch=8,cex=2)
```

Output:



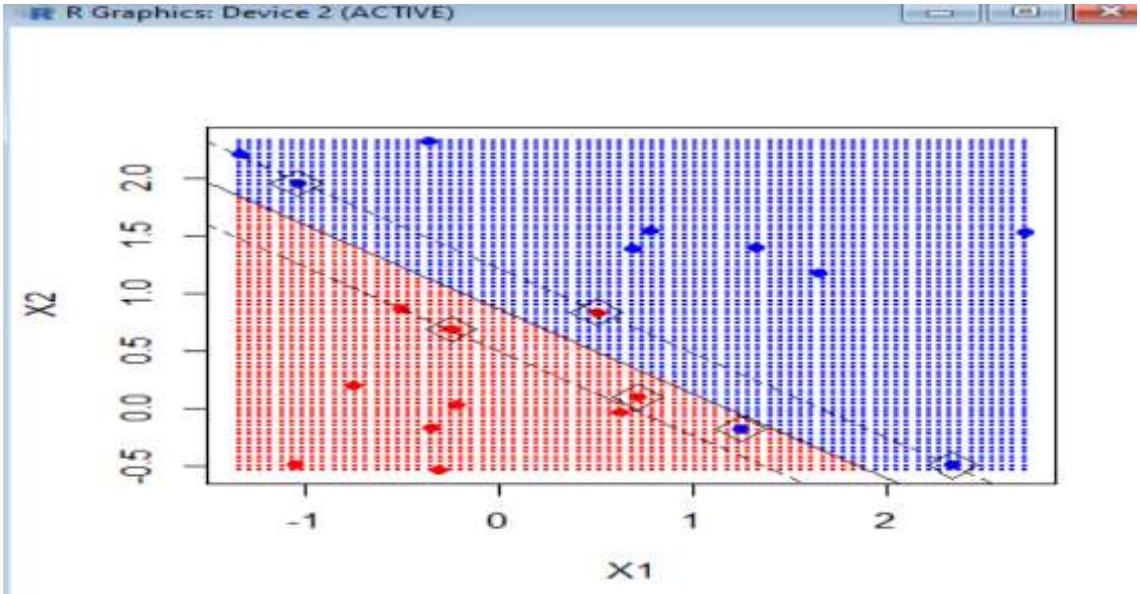


PRATICAL 6) Write an application to implement clustering algorithm.

Code :

```
set.seed(10111)
x=matrix(rnorm(40),20,2)
y=rep(c(-1,1),c(10,10))
x[y==1,]=x[y==1,]+1
plot(x,col=y+3,pch=19)
# Now you load the package e1071 which contains the svm function. > library(e1071)
dat = data.frame(x, y = as.factor(y))
svmfit = svm(y ~ ., data = dat, kernel = "linear", cost = 10, scale = FALSE)
print(svmfit)
plot(svmfit, dat)
make.grid = function(x, n = 75)
{grange = apply(x, 2, range)
x1 = seq(from = grange[1,1], to = grange[2,1], length = n) x2 = seq(from = grange[1,2], to = grange[2,2],
length = n)expand.grid(X1 = x1, X2 = x2)}
xgrid = make.grid(x)
xgrid[1:10,]
ygrid = predict(svmfit, xgrid)
plot(xgrid, col = c("red","blue")[as.numeric(ygrid)], pch = 20, cex = .2)
points(x, col = y + 3, pch = 19)
points(x[svmfit$index,], pch = 5, cex = 2)
beta = drop(t(svmfit$coefs)%*%x[svmfit$index,])
beta0 = svmfit$rho
plot(xgrid, col = c("red", "blue")[as.numeric(ygrid)], pch = 20, cex = .2)
points(x, col = y + 3, pch = 19)
points(x[svmfit$index,], pch = 5, cex = 2)
abline(beta0 / beta[2], -beta[1] / beta[2])
abline((beta0 - 1) / beta[2], -beta[1] / beta[2], lty = 2)
abline((beta0 + 1) / beta[2], -beta[1] / beta[2], lty = 2)
```

OUTPUT:



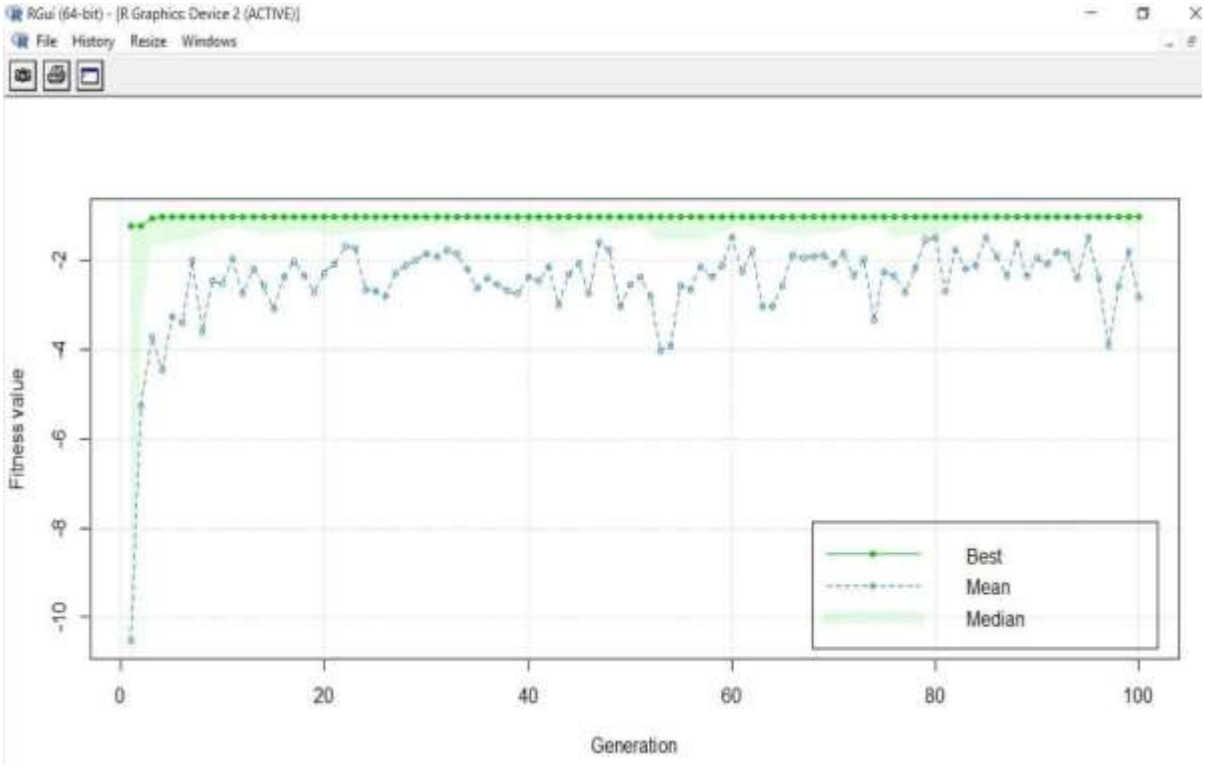
PRATICAL 7) Simulate genetic algorithm with suitable example using Python / R or another platform.

Code:

Packages: GA, Rccp, RcppArmadillo, Cli, Crayon, assertthat

```
f <- function(x) abs(x)+cos(x)
curve(f, -20, 20)
fitness <- function(x) -f(x)
GA <- ga(type = "real-valued", fitness = fitness, lower = -20, upper = 20)
summary(GA)
plot(GA)
```

Output:


RGui (64-bit) - [R Graphics: Device 2 (ACTIVE)]
File History Resize Windows
Fitness value
-2
-4
-6
-8
-10
0
20
40
60
80
100
Generation
Best
Mean
Median

MACHINE LEARNING

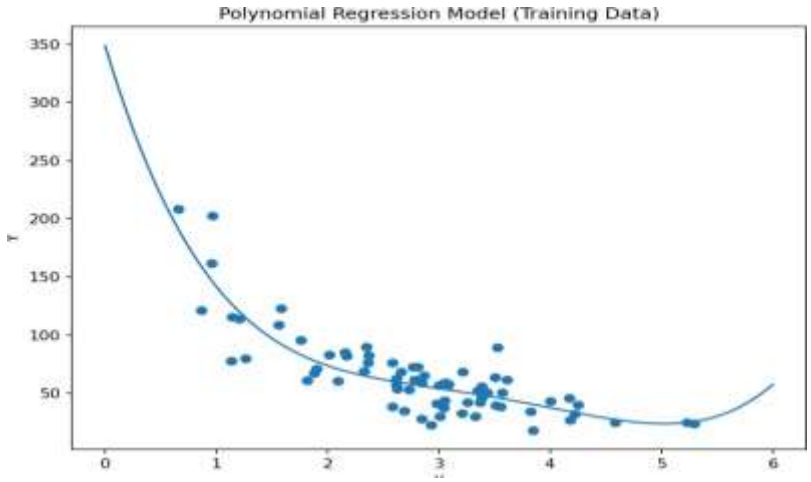
INDEX

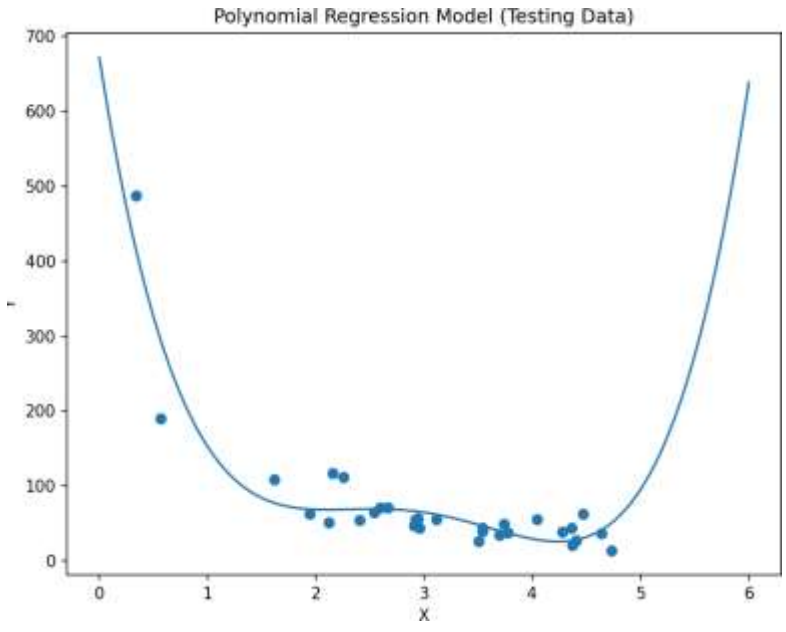
PRATICAL 1A) Design a simple machine learning model to train the training instances and test the same.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score

# Set a random seed for reproducibility
np.random.seed(2)

# Generate random data
x = np.random.normal(3, 1, 100)
y = np.random.normal(150, 40, 100) / x

# Visualize the data
plt.figure(figsize=(8, 6))
plt.scatter(x, y)
plt.title("Scatter Plot of Data")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# Split the data into training and testing sets
train_x, test_x, train_y, test_y = train_test_split(x, y, test_size=0.3)

plt.scatter(train_x,train_y)
plt.show()

plt.scatter(test_x,test_y)
plt.show()

# Create and visualize a polynomial regression model for training data
degree = 4  # Adjust the polynomial degree as needed
train_model = np.poly1d(np.polyfit(train_x, train_y, degree))
myline = np.linspace(0, 6, 200)

plt.figure(figsize=(8, 6))
plt.scatter(train_x, train_y)
plt.plot(myline, train_model(myline))
plt.title("Polynomial Regression Model (Training Data)")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# Calculate and print the R-squared score for the training data
r2_train = r2_score(train_y, train_model(train_x))
print("R-squared score for training data:", r2_train)
```

```
# Create and visualize a polynomial regression model for testing data
test_model = np.poly1d(np.polyfit(test_x, test_y, degree))

plt.figure(figsize=(8, 6))
plt.scatter(test_x, test_y)
plt.plot(myline, test_model(myline))
plt.title("Polynomial Regression Model (Testing Data)")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# Calculate and print the R-squared score for the testing data
r2_test = r2_score(test_y, test_model(test_x))
print("R-squared score for testing data:", r2_test)

# Make predictions using the model
prediction = test_model(5)
print("Prediction for x = 5:", prediction)
```

Output:





PRATICAL 1B) Implement and demonstrate the FIND-S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a .CSV file.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score

# Set a random seed for reproducibility
np.random.seed(2)

# Generate random data
x = np.random.normal(3, 1, 100)
y = np.random.normal(150, 40, 100) / x

# Visualize the data
plt.figure(figsize=(8, 6))
plt.scatter(x, y)
plt.title("Scatter Plot of Data")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# Split the data into training and testing sets
train_x, test_x, train_y, test_y = train_test_split(x, y, test_size=0.3)

plt.scatter(train_x,train_y)
plt.show()

plt.scatter(test_x,test_y)
plt.show()

# Create and visualize a polynomial regression model for training data
degree = 4  # Adjust the polynomial degree as needed
train_model = np.poly1d(np.polyfit(train_x, train_y, degree))
myline = np.linspace(0, 6, 200)

plt.figure(figsize=(8, 6))
plt.scatter(train_x, train_y)
plt.plot(myline, train_model(myline))
plt.title("Polynomial Regression Model (Training Data)")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# Calculate and print the R-squared score for the training data
r2_train = r2_score(train_y, train_model(train_x))
print("R-squared score for training data:", r2_train)
```

```
# Create and visualize a polynomial regression model for testing data
test_model = np.poly1d(np.polyfit(test_x, test_y, degree))

plt.figure(figsize=(8, 6))
plt.scatter(test_x, test_y)
plt.plot(myline, test_model(myline))
plt.title("Polynomial Regression Model (Testing Data)")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# Calculate and print the R-squared score for the testing data
r2_test = r2_score(test_y, test_model(test_x))
print("R-squared score for testing data:", r2_test)

# Make predictions using the model
prediction = test_model(5)
print("Prediction for x = 5:", prediction)
```

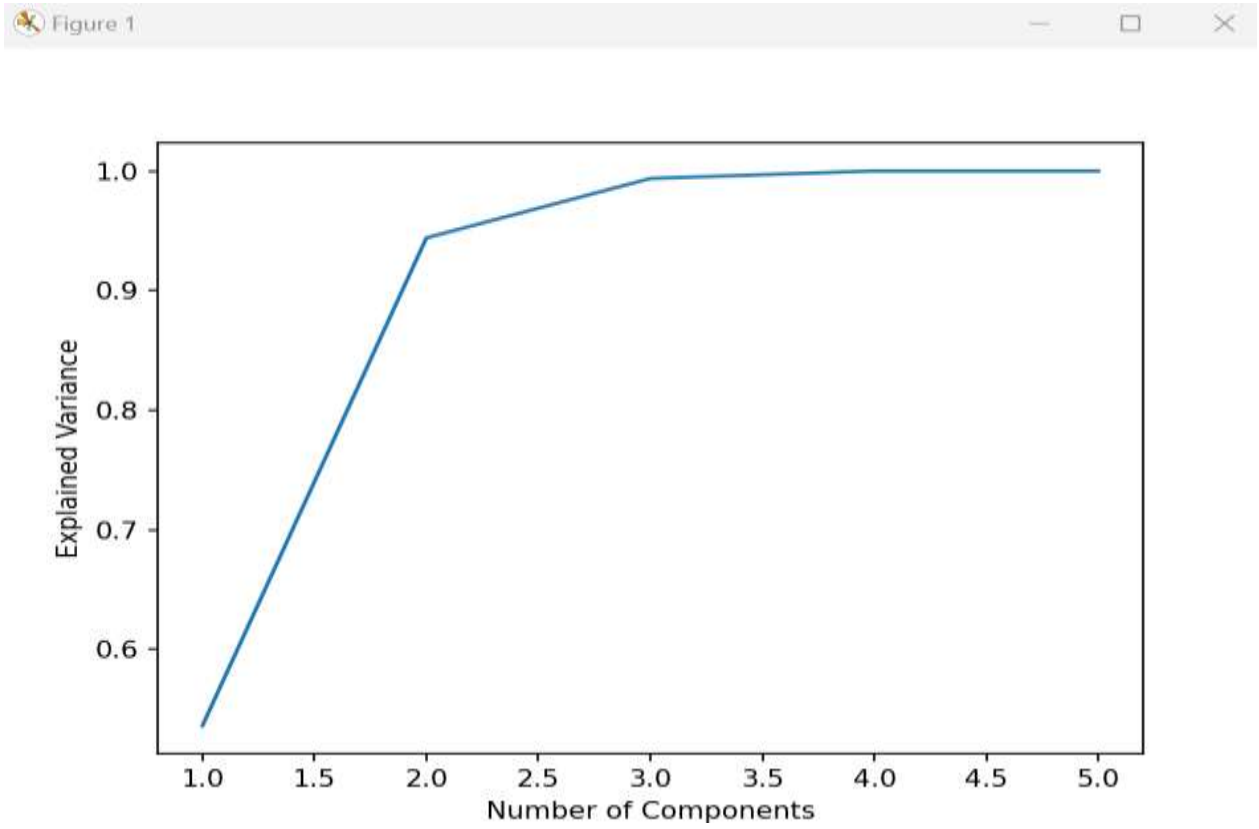
Output:

```
= RESTART: C:\Users\kamle\OneDrive\Desktop\ml\mlp1b.py
<_csv.reader object at 0x00000243F94B36A0>

The given training examples are:
['sky', 'airtemp', 'humidity', 'wind ', 'water', 'forecast', 'enjoysport']
['Sunny', 'Warm', 'Normal', 'Strong', 'Warm', 'Same', 'Yes']
['Sunny', 'Warm', 'High', 'Strong', 'Warm', 'Same', 'Yes']
['Rainy', 'Cold', 'High', 'Strong', 'Warm', 'Change', 'No']
['Sunny', 'Warm', 'High', 'Strong', 'Cool', 'Change', 'Yes']

The positive examples are:
['Sunny', 'Warm', 'Normal', 'Strong', 'Warm', 'Same', 'Yes']
['Sunny', 'Warm', 'High', 'Strong', 'Warm', 'Same', 'Yes']
['Sunny', 'Warm', 'High', 'Strong', 'Cool', 'Change', 'Yes']


The steps of the Find-s algorithm are :
 ['%', '%', '%', '%', '%', '%']
['Sunny', 'Warm', 'Normal', 'Strong', 'Warm', 'Same']
['Sunny', 'Warm', '?', 'Strong', 'Warm', 'Same']
['Sunny', 'Warm', '?', 'Strong', '?', '?']

The maximally specific Find-s hypothesis for the given training examples is :
['Sunny', 'Warm', '?', 'Strong', '?', '?']
```

PRATICAL 2A) Perform Data Loading, Feature selection (Principal Component analysis) and Feature Scoring and Ranking.

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA

# Data Loading
data = pd.read_csv("C:/Users/kamle/OneDrive/Desktop/ml/p2/data.csv") # Replace "data.csv" with your dataset file
X = data.drop('target_column', axis=1)  # Features
y = data['target_column']  # Target variable (if applicable)

# Feature Selection using PCA
pca = PCA()
pca.fit(X)
explained_var_ratio = pca.explained_variance_ratio_

# Plot the explained variance to decide on the number of components to keep
plt.plot(range(1, len(explained_var_ratio) + 1), explained_var_ratio.cumsum())
plt.xlabel('Number of Components')
plt.ylabel('Explained Variance')
plt.show()

# Choose the number of components that explain most of the variance
num_components = 5  # Adjust as needed

# Transform the data with the selected number of components
pca = PCA(n_components=num_components)
X_pca = pca.fit_transform(X)

# Feature Scoring and Ranking
loadings = pca.components_
feature_scores = abs(loadings).mean(axis=0)  # Use mean absolute loading values

# Create a DataFrame to display feature scores
feature_scores_df = pd.DataFrame({'Feature': X.columns, 'Score': feature_scores})

# Sort features by score in descending order
feature_scores_df = feature_scores_df.sort_values(by='Score', ascending=False)

# Display the ranked features
print(feature_scores_df)
```

Output:




```
= RESTART: C:\Users\kamle\OneDrive\Desktop\ml\p2\mlp2a.py
    Feature     Score
0  feature1  0.431647
4  feature5  0.384937
1  feature2  0.374804
3  feature4  0.328244
2  feature3  0.317932
>>
```

PRATICAL 2B) For a given set of training data examples stored in a .CSV file, implement and demonstrate the CandidateElimination algorithm to output a description of the set of all hypotheses consistent with the training examples.

```
import numpy as np
import pandas as pd
# Loading Data from a CSV File
data = pd.DataFrame(data=pd.read_csv('C:/Users/kamle/OneDrive/Desktop/ml/p2/trainingData.csv'))
print(data)
# Separating concept features from Target
concepts = np.array(data.iloc[:,0:-1])
print(concepts)
# Isolating target into a separate DataFrame
# copying last column to target array
target = np.array(data.iloc[:,-1])
print(target)
def learn(concepts, target):

    '''
    learn() function implements the learning method of the Candidate elimination algorithm.
    Arguments:
        concepts - a data frame with all the features
        target - a data frame with corresponding output values
    '''

    # Initialise S0 with the first instance from concepts
    # .copy() makes sure a new list is created instead of just pointing to the same memory location
    specific_h = concepts[0].copy()
    print("\nInitialization of specific_h and general_h")
    print(specific_h)
    #h=["#" for i in range(0,5)]
    #print(h)

    general_h = [["?" for i in range(len(specific_h))] for i in range(len(specific_h))]
    print(general_h)
    # The learning iterations
    for i, h in enumerate(concepts):

        # Checking if the hypothesis has a positive target
        if target[i] == "Yes":
            for x in range(len(specific_h)):

                # Change values in S & G only if values change
                if h[x] != specific_h[x]:
                    specific_h[x] = '?'
                    general_h[x][x] = '?'

        # Checking if the hypothesis has a positive target
        if target[i] == "No":
            for x in range(len(specific_h)):
```

```
            # For negative hyposthesis change values only in G
            if h[x] != specific_h[x]:
                general_h[x][x] = specific_h[x]
            else:
                general_h[x][x] = '?'

        print("\nSteps of Candidate Elimination Algorithm",i+1)
        print(specific_h)
        print(general_h)

    # find indices where we have empty rows, meaning those that are unchanged
    indices = [i for i, val in enumerate(general_h) if val == ['?', '?', '?', '?', '?', '?']]
    for i in indices:
        # remove those rows from general_h
        general_h.remove(['?', '?', '?', '?', '?', '?'])
    # Return final values
    return specific_h, general_h
s_final, g_final = learn(concepts, target)
print("\nFinal Specific_h:", s_final, sep="\n")
print("\nFinal General_h:", g_final, sep="\n")
```

Output:

```
= RESTART: C:\Users\kamle\OneDrive\Desktop\ml\p2\mlp2b.py
     sky airtemp humidity   wind  water forecast enjoysport
0  Sunny    Warm   Normal  Strong  Warm     Same       Yes
1  Sunny    Warm     High  Strong  Warm     Same       Yes
2  Rainy    Cold     High  Strong  Warm   Change        No
3  Sunny    Warm     High  Strong  Cool   Change       Yes
[['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
 ['Sunny' 'Warm' 'High' 'Strong' 'Warm' 'Same']
 ['Rainy' 'Cold' 'High' 'Strong' 'Warm' 'Change']
 ['Sunny' 'Warm' 'High' 'Strong' 'Cool' 'Change']]
['Yes' 'Yes' 'No' 'Yes']

Initialization of specific_h and general_h
['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
[['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?',
'?', '?', '?', '?', '?']]

Steps of Candidate Elimination Algorithm 1
['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
[['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?',
'?', '?', '?', '?', '?']]

Steps of Candidate Elimination Algorithm 2
['Sunny' 'Warm' '?' 'Strong' 'Warm' 'Same']
[['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?',
'?', '?', '?', '?', '?']]

Steps of Candidate Elimination Algorithm 3
['Sunny' 'Warm' '?' 'Strong' 'Warm' 'Same']
[['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'],
['?', '?', '?', '?', '?', 'Same']]

Steps of Candidate Elimination Algorithm 4
['Sunny' 'Warm' '?' 'Strong' '?' '?']
[['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'],
['?', '?', '?', '?', '?', '?']]

Final Specific_h:
['Sunny' 'Warm' '?' 'Strong' '?' '?']

Final General_h:
[['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?']]
```

PRATICAL 3A) Write a program to implement the naïve Bayesian classifier for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets.

```
# import necessary libarities
import pandas as pd
from sklearn import tree
from sklearn.preprocessing import LabelEncoder
from sklearn.naive_bayes import GaussianNB

# load data from CSV
data = pd.read_csv("C:/Users/kamle/OneDrive/Desktop/ml/p3/trainingData.csv")
print("THe first 5 values of data is :\n",data.head())

# obtain Train data and Train output
X = data.iloc[:,:-1]
print("\nThe First 5 values of train data is\n",X.head())

y = data.iloc[:,-1]
print("\nThe first 5 values of Train output is\n",y.head())

# Convert then in numbers
le_outlook = LabelEncoder()
X.Outlook = le_outlook.fit_transform(X.Outlook)

le_Temperature = LabelEncoder()
X.Temperature = le_Temperature.fit_transform(X.Temperature)

le_Humidity = LabelEncoder()
X.Humidity = le_Humidity.fit_transform(X.Humidity)

le_Windy = LabelEncoder()
X.Windy = le_Windy.fit_transform(X.Windy)

print("\nNow the Train data is :\n",X.head())

le_PlayTennis = LabelEncoder()
y = le_PlayTennis.fit_transform(y)
print("\nNow the Train output is\n",y)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.20)

classifier = GaussianNB()
classifier.fit(X_train,y_train)

from sklearn.metrics import accuracy_score
print("Accuracy is:",accuracy_score(classifier.predict(X_test),y_test))
```

Output:

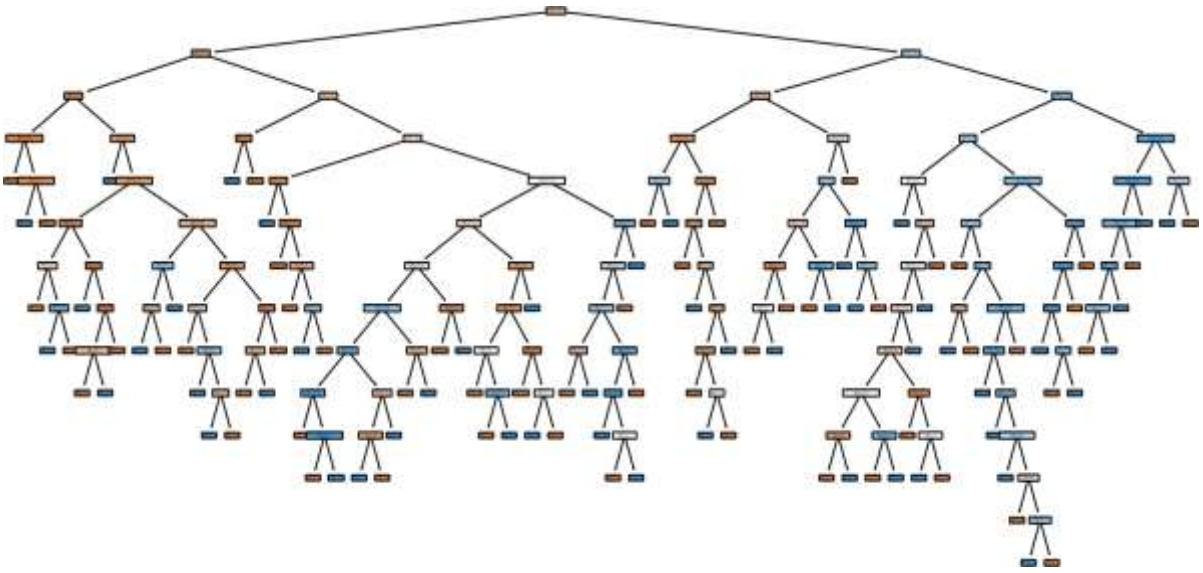
```
THe first 5 values of data is :
      sky airtemp humidity   wind  water forecast enjoysport
0  Sunny    Warm   Normal  Strong  Warm     Same        Yes
1  Sunny    Warm     High  Strong  Warm     Same        Yes
2  Rainy    Cold     High  Strong  Warm   Change         No
3  Sunny    Warm     High  Strong  Cool   Change        Yes

The First 5 values of train data is
      sky airtemp humidity   wind  water forecast
0  Sunny    Warm   Normal  Strong  Warm     Same
1  Sunny    Warm     High  Strong  Warm     Same
2  Rainy    Cold     High  Strong  Warm   Change
3  Sunny    Warm     High  Strong  Cool   Change

The first 5 values of Train output is
 0    Yes
1    Yes
2     No
3    Yes
Name: enjoysport, dtype: object
```

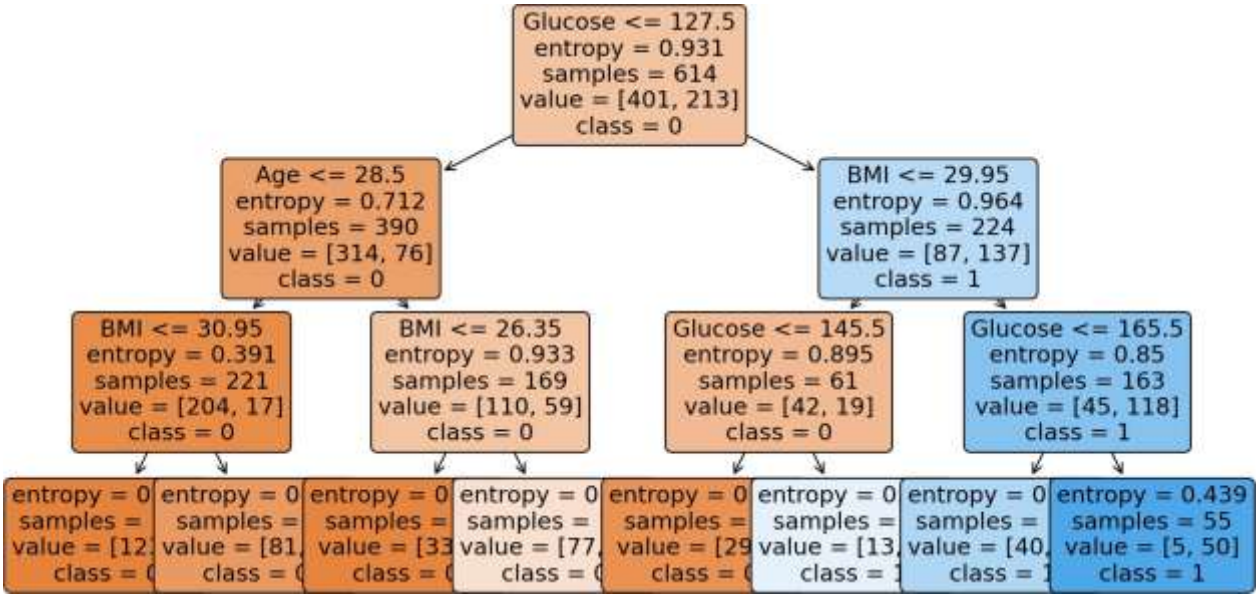
PRATICAL 3B) Write a program to implement Decision Tree and Random forest with Prediction, Test Score and Confusion Matrix.

```
# Import pandas library
import pandas as pd
# Loading dataset
df = pd.read_csv("C:/Users/kamle/OneDrive/Desktop/ml/p3/diabetes_dataset.csv")
df.head()

# Feature variables
x = df.drop(['Outcome'], axis=1)

# Target variable
y = df.Outcome

from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split

# Split the dataset
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)

# Create Decision Tree classifier object
model = DecisionTreeClassifier()

# Train Decision Tree Classifier
model.fit(x_train, y_train)

# Predict the response for the test dataset
y_pred = model.predict(x_test)

# Evaluation using Accuracy score
from sklearn import metrics
print("Accuracy:", metrics.accuracy_score(y_test, y_pred) * 100)

# Evaluation using Confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(cm)

# Accuracy calculation from Confusion matrix
accuracy = (cm[0, 0] + cm[1, 1]) / sum(sum(cm))
print("Accuracy from Confusion Matrix:", accuracy * 100)

# Evaluation using Classification report
from sklearn.metrics import classification_report
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

```
# Checking prediction value
prediction = model.predict([[6, 148, 72, 35, 0, 33.6, 0.627, 50]])
print("Prediction for input:", prediction)

# Import modules for Visualizing Decision trees
from sklearn.tree import export_graphviz
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

# Visualizing Decision Tree
plt.figure(figsize=(20, 10))
plot_tree(model, feature_names=x.columns, class_names=['0', '1'], filled=True, rounded=True)
plt.show()

# Create Decision Tree classifier object with entropy and max_depth
model = DecisionTreeClassifier(criterion="entropy", max_depth=3)

# Train Decision Tree Classifier
model.fit(x_train, y_train)

# Predict the response for the test dataset
y_pred = model.predict(x_test)

# Model Accuracy
print("Accuracy:", metrics.accuracy_score(y_test, y_pred) * 100)

# Better Decision Tree Visualization
plt.figure(figsize=(20, 10))
plot_tree(model, feature_names=x.columns, class_names=['0', '1'], filled=True, rounded=True)
plt.show()
```

Output:





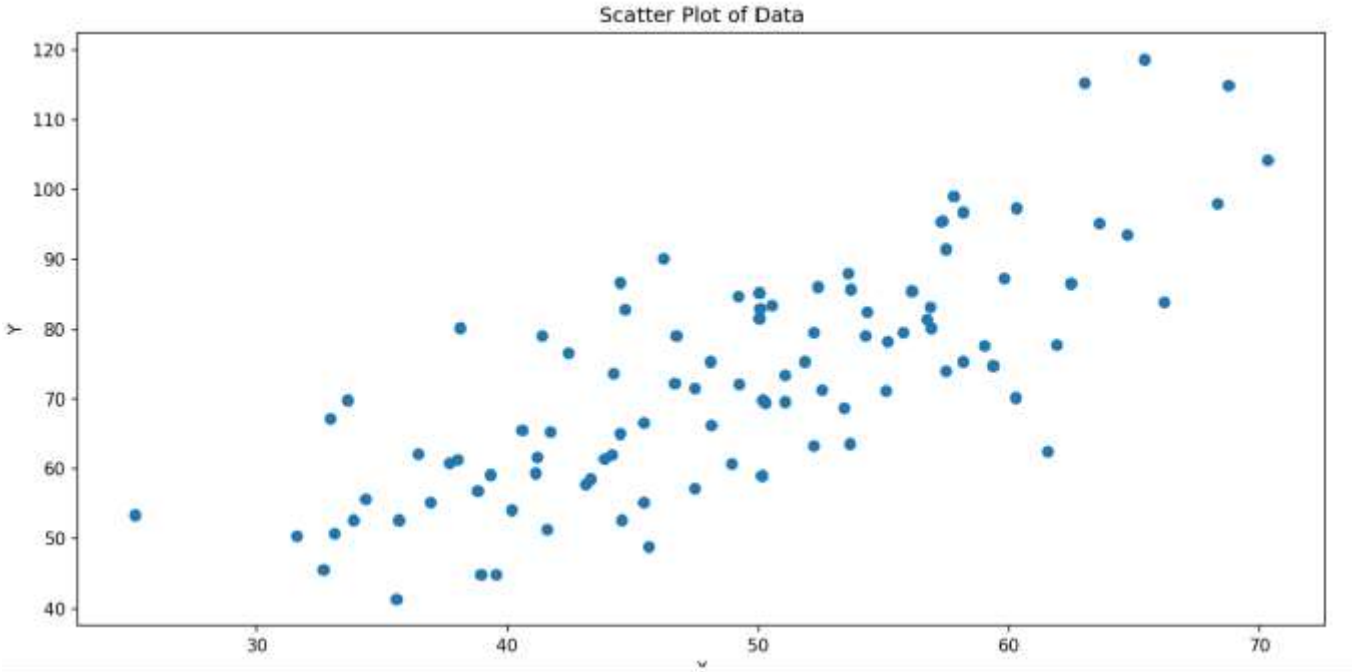

```
Accuracy: 75.32467532467533
Confusion Matrix:
[[76 23]
 [15 40]]
Accuracy from Confusion Matrix: 75.32467532467533
Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.77      0.80        99
           1       0.63      0.73      0.68        55

    accuracy                           0.75       154
   macro avg       0.74      0.75      0.74       154
weighted avg       0.76      0.75      0.76       154

Prediction for input: [1]
Accuracy: 76.62337662337663
```

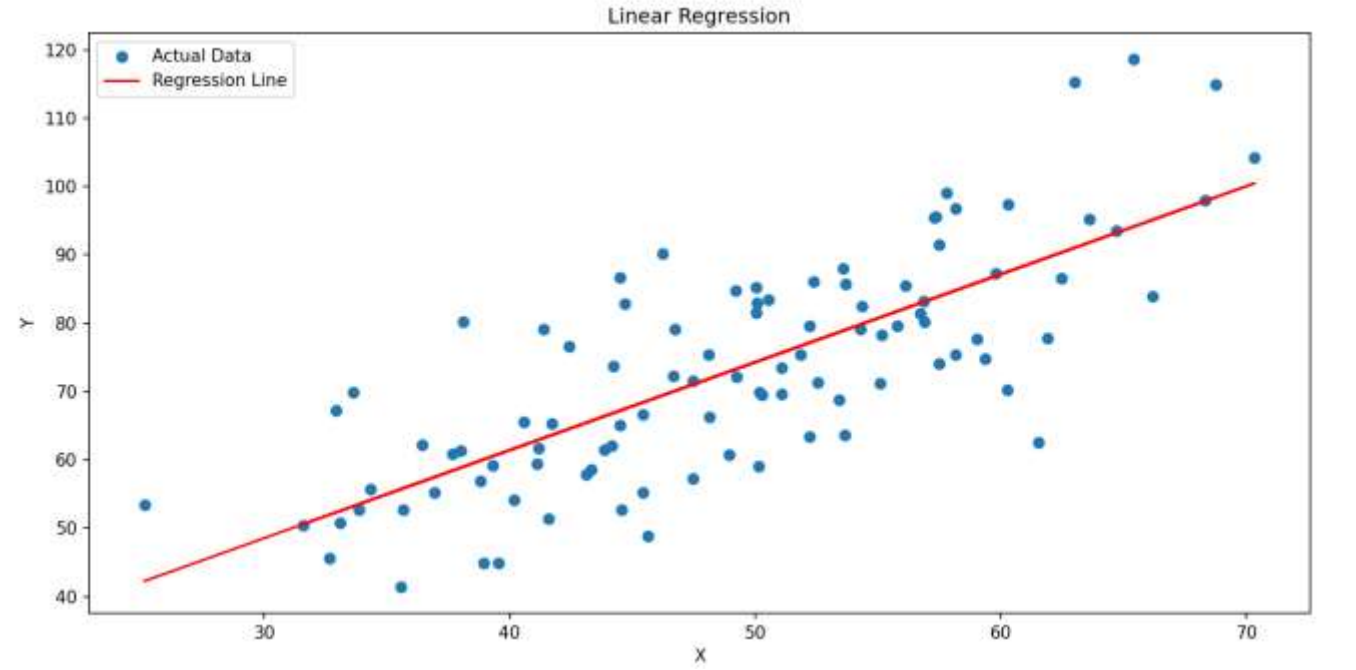
PRATICAL 4A) For a given set of training data examples stored in a .CSV file implement Least Square Regression algorithm.

```
# Importing Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Set the plot size
plt.rcParams['figure.figsize'] = (12.0, 9.0)

# Preprocessing Input Data
data = pd.read_csv('C:/Users/kamle/Downloads/ML/prac4/data.csv')

# Extracting the features X and Y
X = data.iloc[:, 0].values
Y = data.iloc[:, 1].values

# Plotting the data points
plt.scatter(X, Y)
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Scatter Plot of Data')
plt.show()

# Building the model
X_mean = np.mean(X)
Y_mean = np.mean(Y)

num = 0
den = 0

for i in range(len(X)):
    num += (X[i] - X_mean) * (Y[i] - Y_mean)
    den += (X[i] - X_mean) ** 2

m = num / den
c = Y_mean - m * X_mean

print("Slope (m):", m)
print("Intercept (c):", c)

# Making Predictions
Y_pred = m * X + c

# Plotting the regression line
plt.scatter(X, Y, label='Actual Data')
plt.plot(X, Y_pred, color='red', label='Regression Line')
plt.xlabel('X')
```

```
plt.ylabel('Y')
plt.title('Linear Regression')
plt.legend()
plt.show()
```

Output:






```
= RESTART: C:\Users\kamle\Downloads\ML\prac4\a. For a given set of training dat
 examples stored in a .CSV file implement Least Square Regression algorithm.py
Slope (m): 1.2873573699494276
Intercept (c): 9.90860619348318
```

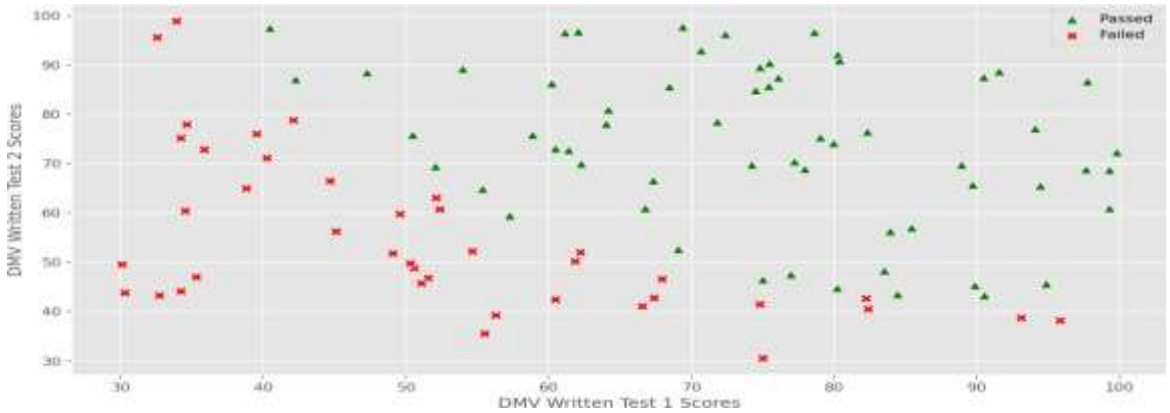
PRATICAL 4B) For a given set of training data examples stored in a .CSV file implement Logistic Regression algorithm.

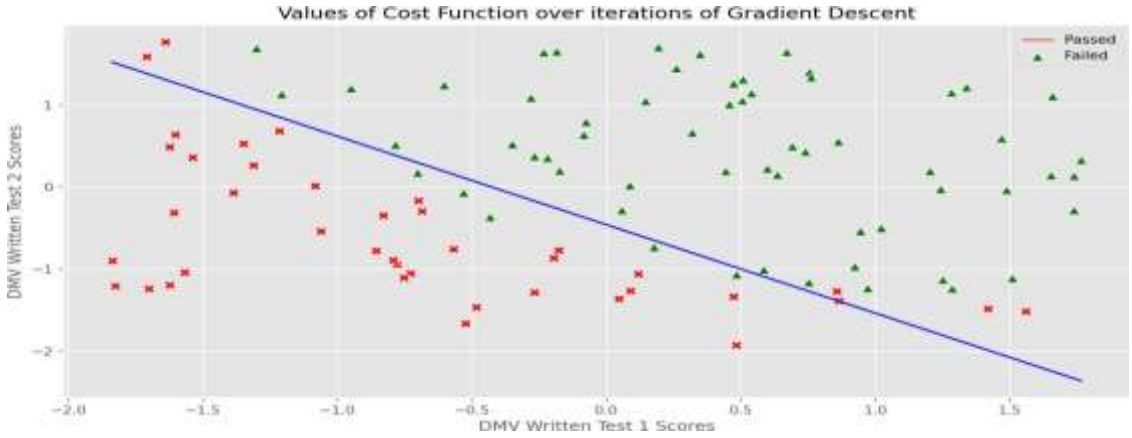
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
plt.style.use("ggplot")
#matplotlib inline

from pylab import rcParams
rcParams['figure.figsize'] = 12, 8

data = pd.read_csv("C:/Users/kamle/Downloads/ML/prac4/DMV_Written_Tests.csv")
data.head()

scores = data[['DMV_Test_1', 'DMV_Test_2']].values
results = data['Results'].values

passed = (results == 1).reshape(100, 1)
failed = (results == 0).reshape(100, 1)

ax = sns.scatterplot(x=scores[passed[:, 0], 0],
                     y=scores[passed[:, 0], 1],
                     marker="^",
                     color="green",
                     s=60)
sns.scatterplot(x=scores[failed[:, 0], 0],
                y=scores[failed[:, 0], 1],
                marker="X",
                color="red",
                s=60)

ax.set(xlabel="DMV Written Test 1 Scores", ylabel="DMV Written Test 2 Scores")
ax.legend(["Passed", "Failed"])
plt.show();

def logistic_function(x):
    return 1 / (1 + np.exp(-x))

logistic_function(0)

def compute_cost(theta, x, y):
    m = len(y)
    y_pred = logistic_function(np.dot(x, theta))
    error = (y * np.log(y_pred)) + ((1 - y) * np.log(1 - y_pred))
    cost = -1 / m * sum(error)
    gradient = 1 / m * np.dot(x.transpose(), (y_pred - y))
```

```
    return cost[0], gradient

mean_scores = np.mean(scores, axis=0)
std_scores = np.std(scores, axis=0)
scores = (scores - mean_scores) / std_scores  # standardization

rows = scores.shape[0]
cols = scores.shape[1]

X = np.append(np.ones((rows, 1)), scores, axis=1) # include intercept
y = results.reshape(rows, 1)

theta_init = np.zeros((cols + 1, 1))
cost, gradient = compute_cost(theta_init, X, y)

print("Cost at initialization", cost)
print("Gradient at initialization:", gradient)

def gradient_descent(x, y, theta, alpha, iterations):
    costs = []
    for i in range(iterations):
        cost, gradient = compute_cost(theta, x, y)
        theta -= (alpha * gradient)
    return theta, costs

theta, costs = gradient_descent(X, y, theta_init, 1, 200)
print("Theta after running gradient descent:", theta)

plt.plot(costs)
plt.xlabel("Iterations")
plt.ylabel("$J(\\Theta)$")
plt.title("Values of Cost Function over iterations of Gradient Descent");

sns.scatterplot(x=X[passed[:, 0], 1],
                y=X[passed[:, 0], 2],
                marker="^",
                color="green",
                s=60)
ax = sns.scatterplot(x=X[failed[:, 0], 1],
                     y=X[failed[:, 0], 2],
                     marker="X",
                     color="red",
                     s=60)

ax.legend(["Passed", "Failed"])
ax.set(xlabel="DMV Written Test 1 Scores", ylabel="DMV Written Test 2 Scores")

x_boundary = np.array([np.min(X[:, 1]), np.max(X[:, 1])])
y_boundary = -(theta[0] + theta[1] * x_boundary) / theta[2]

sns.lineplot(x=x_boundary, y=y_boundary, color="blue")
plt.show();
```

```
def predict(theta, x):
    results = x.dot(theta)
    return results > 0

p = predict(theta, X)
print("Training Accuracy:", sum(p == y)[0], "%")

test = np.array([50, 79])
test = (test - mean_scores) / std_scores
test = np.append(np.ones(1), test)
probability = logistic_function(test.dot(theta))
print("A person who scores 50 and 79 on their DMV written tests have a",
      np.round(probability[0], 2), "probability of passing.")
```

Output:






```
Cost at initialization 0.693147180559946
Gradient at initialization: [[-0.1       ]
 [-0.28122914]
 [-0.25098615]]
Theta after running gradient descent: [[1.50850586]
 [3.5468762 ]
 [3.29383709]]
Training Accuracy: 89 %
A person who scores 50 and 79 on their DMV written tests have a 0.71 probability of passing.
```

PRATICAL 5A) Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample.

```
import numpy as np
import math
import csv

def read_data(filename):
    with open(filename, 'r') as csvfile:
        datareader = csv.reader(csvfile, delimiter=',')
        headers = next(datareader)
        metadata = []
        traindata = []
        for name in headers:
            metadata.append(name)
        for row in datareader:
            traindata.append(row)

    return (metadata, traindata)

class Node:
    def__init__(self, attribute):
        self.attribute = attribute
        self.children = []
        self.answer = ""

    def__str_(self):
        return self.attribute

def subtables(data, col, delete):
    dict = {}
    items, counts = np.unique(data[:, col], return_counts=True)

    for x in range(items.shape[0]):
        dict[items[x]] = data[data[:, col] == items[x]]
        if delete:
            dict[items[x]] = np.delete(dict[items[x]], col, 1)

    return items, dict

def entropy(S):
    items, counts = np.unique(S, return_counts=True)
    total_size = S.size
    entropies = np.zeros(counts.shape)

    for x in range(items.shape[0]):
        ratio = counts[x] / total_size
```

```
        entropies[x] = ratio * math.log(ratio, 2)

    return -np.sum(entropies)

def gain_ratio(data, col):
    items, dict = subtables(data, col, delete=False)

    total_size = data.shape[0]
    entropies = np.zeros((items.shape[0], 1))
    intrinsic = np.zeros((items.shape[0], 1))

    for x in range(items.shape[0]):
        ratio = dict[items[x]].shape[0] / total_size
        entropies[x] = ratio * entropy(dict[items[x]][:, -1])
        intrinsic[x] = ratio * math.log(ratio, 2)

    total_entropy = entropy(data[:, -1])
    iv = -np.sum(intrinsic)

    for x in range(entropies.shape[0]):
        total_entropy -= entropies[x]

    return total_entropy / iv

def create_node(data, metadata):
    if data.size == 0:
        node = Node("")
        node.answer = "No data"
        return node

    if (np.unique(data[:, -1])).shape[0] == 1:
        node = Node("")
        node.answer = np.unique(data[:, -1])[0]
        return node

    gains = np.zeros((data.shape[1] - 1, 1))

    for col in range(data.shape[1] - 1):
        gains[col] = gain_ratio(data, col)

    split = np.argmax(gains)

    node = Node(metadata[split])
    metadata = np.delete(metadata, split, 0)

    items, dict = subtables(data, split, delete=True)

    for x in range(items.shape[0]):
        child = create_node(dict[items[x]], metadata)
        node.children.append((items[x], child))

    return node
```

```
def empty(size):
    s = ""
    for x in range(size):
        s += "  "
    return s

def print_tree(node, level):
    if node.answer != "":
        print(empty(level), "Answer:", node.answer)
        return
    print(empty(level), "Attribute:", node.attribute)
    for value, n in node.children:
        print(empty(level + 1), f"Value: {value}")
        print_tree(n, level + 2)

metadata, traindata = read_data("C:/Users/kamle/Downloads/ML/prac5/tennisdata.csv")
data = np.array(traindata)
node = create_node(data, metadata)
print_tree(node, 0)
```

Output:

```
 Attribute: Outlook
    Value: Overcast
       Answer: Yes
    Value: Rainy
       Attribute: Windy
          Value: False
             Answer: Yes
          Value: True
             Answer: No
    Value: Sunny
       Attribute: Humidity
          Value: High
             Answer: No
          Value: Normal
             Answer: Yes
```

PRATICAL 5B) Write a program to implement k-Nearest Neighbour algorithm to classify the iris data set.

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import datasets
iris=datasets.load_iris()
print("Iris Data set loaded...")
x_train, x_test, y_train, y_test = train_test_split(iris.data,iris.target,test_size=0.1)
#random_state=0
for i in range(len(iris.target_names)):
    print("Label", i , "-",str(iris.target_names[i]))
classifier = KNeighborsClassifier(n_neighbors=2)
classifier.fit(x_train, y_train)
y_pred=classifier.predict(x_test)
print("Results of Classification using K-nn with K=1 ")
for r in range(0,len(x_test)):
    print(" Sample:", str(x_test[r]), " Actual-label:", str(y_test[r])," Predicted-label:", str(y_pred[r]))
    print("Classification Accuracy :" , classifier.score(x_test,y_test));
```

Output:

```
Iris Data set loaded...
Label 0 - setosa
Label 1 - versicolor
Label 2 - virginica
Results of Classification using K-nn with K=1
 Sample: [4.4 3.2 1.3 0.2]  Actual-label: 0  Predicted-label: 0
Classification Accuracy : 1.0
 Sample: [4.6 3.4 1.4 0.3]  Actual-label: 0  Predicted-label: 0
Classification Accuracy : 1.0
 Sample: [6.1 2.8 4.7 1.2]  Actual-label: 1  Predicted-label: 1
Classification Accuracy : 1.0
 Sample: [5.4 3.7 1.5 0.2]  Actual-label: 0  Predicted-label: 0
Classification Accuracy : 1.0
 Sample: [6.3 3.3 4.7 1.6]  Actual-label: 1  Predicted-label: 1
Classification Accuracy : 1.0
 Sample: [5.7 4.4 1.5 0.4]  Actual-label: 0  Predicted-label: 0
Classification Accuracy : 1.0
 Sample: [6.4 2.8 5.6 2.1]  Actual-label: 2  Predicted-label: 2
Classification Accuracy : 1.0
 Sample: [5.7 2.9 4.2 1.3]  Actual-label: 1  Predicted-label: 1
Classification Accuracy : 1.0
 Sample: [5.  3.5 1.6 0.6]  Actual-label: 0  Predicted-label: 0
Classification Accuracy : 1.0
 Sample: [5.2 3.4 1.4 0.2]  Actual-label: 0  Predicted-label: 0
Classification Accuracy : 1.0
 Sample: [5.4 3.9 1.7 0.4]  Actual-label: 0  Predicted-label: 0
Classification Accuracy : 1.0
 Sample: [6.3 2.7 4.9 1.8]  Actual-label: 2  Predicted-label: 2
Classification Accuracy : 1.0
 Sample: [6.7 3.  5.2 2.3]  Actual-label: 2  Predicted-label: 2
Classification Accuracy : 1.0
 Sample: [6.5 3.  5.5 1.8]  Actual-label: 2  Predicted-label: 2
Classification Accuracy : 1.0
 Sample: [5.7 3.8 1.7 0.3]  Actual-label: 0  Predicted-label: 0
Classification Accuracy : 1.0
```

PRATICAL 6A) Implement the different Distance methods (Euclidean) with Prediction, Test Score and Confusion Matrix.

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('C:/Users/kamle/Downloads/ML/prac6/Social_Network_Ads.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, -1].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Training the K-NN model on the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test, y_pred)

# Print the results
print("Confusion Matrix:")
print(cm)
print("\nAccuracy Score:", ac)
```

Output:

```
Confusion Matrix:
[[55  3]
 [ 1 21]]

Accuracy Score: 0.95
```

PRATICAL 6B) Implement the classification model using clustering for the following techniques with K means clustering with Prediction, Test Score and Confusion Matrix

```
# read in the iris data
from sklearn.datasets import load_iris
iris = load_iris()

# create X (features) and y (response)
X = iris.data
y = iris.target

# Import the class
from sklearn.linear_model import LogisticRegression
# instantiate the model (using the default parameters)
logreg = LogisticRegression()

# fit the model with data
logreg.fit(X, y)

# predict the response values for the observations in X
logreg.predict(X)

# Store the predicted response values
y_pred = logreg.predict(X)

# Check how many Predictions were generated
len(y_pred)

# Computer classification accuracy for the logistic regression model
from sklearn import metrics

print(metrics.accuracy_score(y, y_pred))

from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X, y)
y_pred = knn.predict(X)
print(metrics.accuracy_score(y, y_pred))

knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X, y)
y_pred = knn.predict(X)
print(metrics.accuracy_score(y, y_pred))

# print the shapes of X and y
# X is our features matrix with 150 x 4 dimensions
print(X.shape)
# y is our response vector with 150 x 1 dimension
print(y.shape)
```

```
# STEP 1: split X and y into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=4)

# print the shapes of the new X objects
print(X_train.shape)
print(X_test.shape)

# print the shapes of the new y objects
print(y_train.shape)
print(y_test.shape)

# STEP 2: train the model on the training set
logreg = LogisticRegression()
logreg.fit(X_train, y_train)

# STEP 3: make predictions on the testing set
y_pred = logreg.predict(X_test)

# Computer actual response values (y_test) with predicted response values (y_pred)
print(metrics.accuracy_score(y_test, y_pred))

knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
print(metrics.accuracy_score(y_test, y_pred))

# try k-1 through k-25 and record testing accuracy
k_range = range(1, 26)

# we can create python dictionary using [] or dict()
scores = []

# we use a loop through the range 1 to 26
# we append the scores in the dictionary
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    scores.append(metrics.accuracy_score(y_test, y_pred))

print(scores)

# import Matplotlib (scientific plotting library)
import matplotlib.pyplot as plt

# plot the relationship between k and testing accuracy
plt.plot(k_range, scores)
plt.xlabel('Value of k for KNN')
plt.ylabel('Testing Accuracy')
```

```
# instantiate the model with the best-known parameters
knn = KNeighborsClassifier(n_neighbors=11)

# train the model with X and y (not X_train and y_train)
knn.fit(X, y)

# make a prediction for an out-of-sample observation
knn.predict([[3, 5, 4, 3]])
```

Output:

```
0.9733333333333334
0.9666666666666667
1.0
(150, 4)
(150,)
(90, 4)
(60, 4)
(90,)
(60,)
0.9666666666666667
0.9666666666666667
[0.95, 0.95, 0.9666666666666667, 0.9666666666666667, 0.9666666666666667, 0.9833333333333333, 0.9833333333333333, 0.9833333333333333, 0.9833333333333333, 0.9833333333333333, 0.9833333333333333, 0.9833333333333333, 0.9833333333333333, 0.9833333333333333, 0.9666666666666667, 0.9833333333333333, 0.9666666666666667, 0.9666666666666667, 0.9666666666666667, 0.9666666666666667, 0.95, 0.95]
```

PRATICAL 7A) Implement the classification model using clustering for the following techniques with hierarchical clustering with Prediction, Test Score and Confusion Matrix.

```
import matplotlib.pyplot as plt
import pandas as pd
import scipy.cluster.hierarchy as sch
from sklearn.cluster import AgglomerativeClustering

dataset = pd.read_csv('C:/Users/kamle/Downloads/ML/prac7/abalone_csv.csv')
X = dataset.iloc[:, [3, 4]].values

# Dendrogram
dendrogram = sch.dendrogram(sch.linkage(X, method="ward"))
plt.title('Dendrogram')
plt.xlabel('Sample Index')
plt.ylabel('Euclidean Distances')
plt.show()

# Hierarchical Clustering
hc = AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='ward')
y_hc = hc.fit_predict(X)
print("Prediction Values: ", y_hc)
# Scatter Plot
plt.scatter(X[y_hc == 0, 0], X[y_hc == 0, 1], s=100, c='red', label='Cluster 1')
plt.scatter(X[y_hc == 1, 0], X[y_hc == 1, 1], s=100, c='blue', label='Cluster 2')
plt.scatter(X[y_hc == 2, 0], X[y_hc == 2, 1], s=100, c='green', label='Cluster 3')
plt.scatter(X[y_hc == 3, 0], X[y_hc == 3, 1], s=100, c='cyan', label='Cluster 4')
plt.scatter(X[y_hc == 4, 0], X[y_hc == 4, 1], s=100, c='magenta', label='Cluster 5')

# Plot Settings
plt.title('Clusters of Abalones (Hierarchical Clustering Model)')
plt.xlabel('Annual income (k$)') # Update with appropriate label
plt.ylabel('Spending score(1 to 100)') # Update with appropriate label
plt.legend()
plt.show()
```

Output:

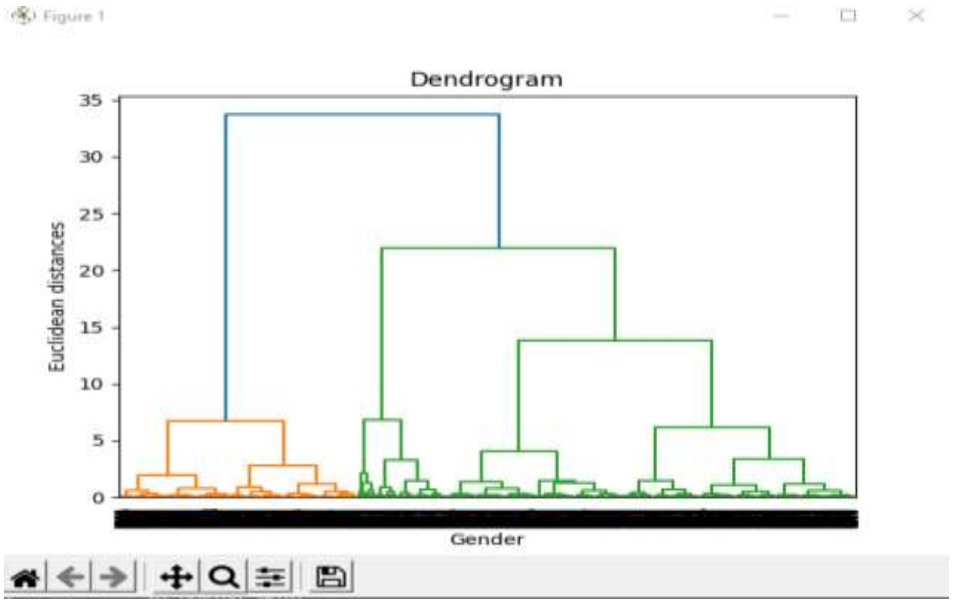


Figure 1
Dendrogram
Euclidean distances
35
30
25
20
15
10
5
0
Gender

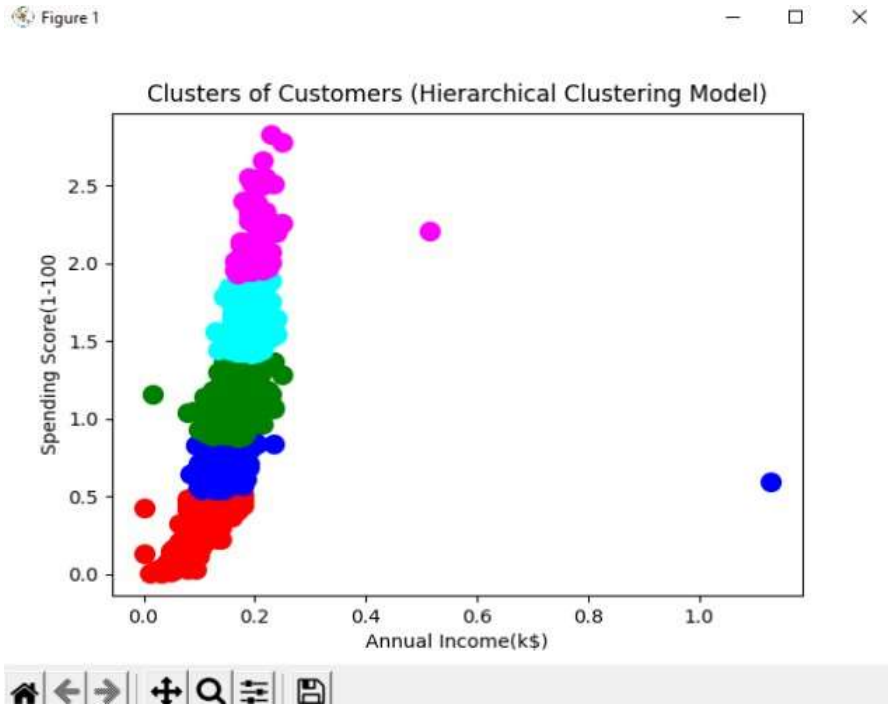


Figure 1
Clusters of Customers (Hierarchical Clustering Model)
Spending Score(1-100
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Annual Income(k$)

PRATICAL 7B) Perform Text pre-processing, Text clustering, classification with Prediction, Test Score and Confusion Matrix.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# csv_file = 'Restaurant_Reviews.tsv'
csv_file = 'C:/Users/kamle/Downloads/ML/prac7/Restaurant_Reviews.tsv'
dataset = pd.read_csv(csv_file, delimiter='\t', quoting=3)
# Text preprocessing using Natural Language Toolkit (nltk)
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
corpus = []
for i in range(0, 1000):
    review = re.sub('[^a-zA-Z]', '', dataset['Review'][i]) # Remove non-alphabetic characters
    review = review.lower()  # Convert to lowercase
    review = review.split() # Tokenize the words
    ps = PorterStemmer() # Stemming and removing stopwords
    review = [ps.stem(word) for word in review if not word in set(stopwords.words('english'))]
    # Join the words to form the processed review
    review = ' '.join(review)
    corpus.append(review)
# Creating the bag of words model
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=1500)
X = cv.fit_transform(corpus).toarray()
Y = dataset.iloc[:, 1].values
# Splitting the dataset into the training set and test set
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=100)
# Fitting Naive Bayes to the training set
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, Y_train)
# Predicting the test set results
```

```
Y_pred = classifier.predict(X_test)
# Model Accuracy
from sklearn import metrics
from sklearn.metrics import confusion_matrix
print("Accuracy:", metrics.accuracy_score(Y_test, Y_pred))
# Making the confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test, Y_pred)
print(cm)
```

Output:

```
Accuracy: 0.54
[[  1 115]
 [  0 134]]
>
```

PRATICAL 8A) Write a program to construct a Bayesian network considering medical data. Use this model to demonstrate the diagnosis of heart patients using standard Heart Disease Data Set.

```
import numpy as np
import pandas as pd
from pgmpy.models import BayesianModel
from pgmpy.estimators import MaximumLikelihoodEstimator
from pgmpy.inference import VariableElimination

# Read Cleveland Heart Disease data
heartDisease = pd.read_csv('C:/Users/kamle/Downloads/ML/prac8/heart.csv')
heartDisease = heartDisease.replace('?', np.nan)

# Display the data
print('Few examples from the dataset are given below')
print(heartDisease.head())

# Model Bayesian Network
model = BayesianModel([('age', 'trestbps'), ('age', 'fbs'),
                       ('sex', 'trestbps'), ('exang', 'trestbps'), ('trestbps', 'heartdisease'),
                       ('fbs', 'heartdisease'), ('heartdisease', 'restecg'),
                       ('heartdisease', 'thalach'), ('heartdisease', 'chol')])

# Learning CPDs using Maximum Likelihood Estimators
print('\nLearning CPD using Maximum likelihood estimators')
model.fit(heartDisease, estimator=MaximumLikelihoodEstimator)

# Inferencing with Bayesian Network
print('\nInferencing with Bayesian Network:')
HeartDisease_infer = VariableElimination(model)

# Computing the Probability of HeartDisease given Age
print('\n1. Probability of HeartDisease given Age=28')
q = HeartDisease_infer.query(variables=['heartdisease'], evidence={'age': 28})
print(q['heartdisease'])

# Computing the Probability of HeartDisease given cholesterol
```

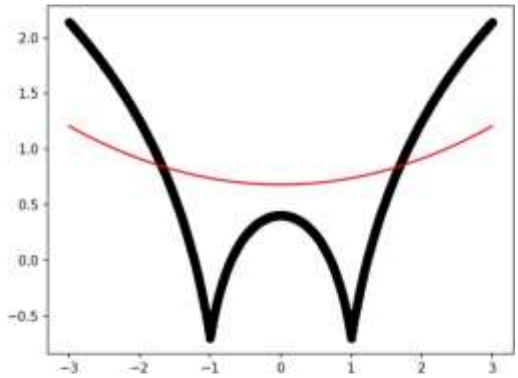
```
print('\n2. Probability of HeartDisease given cholesterol=100')
q = HeartDisease_infer.query(variables=['heartdisease'], evidence={'chol': 100})
print(q['heartdisease'])
```

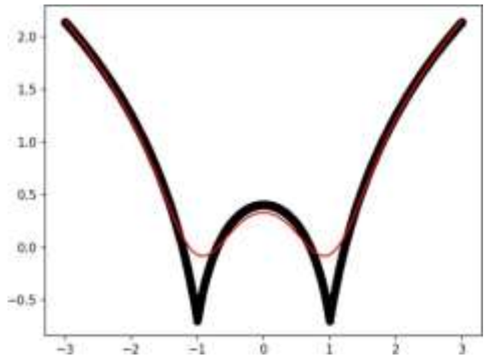
Output:

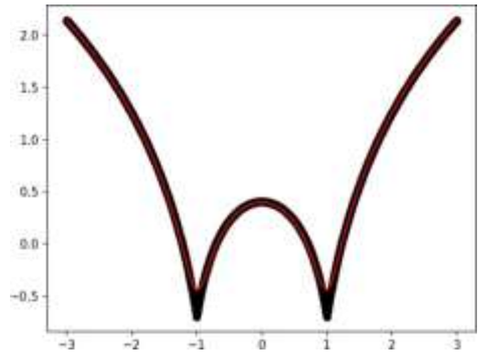
PRATICAL 8B) Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs.

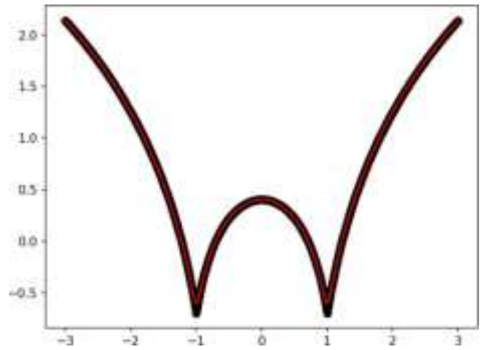
```
import numpy as np
import matplotlib.pyplot as plt
def local_regression(x0, X, Y, tau):
    x0 = [1, x0]
    X = [[1, i] for i in X]
    X = np.asarray(X)
    xw = (X.T) * np.exp(np.sum((X - x0) ** 2, axis=1) / (-2 * tau))
    beta = np.linalg.pinv(xw @ X) @ xw @ Y @ x0
    return beta
def draw(tau):
    prediction = [local_regression(x0, X, Y, tau) for x0 in domain]
    plt.plot(X, Y, 'o', color='black')
    plt.plot(domain, prediction, color='red')
    plt.show()
X = np.linspace(-3, 3, num=1000)
domain = X
Y = np.log(np.abs(X ** 2 - 1) + .5)
draw(10)
draw(0.1)
draw(0.01)
draw(0.001)
```

Output:


2.0
1.5
1.0
0.5
0.0
−0.5
−3 −2 −1 0 1 2 3


2.0
1.5
1.0
0.5
0.0
−0.5
−3 −2 −1 0 1 2 3


2.0
1.5
1.0
0.5
0.0
−0.5
−3 −2 −1 0 1 2 3


2.0
1.5
1.0
0.5
0.0
−0.5
−3 −2 −1 0 1 2 3

Robotic Process Automation

INDEX

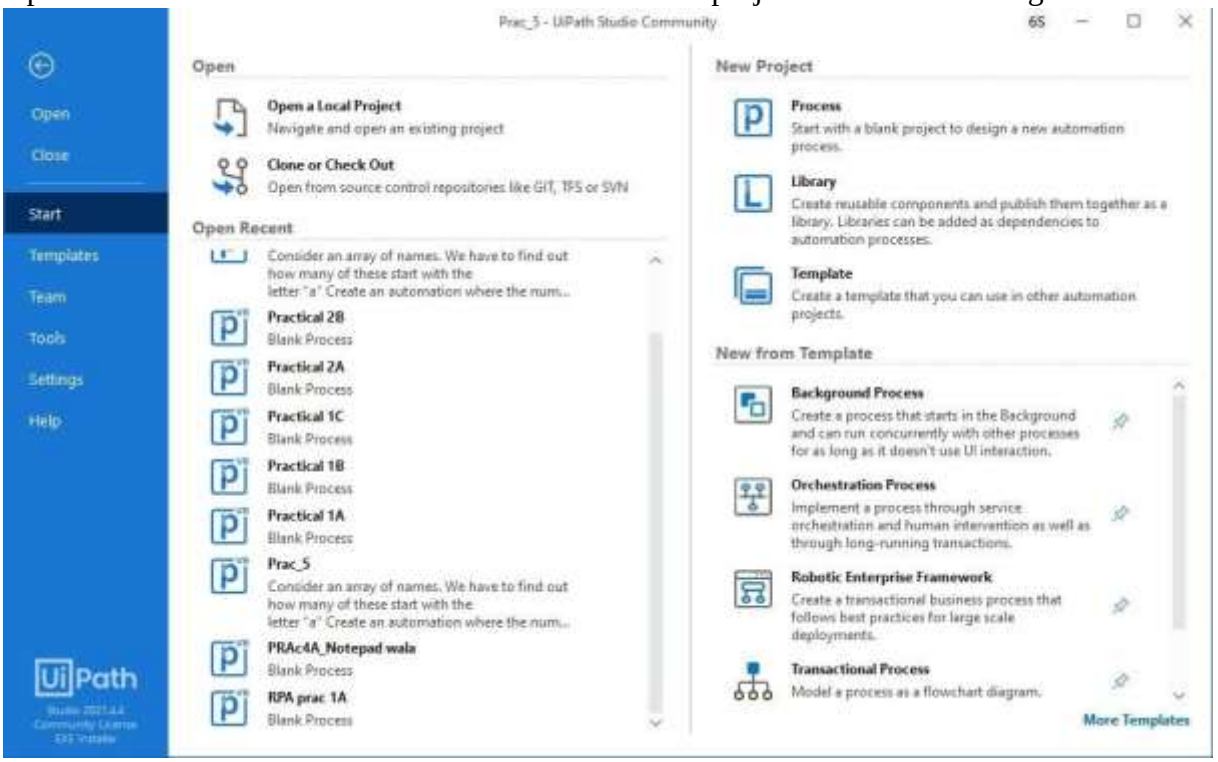
Practical no 1

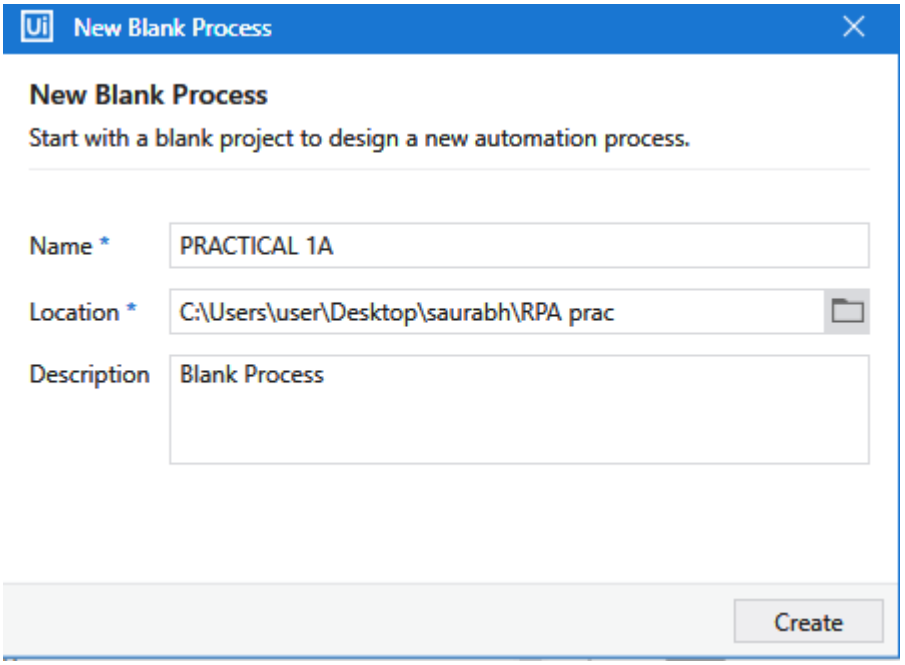
1 A) Create a simple sequence-based project.

Aim: Use two input dialogs for First Name and Last Name store in a variable and show in Message Box.

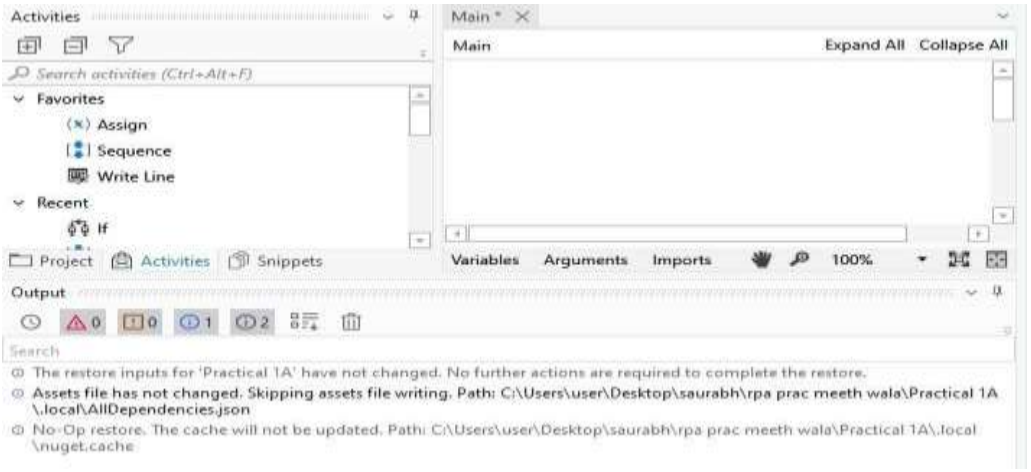
Steps:

1. Open UiPath Studio and click on **Blank** to start a fresh project. Give it a meaningful name.



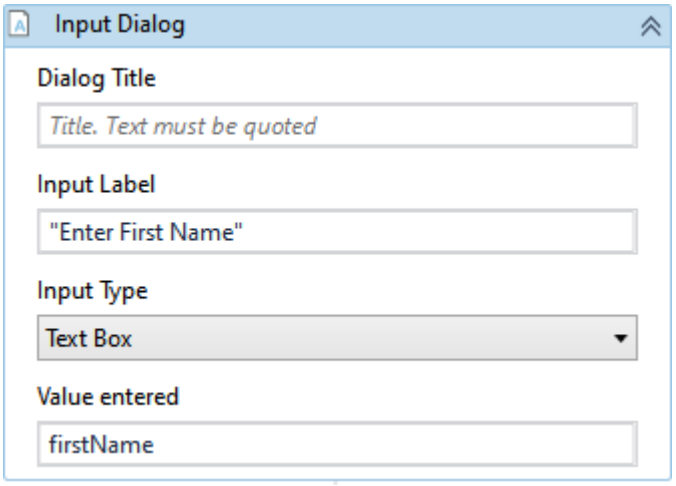


2. On the Designer panel, drag and drop a **Sequence** activity from the **Activities** panel.

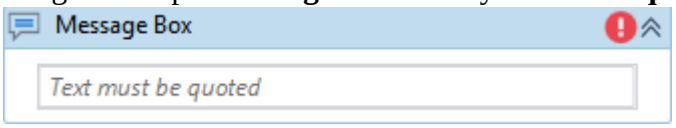


3. Search for **Input dialog** in the Search panel of the **Activities** panel. Drag and drop the **Input dialog** activity inside the **Sequence**.

   Write the appropriate message on the **Label** of this **Input dialog** to ask for the user's name. In our case, we have put in "Enter First Name"
4. Drag another **Input dialog** activity into the **Sequence**.

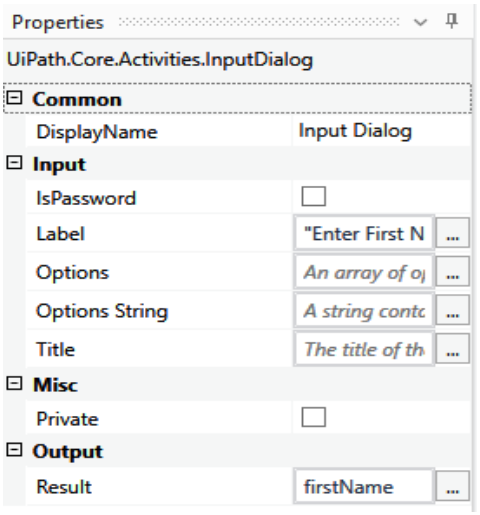


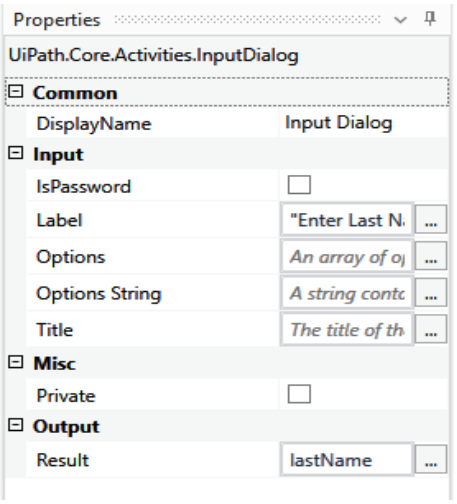
   Write the Label as "Enter Last Name"
5. Drag and drop a **Message box** activity into the **Sequence**.



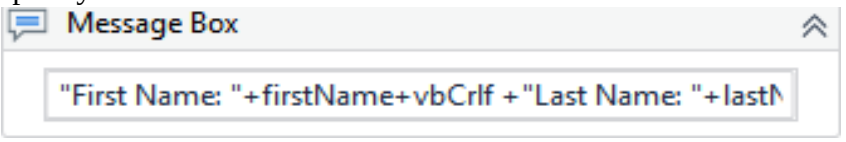
6. Next, create two **variables** and give them the desired names. These variables will receive the text that the user has entered in the **Input dialog** boxes.

7. We now have to specify the **Result** property (in the **Properties** panel) of the **Input dialog** box. On specifying the variable name there, it will receive the text that the user entered.

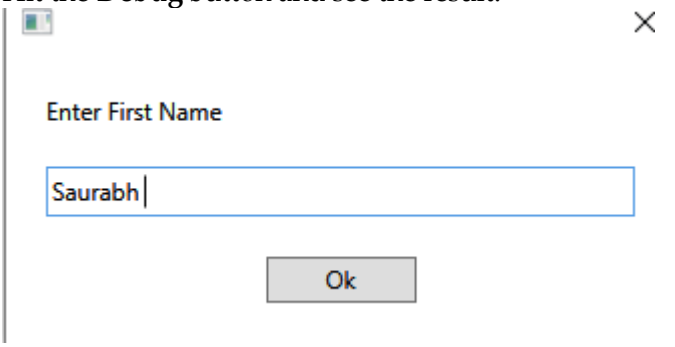


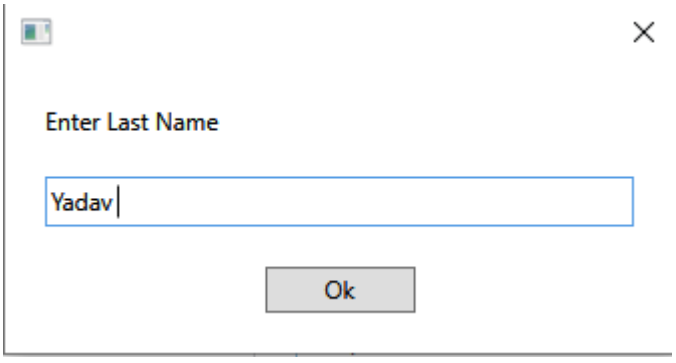


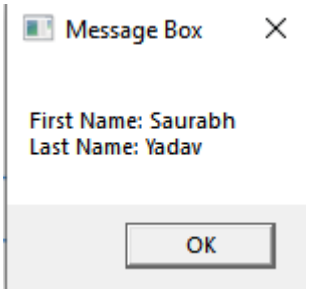
8. Specify the variables that we have created with other text in the Text area of the **Message box**.



9. Hit the **Debug** button and see the result.





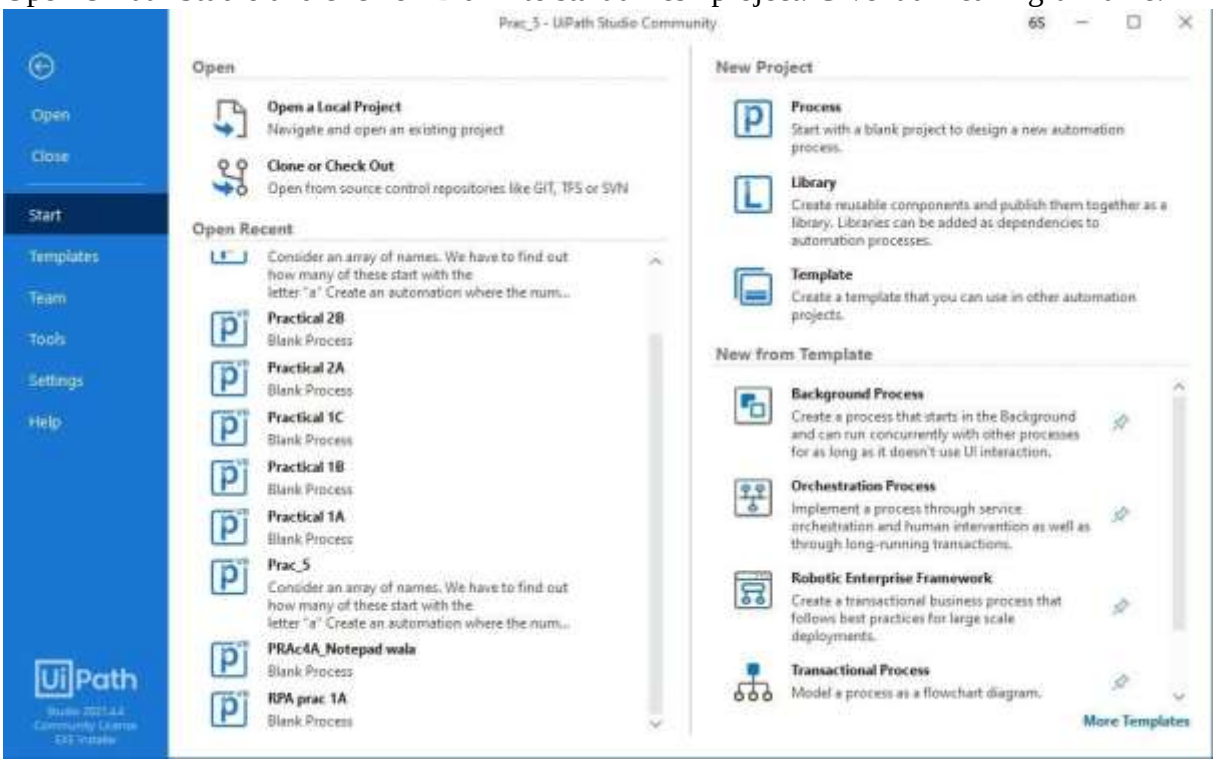


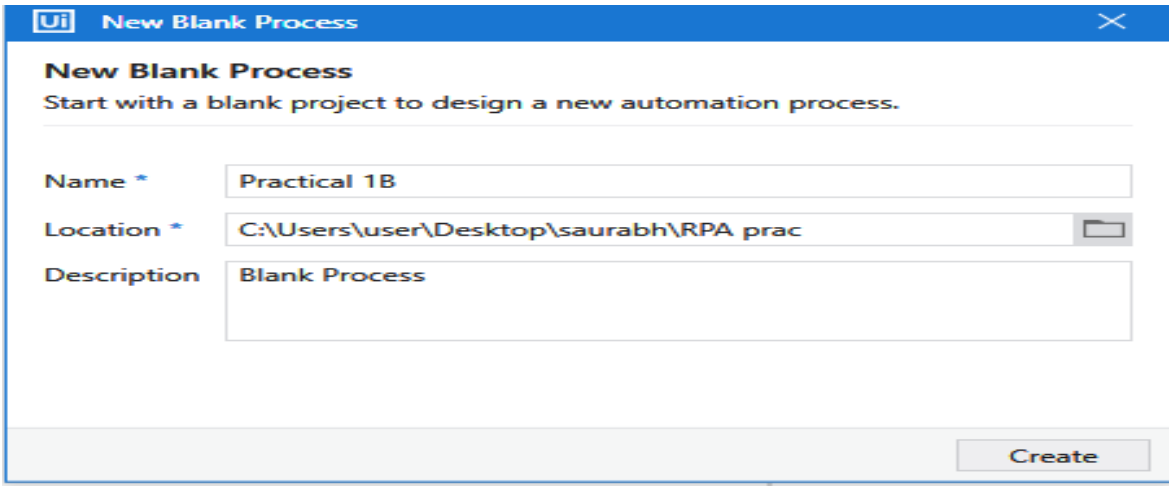
1 B) Create a simple flowchart-based project

Aim: Use two input dialogs for First Name and Last Name store in a variable and show in Message Box.

Steps:

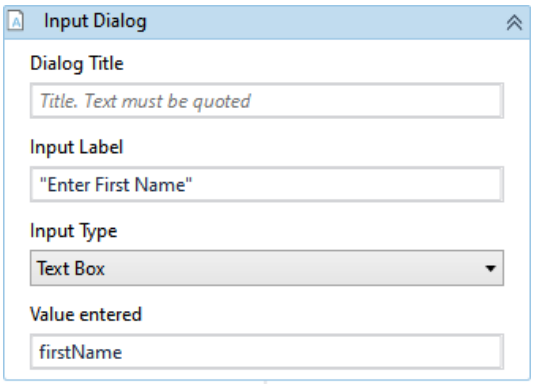
1. Open UiPath Studio and click on **Blank** to start a fresh project. Give it a meaningful name.





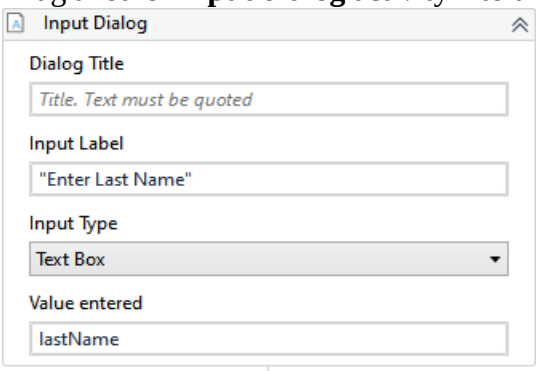
2. On the Designer panel, drag and drop a **Flowchart** activity from the **Activities** panel. Double click into Flowchart and drag **Sequence** activity.

3. Search for **Input dialog** in the Search panel of the **Activities** panel. Drag and drop the **Input dialog** activity inside the **Sequence**.



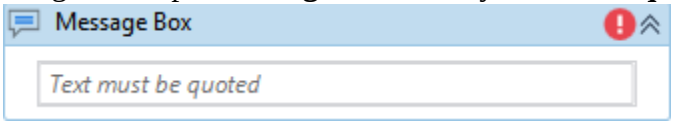
Write the appropriate message on the **Label** of this **Input dialog** to ask for the user's name. In our case, we have put in "Enter First Name"

4. Drag another **Input dialog** activity into the **Sequence**.



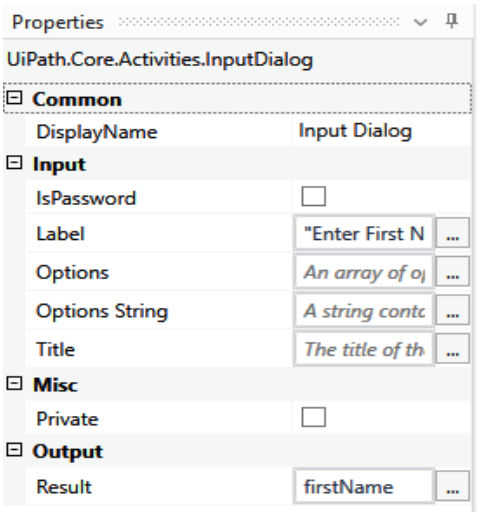
Write the Label as "Enter Last Name"

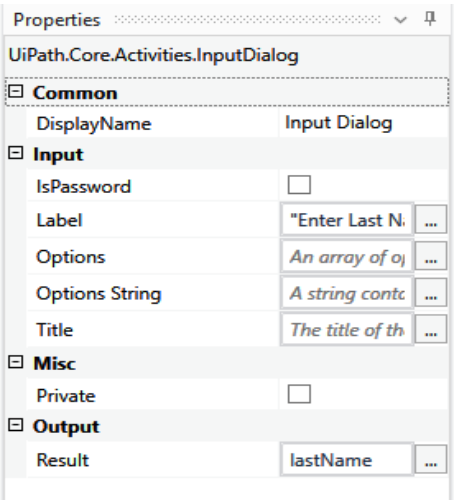
5. Drag and drop a **Message box** activity into the **Sequence**.



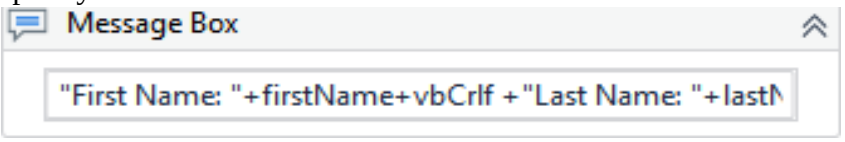
6. Next, create two **variables** and give them the desired names. These variables will receive the text that the user has entered in the **Input dialog** boxes.

7. We now have to specify the **Result** property (in the **Properties** panel) of the **Input dialog** box. On specifying the variable name there, it will receive the text that the user entered.

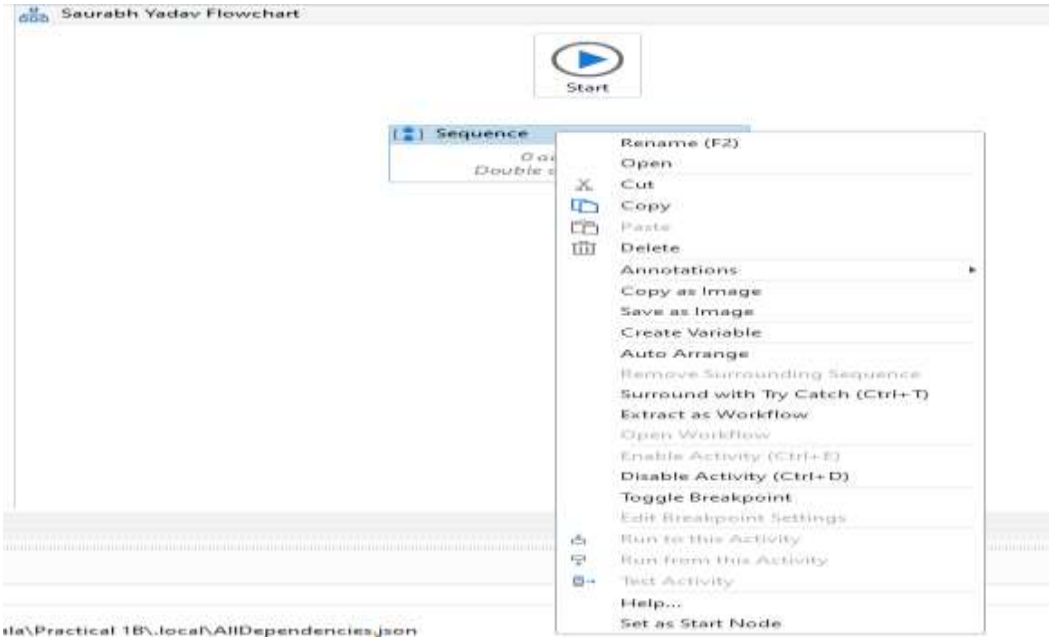


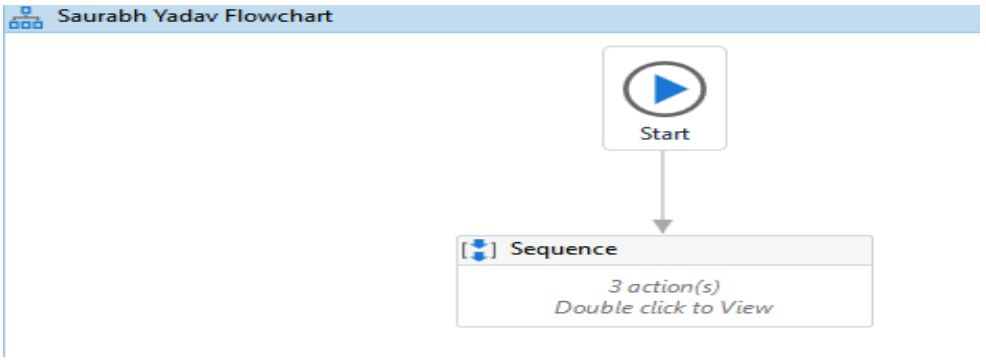


8. Specify the variables that we have created with other text in the Text area of the **Message box**.

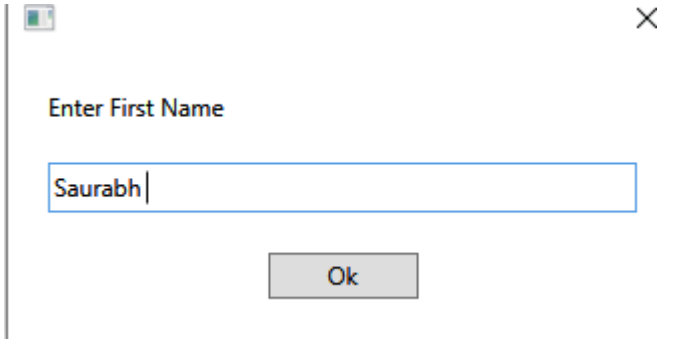


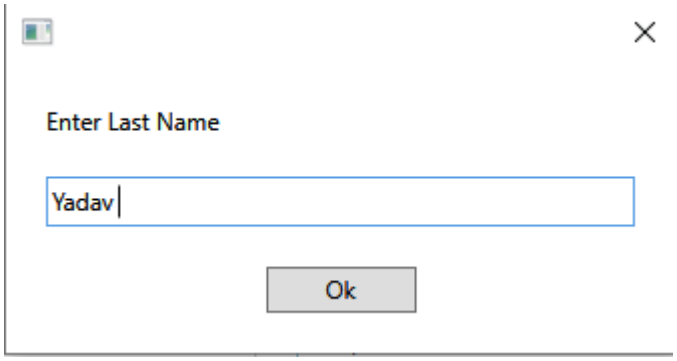
9. We need to connect the **Sequence** to the **Start** icon. This can be done by right-clicking on the **Sequence** activity and choosing the **Set as Start node** option.

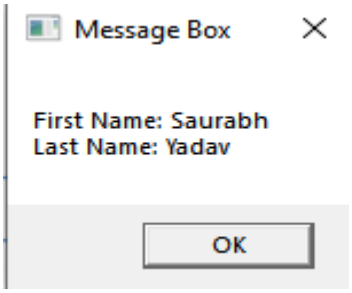




10. Hit the **Debug** button and see the result.


Enter First Name
Saurabh
Ok


Enter Last Name
Yadav
Ok

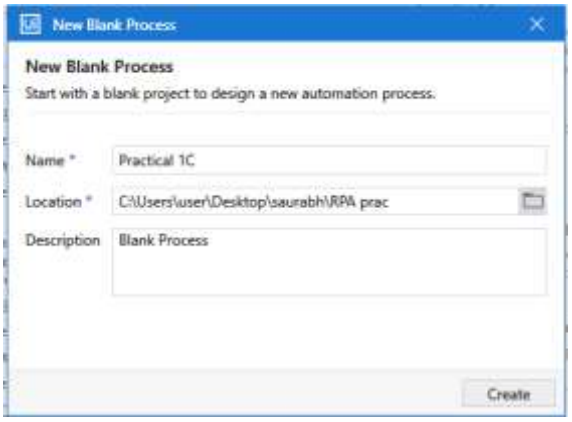

Message Box
First Name: Saurabh
Last Name: Yadav
OK

1 C) Create an UiPath Robot which can empty recycle bin in Gmail solely on basis of recording.

Aim: Use Web Recorder to empty trash in gmail.

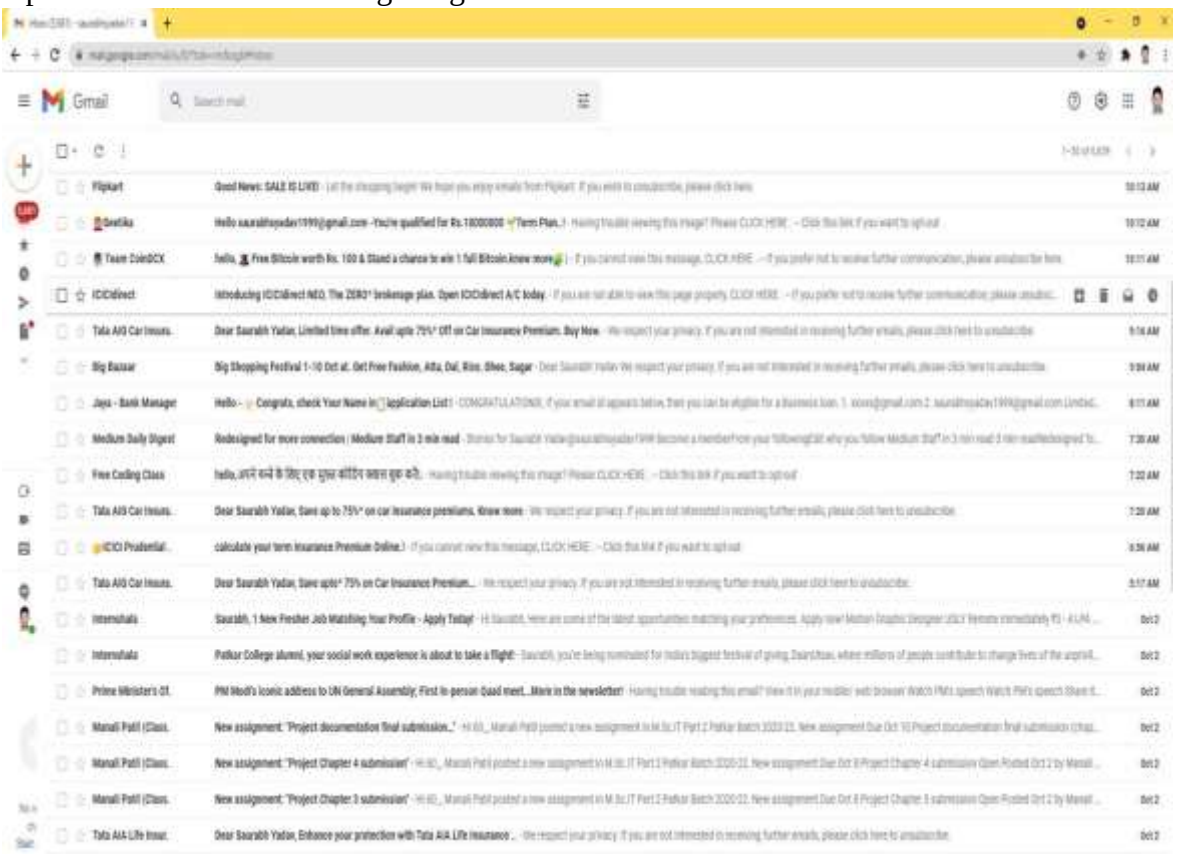
Steps:

1. Open UiPath Studio and click on **Blank** to start a fresh project. Give it a meaningful name.

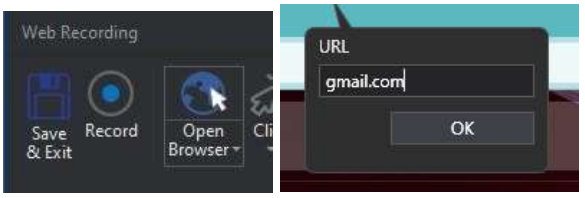


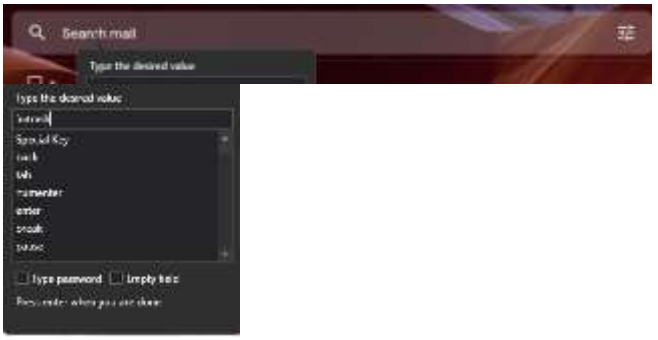
2. Choose Web recorder from the Recording drop-down list:

3. Open a Chrome browser and go to gmail.com.

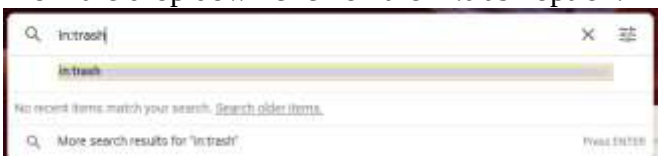


4. On the web recording panel click on “Open Browser” option and select the chrome window. In the url prompt type gmail.com.

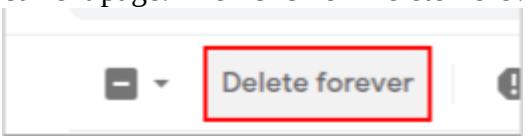




5. From the drop down click on the in:trash option.

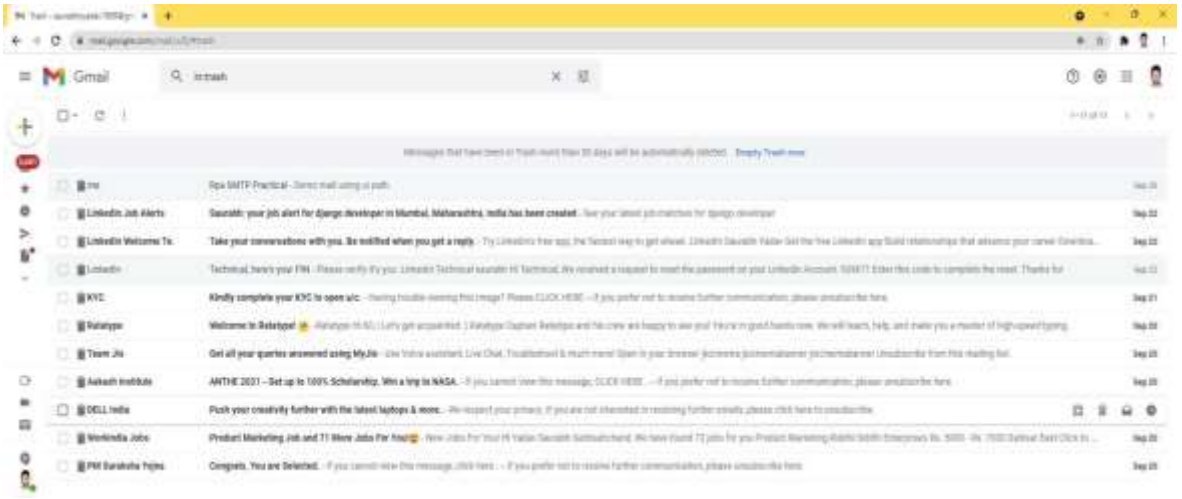


6. Once in the trash section of Gmail. Click on the checkbox button to select all the mail of the current page. Then click on Delete Forever button that appears.



7. Press Esc key and click on Save and Exit option in the web recording panel.

8. Now Run the program by clicking on Debug and see the output.



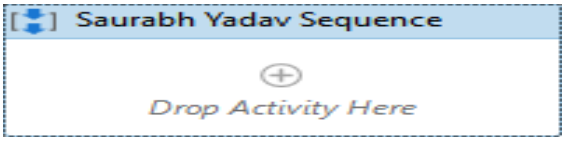
Practical no 2

2 A) Automate UiPath Number Calculation (Addition, Subtraction, Multiplication, Division of numbers).

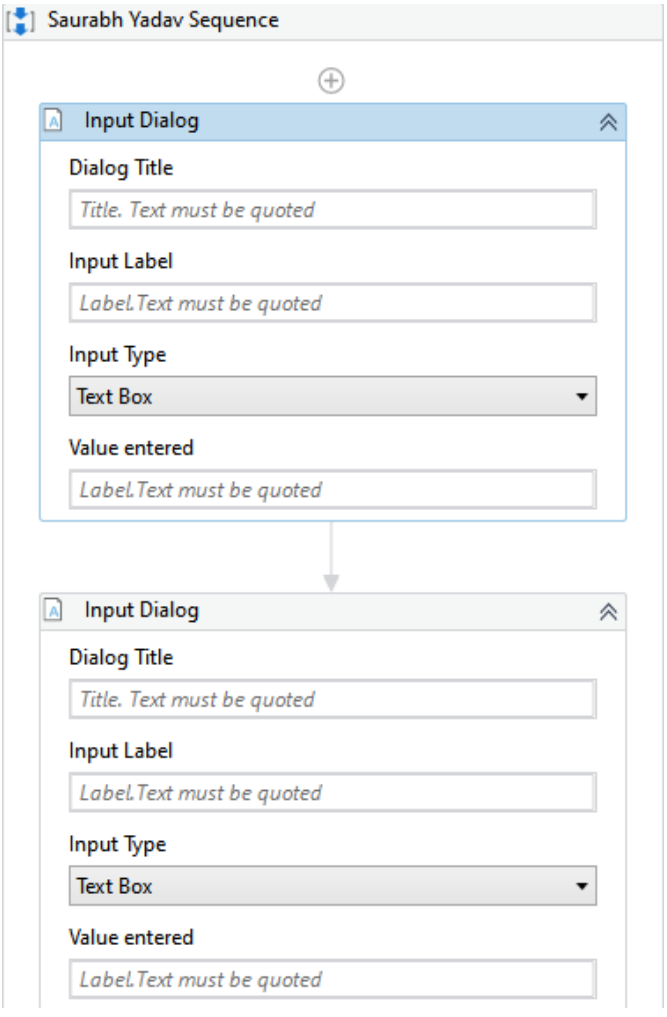
Aim: Use two Input dialogs for Numbers and do following calculations and show in Message Box.

Steps:

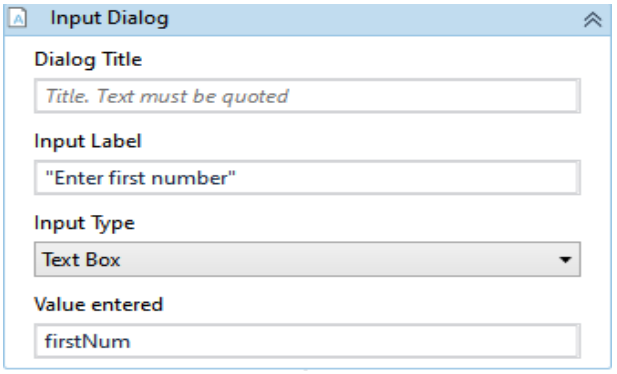
1. Create a new Blank Project and give it an appropriate name. Drag a Sequence activity from Activity tab.



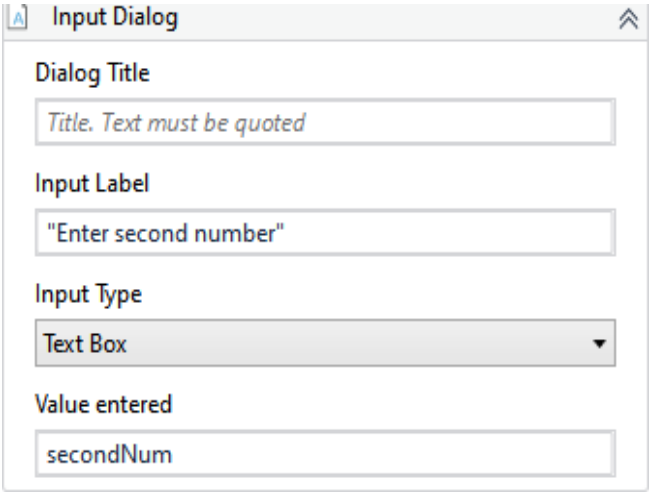
2. Search Input Dialog activity and drag and drop the activity twice.



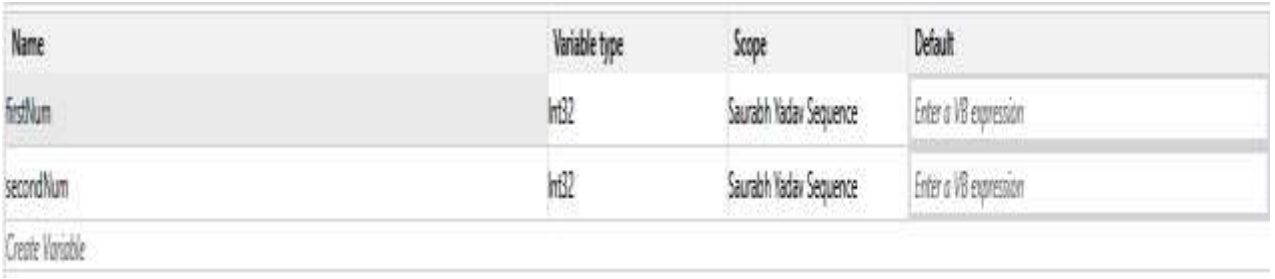
Give the first input dialog label as “Enter first number”.

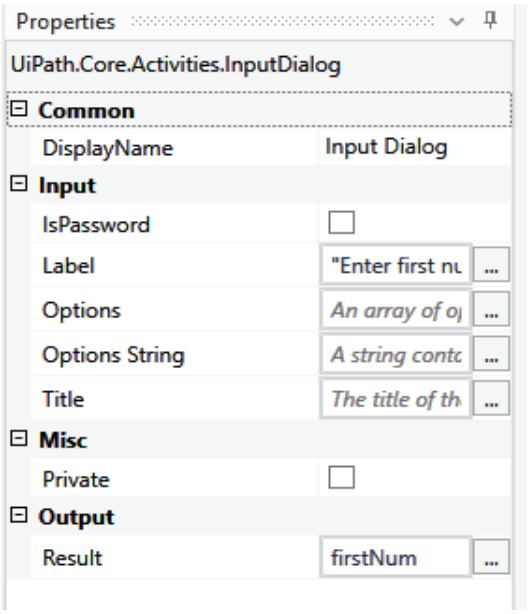


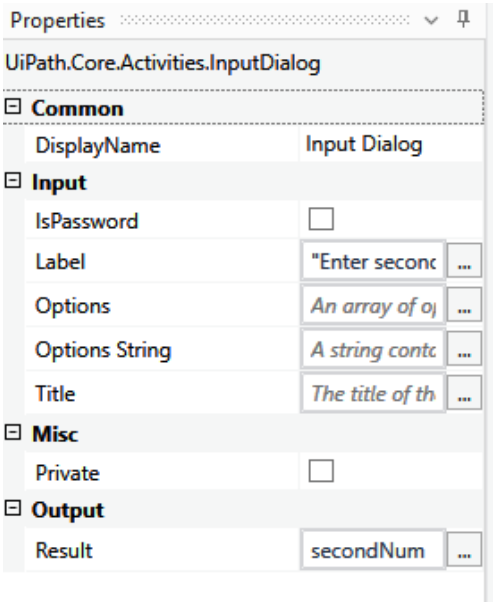
Give the second dialog label as “Enter second number”.



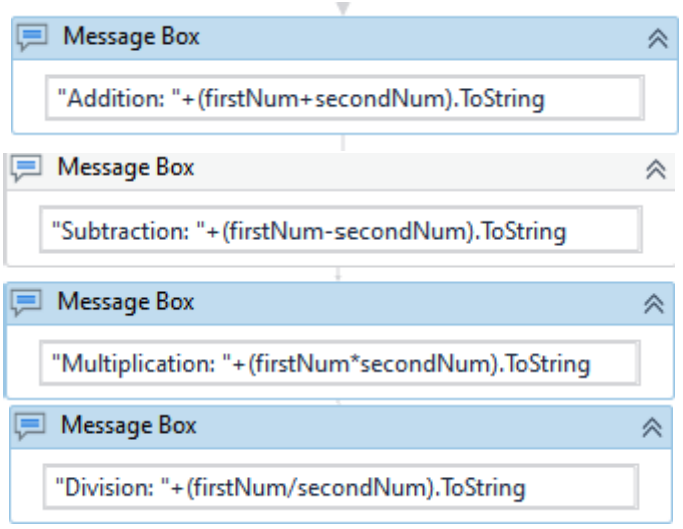
3. Create the variables to store both inputs, then enter the variables in the input dialog Result attributes.



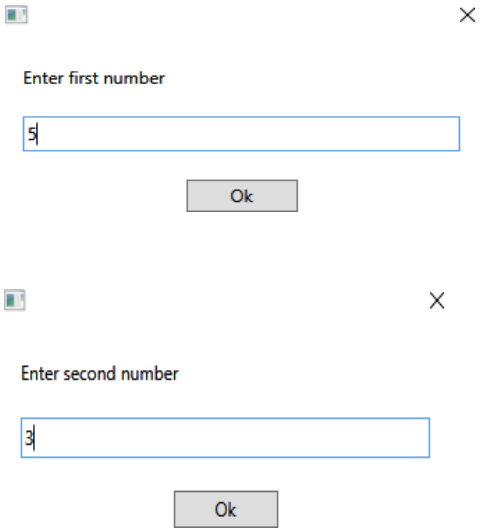


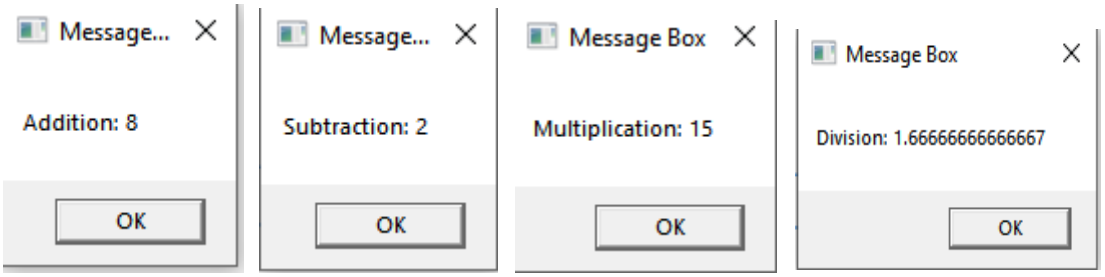


4. Drag and drop message box four times and perform the addition, subtraction, multiplication and division operation in each of the message boxes.



5. Hit the **Debug** button and see the result



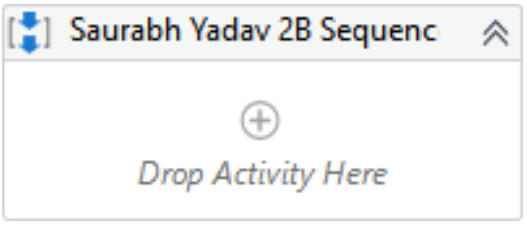


2 B) Create an automation UiPath project using different types of variables (number, datetime, Boolean, generic, array, data table)

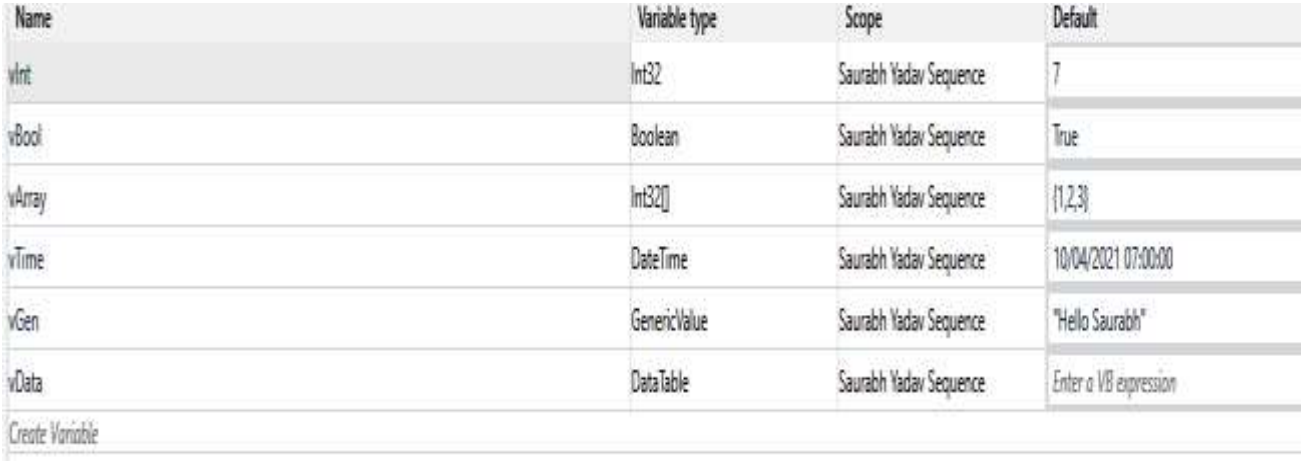
Aim: Create Different type of variable and provide default value and show in Message Box.

Steps:

1. Create a new Blank Project and give it an appropriate name. Drag a Sequence activity from Activity tab.



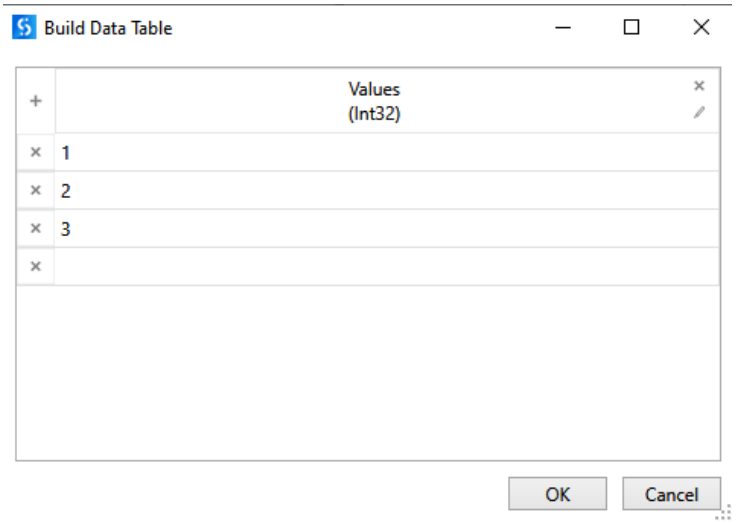
2. Create number(Int32), datetime, Boolean, string, generic, array variables and assign them default value.



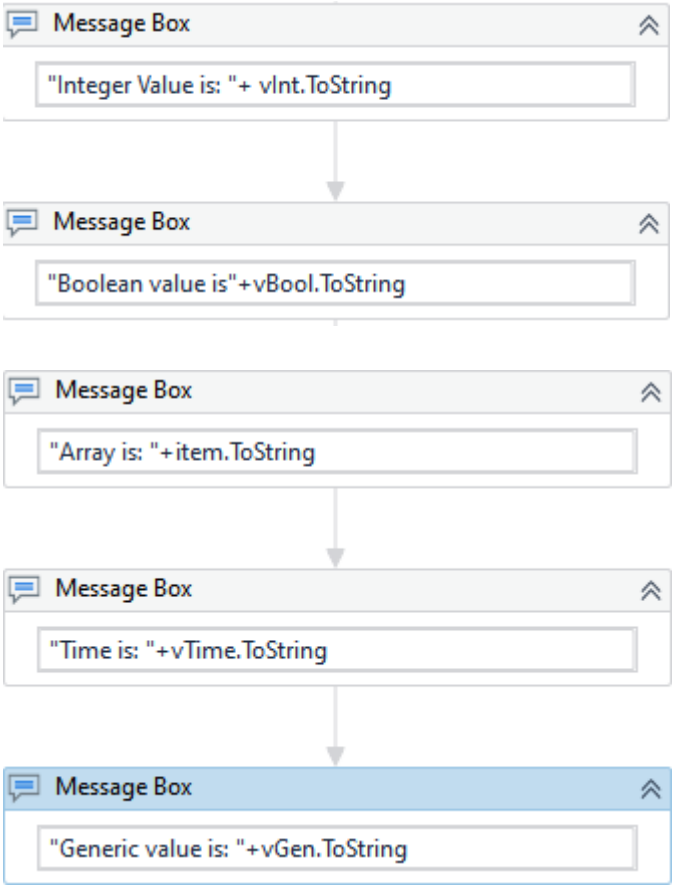
Name	Variable type	Scope	Default
vInt	Int32	Saurabh Yadav Sequence	7
vBool	Boolean	Saurabh Yadav Sequence	True
vArray	Int32[]	Saurabh Yadav Sequence	{1,2,3}
vTime	DateTime	Saurabh Yadav Sequence	10/04/2021 07:00:00
vGen	GenericValue	Saurabh Yadav Sequence	"Hello Saurabh"
vData	DataTable	Saurabh Yadav Sequence	Enter a VB expression
Create Variable			

3. Create a datable variable and add a build dataTable activity.

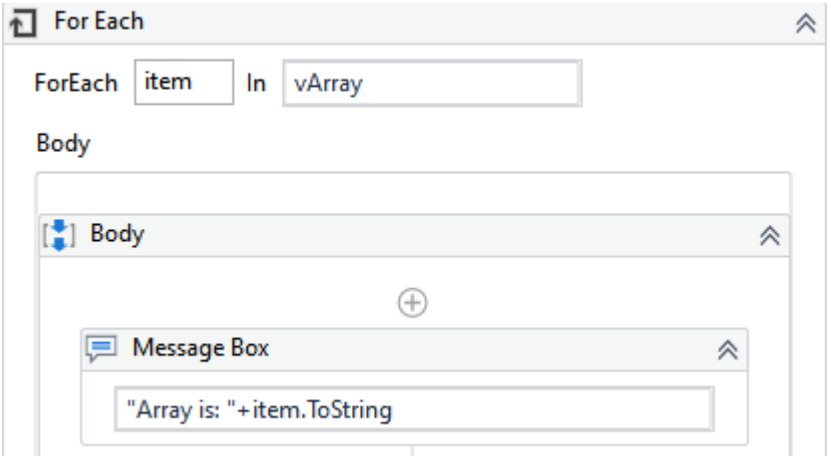
4. Create the values of data table.



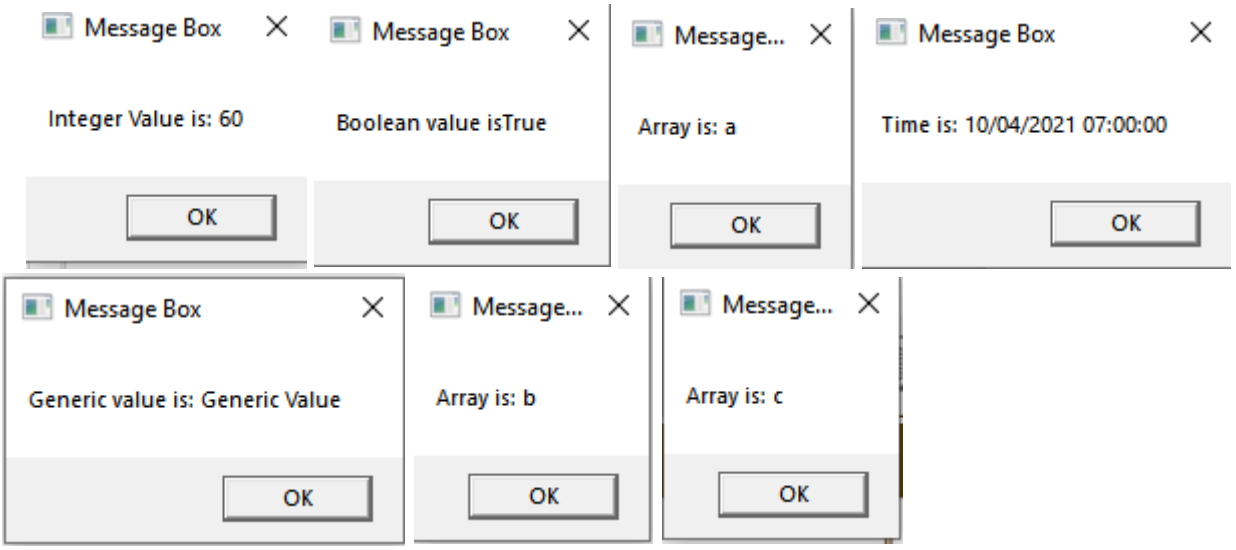
5. Add message box to show the number, datetime, Boolean, string, and generic variable.



6. Add for each activity and message box to show values of array variable.


For Each
ForEach
item
In
vArray
Body
Body
Message Box
"Array is: "+item.ToString

Output:


Message Box
Integer Value is: 60
OK
Message Box
Boolean value isTrue
OK
Message...
Array is: a
OK
Message Box
Time is: 10/04/2021 07:00:00
OK
Message Box
Generic value is: Generic Value
OK
Message...
Array is: b
OK
Message...
Array is: c
OK

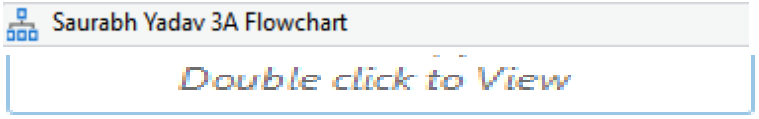
Practical no 3

3 A) Create an automation UiPath Project using decision statements.

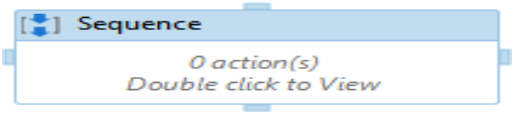
Aim: Use one input dialog for Number, create a process to find provided Number is odd or even using condition (Flow Chart).

Steps:

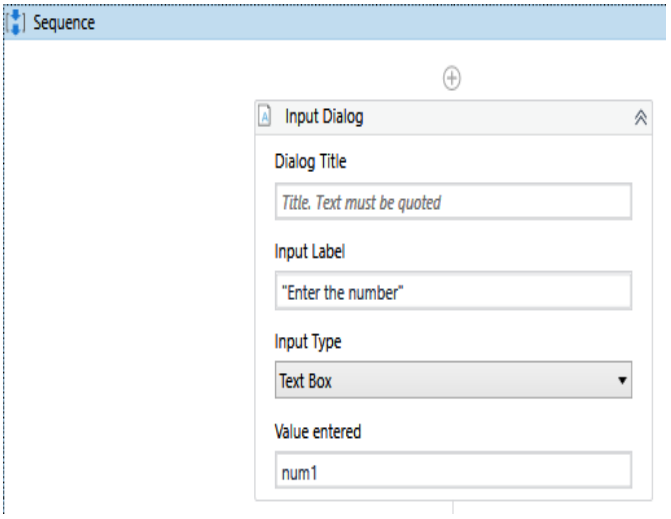
6. Create a new Blank Project and give it an appropriate name. Drag a Flowchart activity from Activity tab.



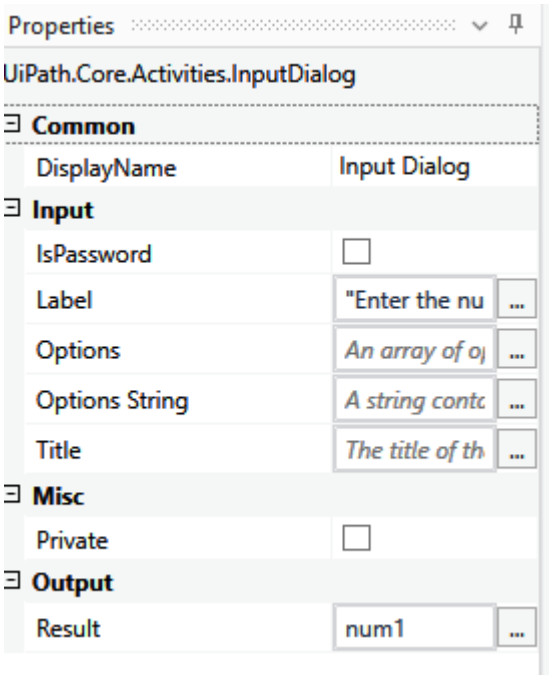
7. Double click on flowchart and drag a sequence activity.



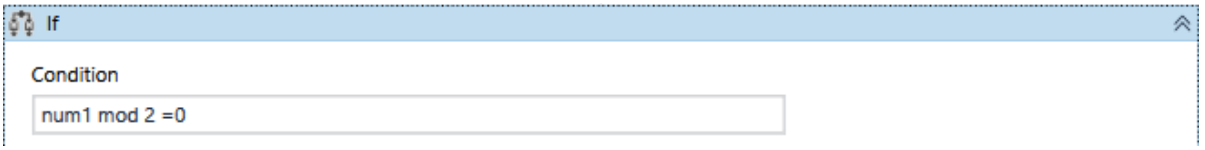
8. Drag an Input Dialog activity and give label “Enter the number”.



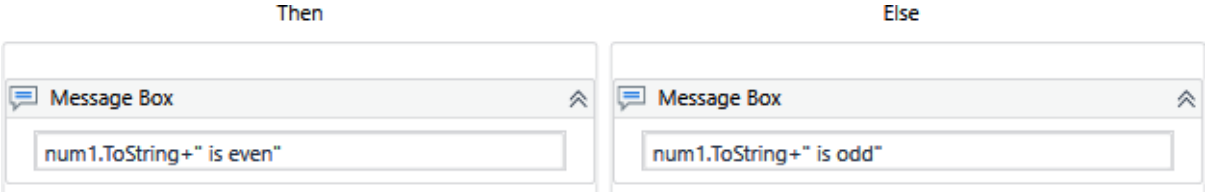
9. Create a variable of Int32 type and enter the variable in result attribute of Input Dialog.



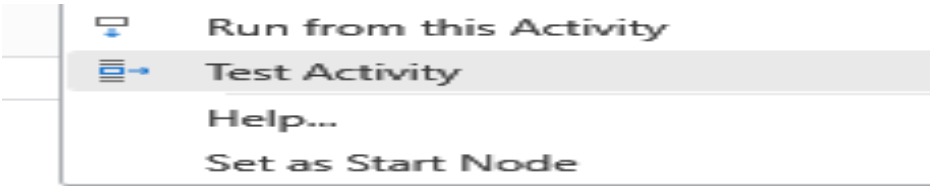
10. Write the following expression in Condition: num mod 2 = 0.



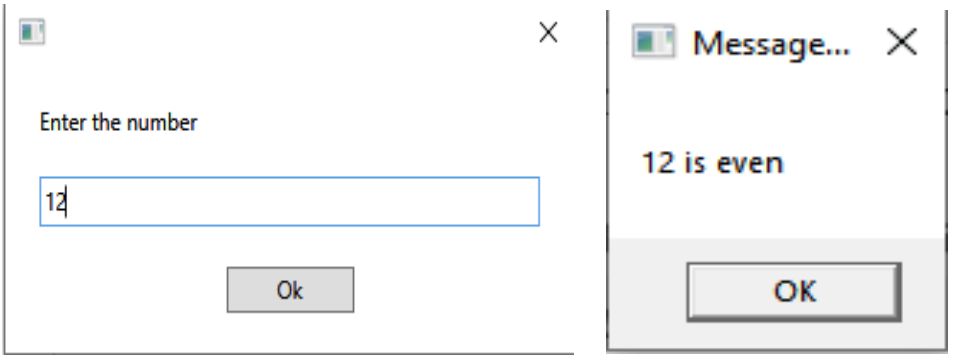
11. In the Message Box in then part write “Number is even” and in the Message Box in else part write “Number is odd”.



12. Connect the sequence to the start node by right clicking on sequence and selecting the “Set as start node” option.



Output:

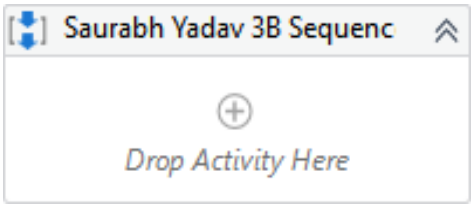


3 B) Create an automation UiPath Project using looping statements.

Aim: Create a dummy List of fruits. Loop each and print in Message Box.

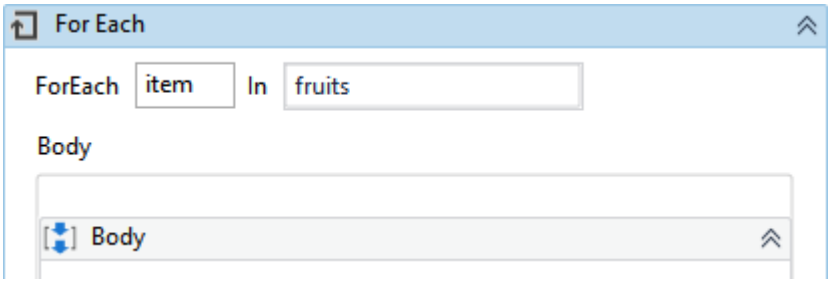
Steps:

1. Create a new Blank Project and give it an appropriate name. Drag a Sequence activity from Activity tab.

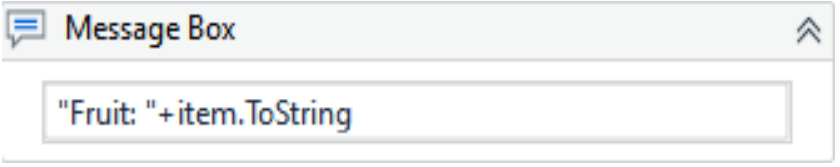


2. Create an Array of String Type named fruits. Give default value as {“Apple”, “Mango”, “Banana”, “Orange”, “Watermelon”}.

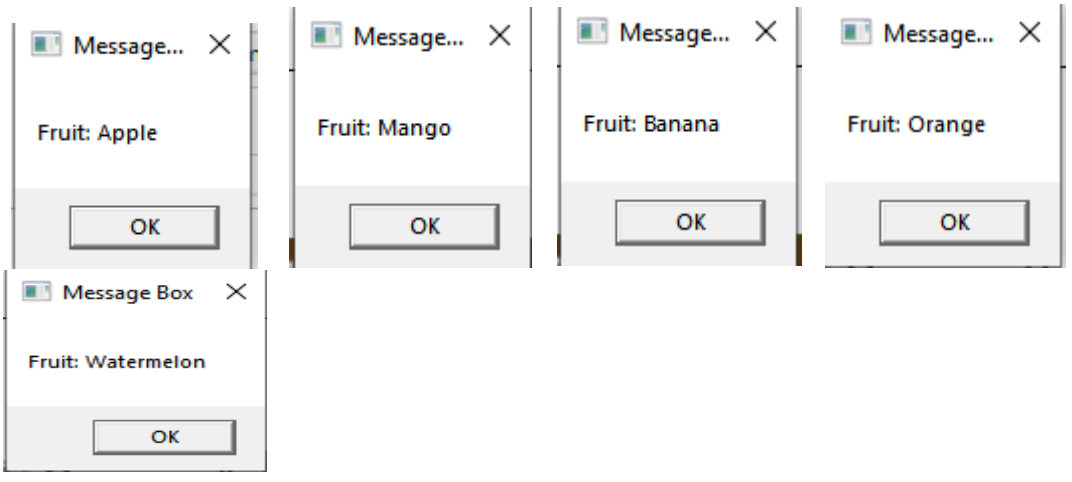
3. Add a for each activity and enter the fruits array as the iterating variable.



4. Add a Message Box and print item.



Output:



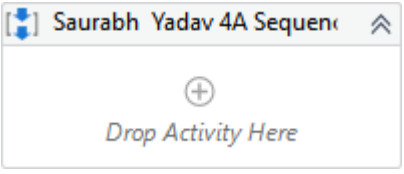
Practical : 4

4 A) Automate any process using basic recording.

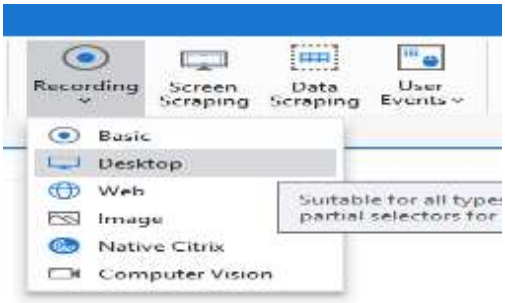
Aim: Automate Existing Notepad using basic Recorder

Steps:

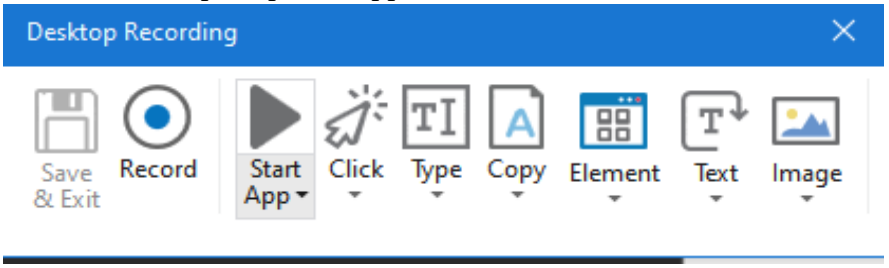
1. Create a new Blank Project and give it an appropriate name. Drag a Sequence activity from Activity tab.



2. Select Desktop Recording under Recording.



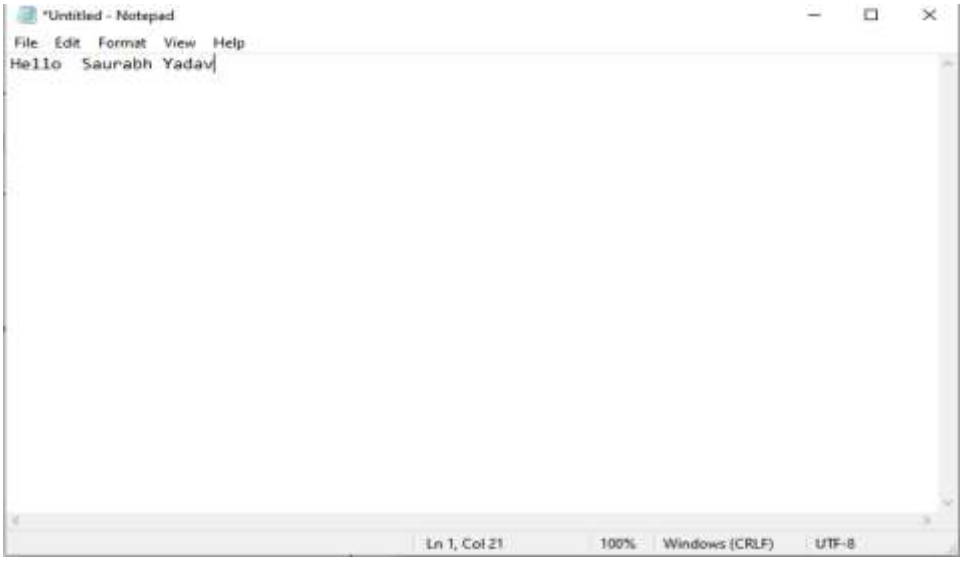
3. Open a new notepad, then on desktop recording click on open application and select notepad the click ok on the prompt that appears.



4. Start recording by clicking on record button in recording panel, then click in the text area of notepad.

5. In the type into prompt enter the text you want to be typed.

Output:



4 . B) Automate any process using desktop recording.

Aim: Use "double Ui" and automate stuff (Scrape Text, Input Text, Click Button)

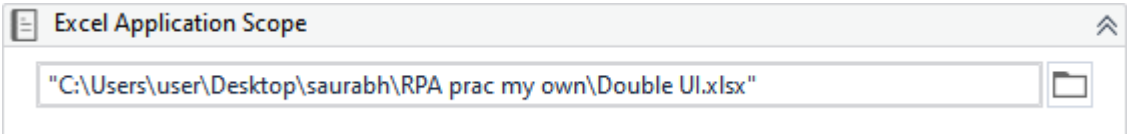
Insert the data from the following excel file.

A	B	C	D	E	F	G
Sr no.	Cash In	Check_1	Check_2	Total	Transactic	Status
1	100	200	400			
2	200	300	600			
3	300	400	800			
4	400	500	1000			
5	500	600	1200			
6	600	700	1400			
7	700	800	1600			
8	800	900	1800			

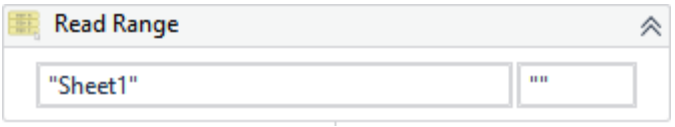
Steps:

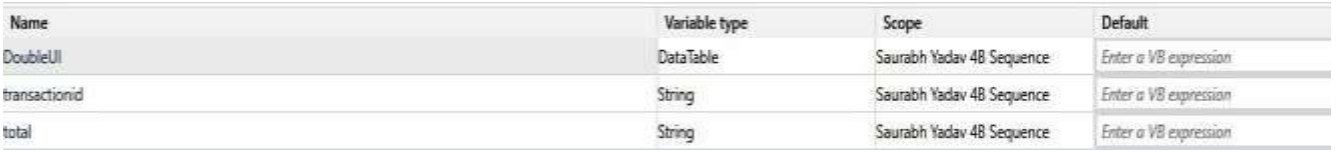
1. Create a new Blank Project and give it an appropriate name. Drag a Sequence activity from Activity tab.

2. Add Excel Application Scope Activity and enter the path of the excel file.



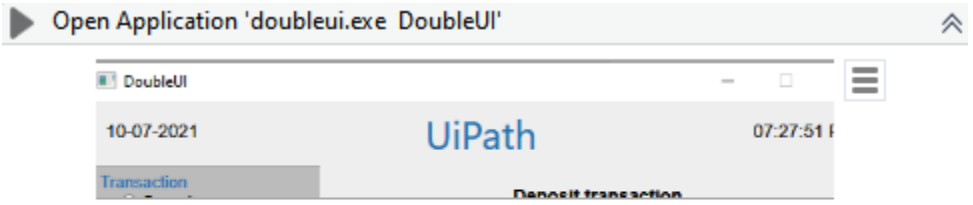
3. Add Read Range and create a DataTable to store the data that will be read by read range.

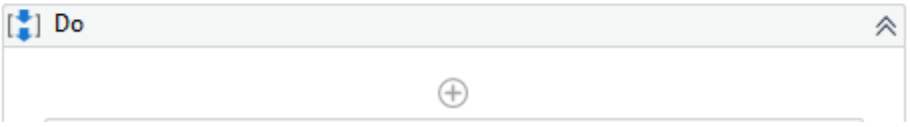




Name	Variable type	Scope	Default
DoubleUI	DataTable	Saurabh Yadav 4B Sequence	Enter a VB expression
transactionid	String	Saurabh Yadav 4B Sequence	Enter a VB expression
total	String	Saurabh Yadav 4B Sequence	Enter a VB expression

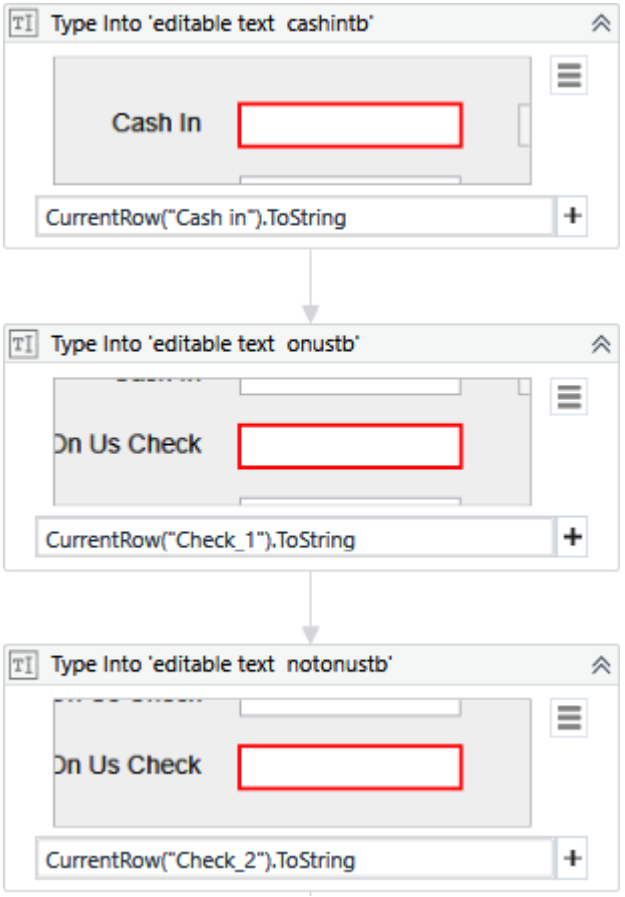
4. Add Open Application activity, start Double UI then click on “Indicate window on screen” option and select Double UI.



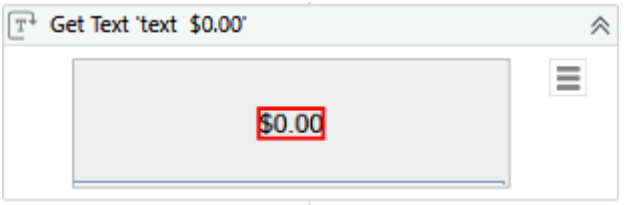


5. Add For each row in DataTable activity and iterate over DoubleUI datatable.

6. Add three type into activity in the for each activity. Indicate to the three textboxes in the Double UI app. Enter the DataTable element you wan to enter in the Text attribute.

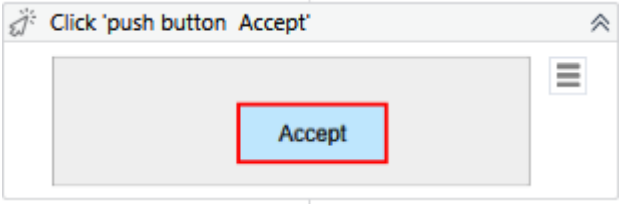


7. Add two get text activities and create two variables to store the transaction is andtotal. Indicate these elements to the Get text activities.

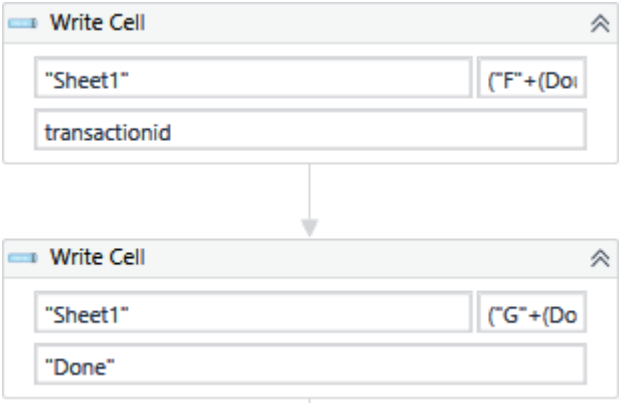


In case of Total remove the name attribute from the Edit Selector.

8. Add Click activity and indicate to Accept Button in Double UI.

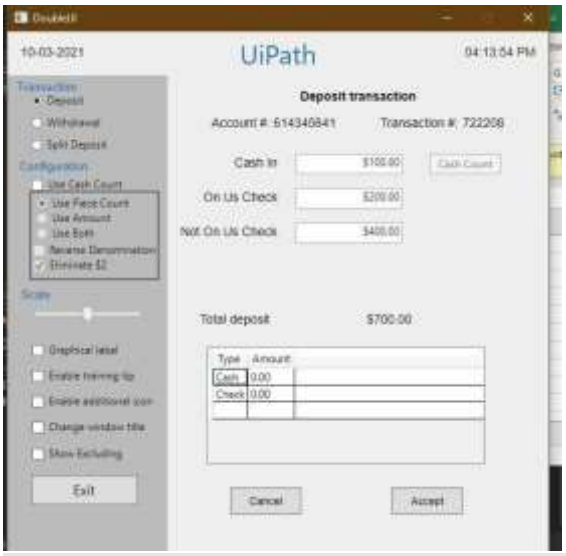


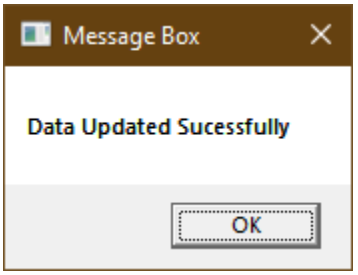
9. Add three Write Cell Activities to enter the transaction id and total and to Update the status column. ("E"+(DoubleUI.Rows.IndexOf(CurrentRow)+2).ToString)



10. Finally add a Message Box to print the success of the updation of Excel file.

Output:





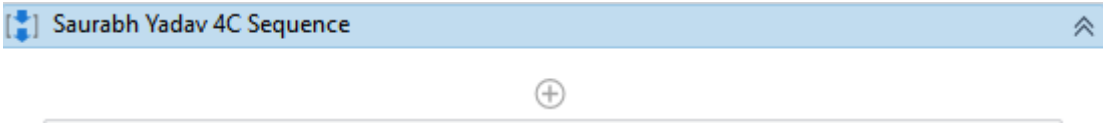
	A	B	C	D	E	F	G
1	Sr no.	Cash In	Check_1	Check_2	Total	Transactic	Status
2	1	100	200	400	700	722208	Done
3	2	200	300	600	1100	722209	Done
4	3	300	400	800	1500	722210	Done
5	4	400	500	1000	1900	722211	Done
6	5	500	600	1200	2300	722212	Done
7	6	600	700	1400	2700	722213	Done
8	7	700	800	1600	3100	722214	Done
9	8	800	900	1800	3500	722215	Done

4 C) Automate any process using web recording.

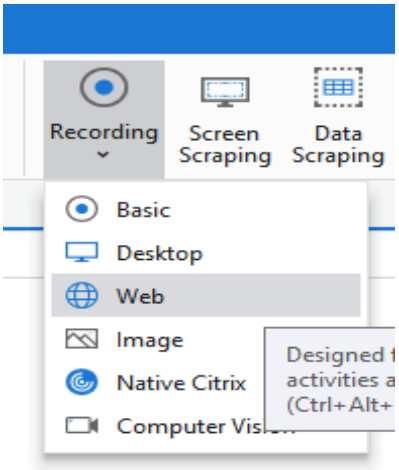
Aim: Use "google.com" and automate stuff (Scrape Text, Input Text, Click Button)

Steps:

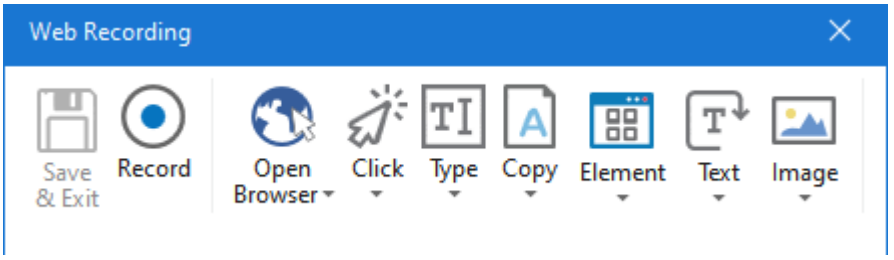
1. Create a new Blank Project and give it an appropriate name. Drag a Sequence activity from Activity tab.



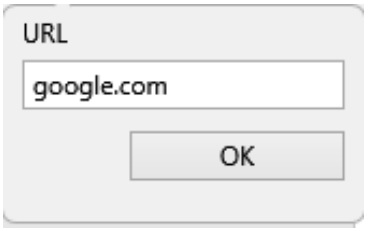
2. Select web recording under the recording option.



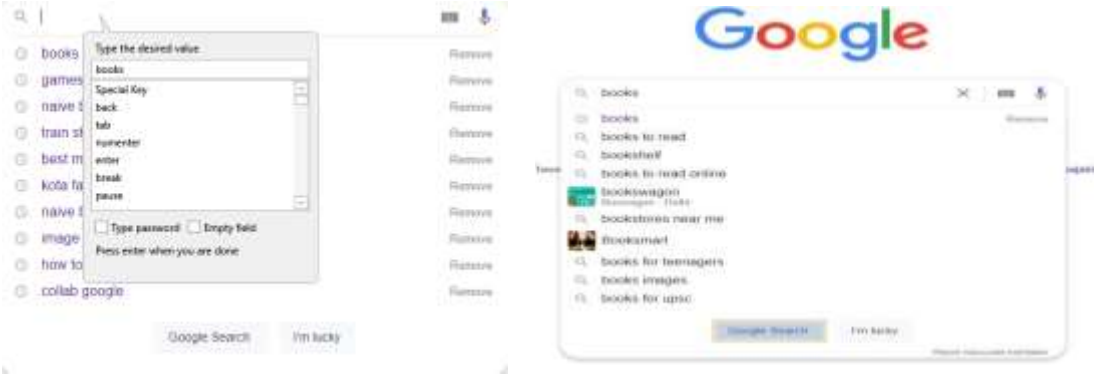
3. Open a Chrome window and from the web recording dialog select open browser and click on the Chrome window.



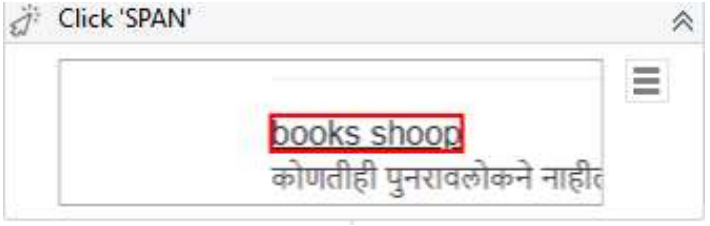
4. In the url dialog enter google.com and press enter.



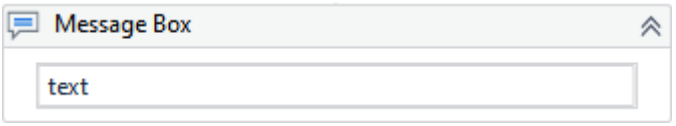
5. Now start recording by clicking on Record and click on the search button. Type the desired search and click on the Google search button. The press Esc.



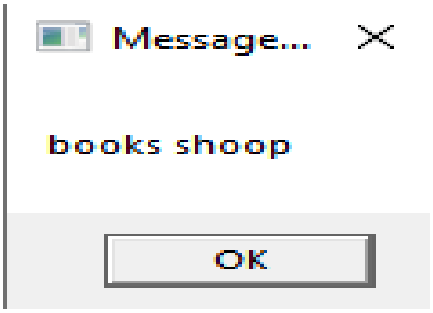
6. Now press save and exit. Add get text activity and select the text you want to scrape.



7. Add message box to print scrapped text.



Output:

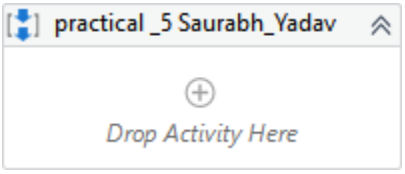


Practical no 5

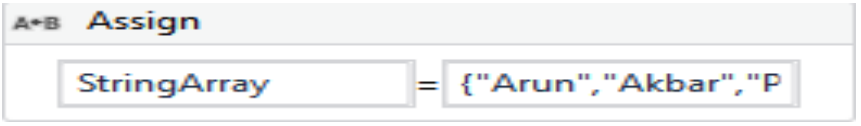
Aim: Consider an array of names. We have to find out how many of these start with the letter “a” Create an automation where the number of names starting with “a” is counted and the result is displayed.

Steps:

1. Create a sequence.

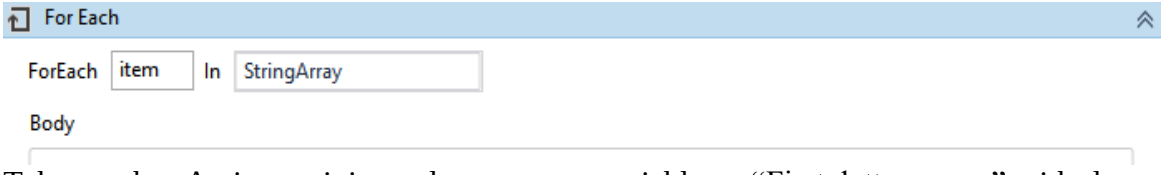


2. Use 1 Assign Activity
   Create array string Eg:
   {"Arun","Akbar","Pankaj","Bharat","Saurabh",”Meeth”,”Akshay”}

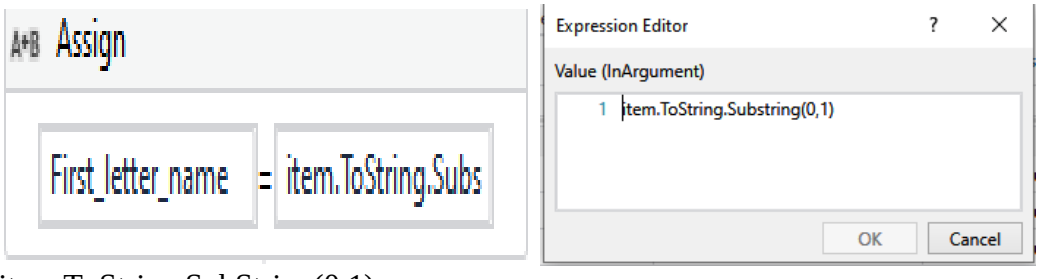


3. In Variable section take datatype as Array of[T] -> String

4. Take For each loop where each item in stringArray will occur repeatedly.

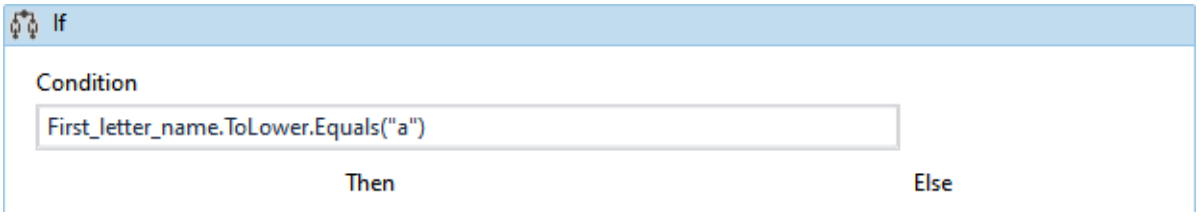


5. Take another Assign activity and create new variable as “First_letter_name” with datatype as String.

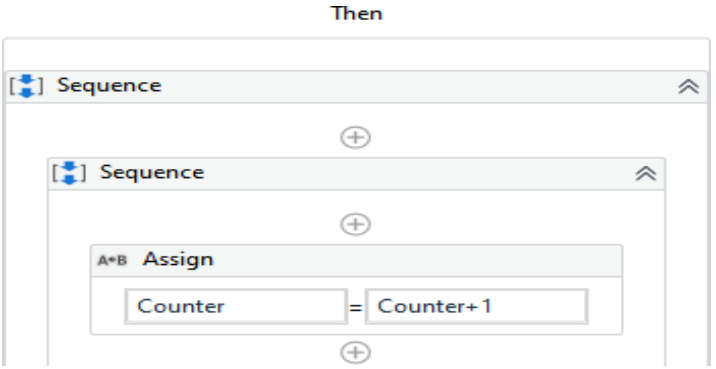


   item.ToString.SubString(0,1)
   Where 0 is the starting index value.
   1 is the length of the name.

6. Now Take If condition -First_letter_name.ToLower.Equals("a") this condition states that take the letter from the name list of array which starts with alphabet “A”or “a”.

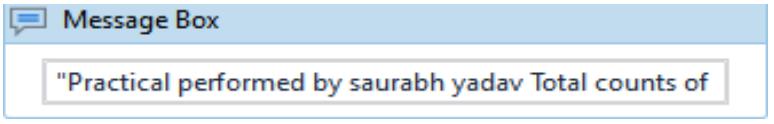


7. Then: take assign activity create Counter variable to count the number of names in the array starting with “A”, give default value of Counter variable as 0.

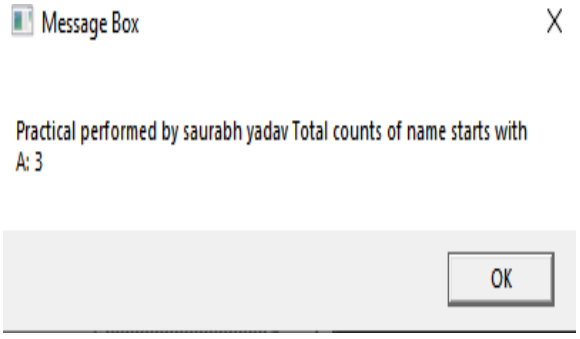


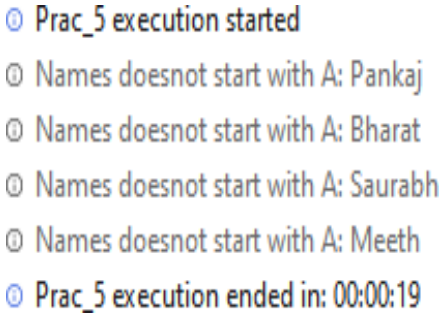
8. Else: take write line activity and put as " Names doesnot start with A:"+item.ToString

9. Add a message box after the if activity and show the count of Names starting with “A”.



10. Hit the Debug button and see the result.





Practical no 6

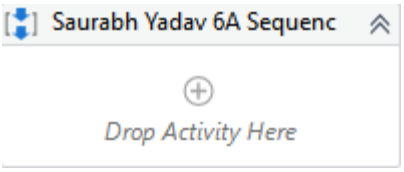
6A. Create an application automating the read, write and append operation on excel file.

Aim: Using Excel sheet data perform some calculation on the data writing it into the cells

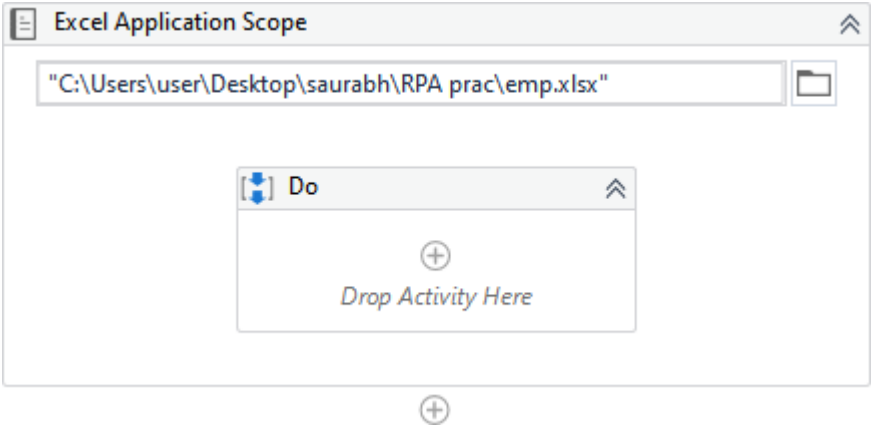
Steps: Create a sample excel as shown below

	Emp_Id	F_name	L_name	Salary	Expenses	Saving
2	1001	Saurabh	Yadav	100000	10000	
3	1002	Pankaj	Gavali	200000	20000	
4	1003	Bharat	Bhagat	300000	30000	
5	1004	Virat	Kohli	400000	40000	
6	1005	Mahendra	Dhoni	500000	50000	
7						

1. Start project as sequence

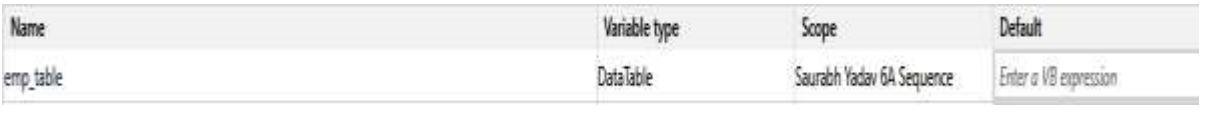


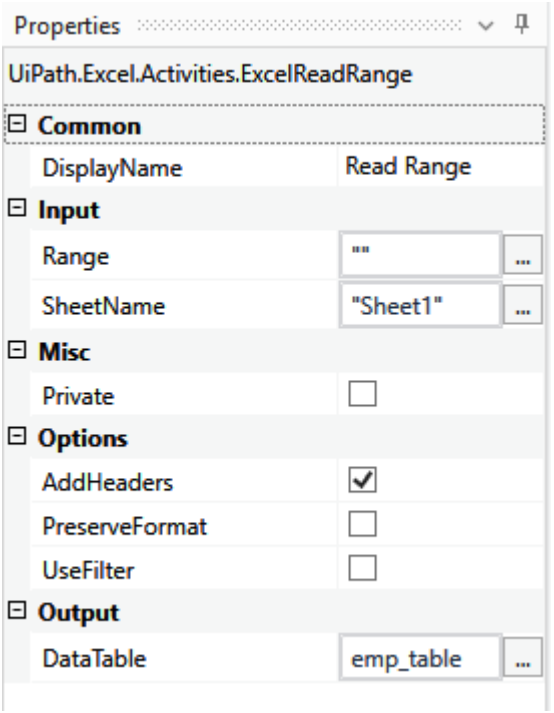
2. Drag Excel Application Scope Activity from Activity Panel and enter the excel file path.



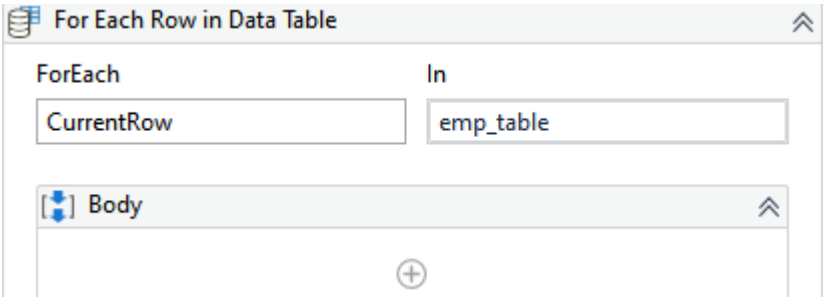
3. Add “Read Range” from activities and add the range of the cells if required.

4. Create a variable of type DataTable (here variable name: emp_table variable type: DataTable). Enter the variable in the Read Range Output>DatableTable attribute.

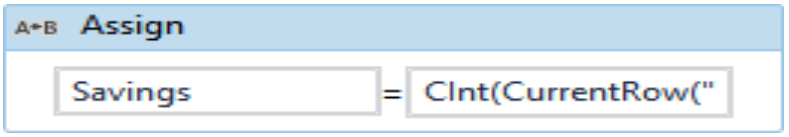




5. Select “For Each Row” and enter Input DataTable (here emp_table).

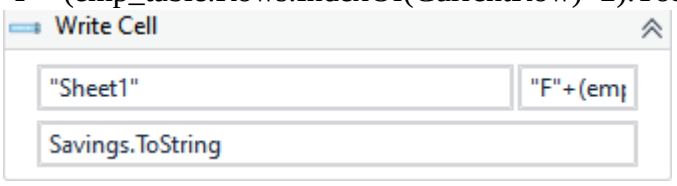


6. Inside Body insert “Assign” variable (Saving) with int32 as variable type .
7. Enter the following in value part of assign CInt(CurrentRow("Salary"))-CInt(CurrentRow("Expenses")).

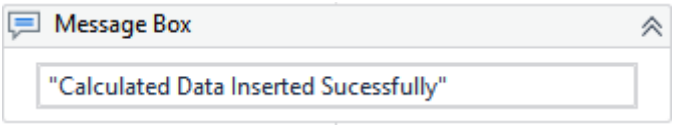


8. A “Message Box” to print the result converting it to string.

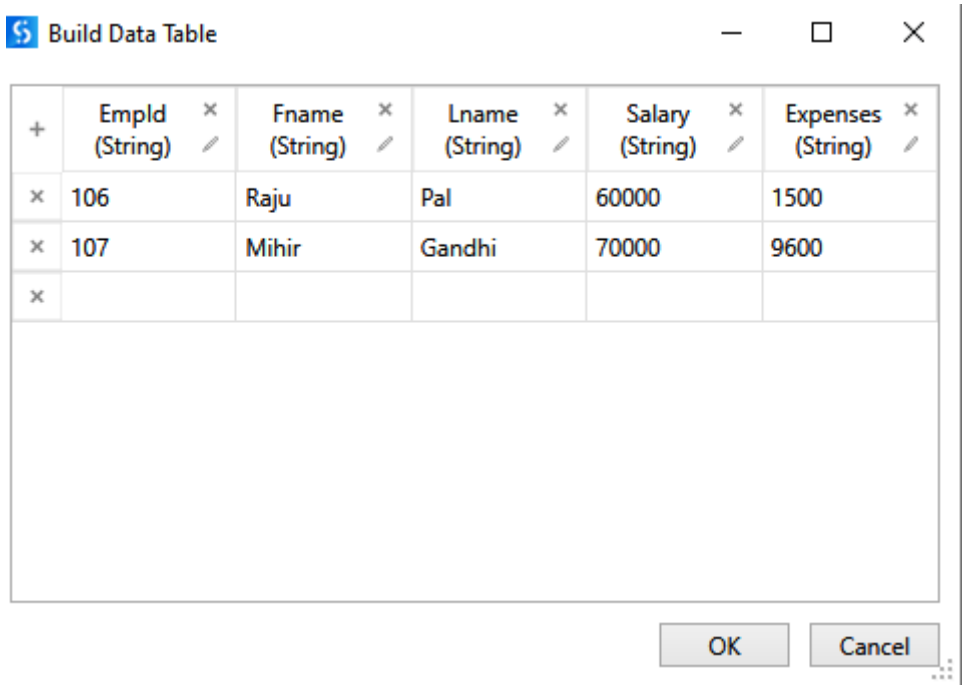
9. To insert the Saving value in the excel use Write Cell Activity. Give the Savings Variable in Value. In Range type: "F"+(emp_table.Rows.IndexOf(CurrentRow)+2).ToString.



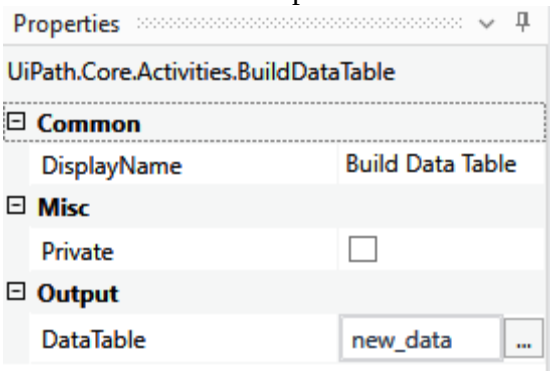
10. Add a Message Box to indicate that the calculated values have been successfully inserted.



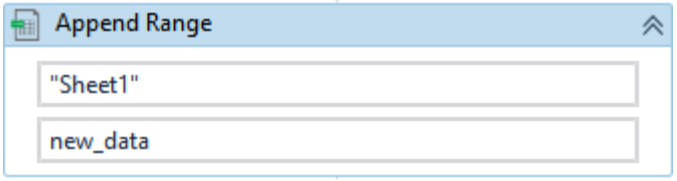
11. In order to append new data create data table using “Build Data Table” and insert the values.



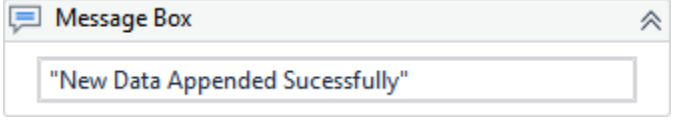
12. Create a new variable with name: new_data and variable type: DataTable and pass it to the DataTable field in Output attribute of Build datatable activity.



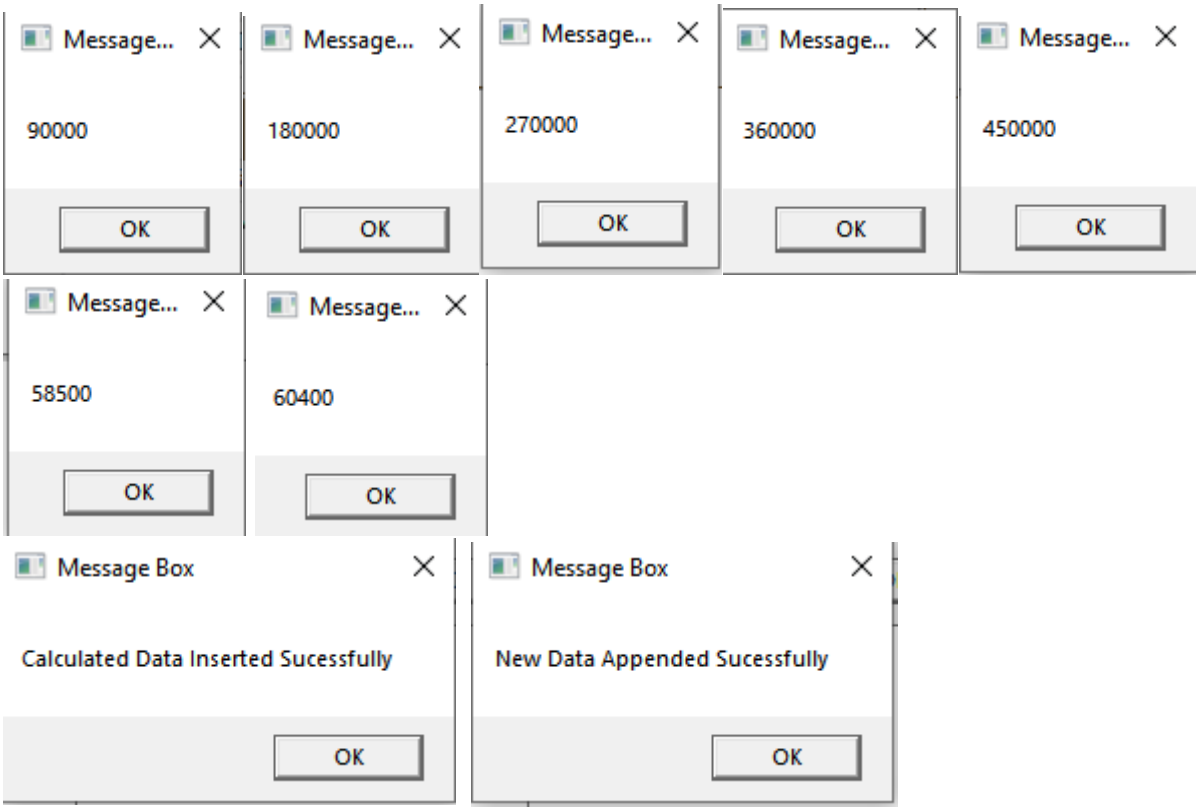
13. Select “Append Range” and insert DataTable name (“new_data”).



14. Finally, a “Message Box” to print a confirmation message.



Output:



Final Output:

Emp_Id	F_name	L_name	Salary	Expenses	Saving
1001	Saurabh	Yadav	100000	10000	90000
1002	Pankaj	Gavali	200000	20000	180000
1003	Bharat	Bhagat	300000	30000	270000
1004	Virat	Kohli	400000	40000	360000
1005	Mahendra	Dhoni	500000	50000	450000
106	Raju	Pal	60000	1500	58500
107	Mihir	Gandhi	70000	9600	60400

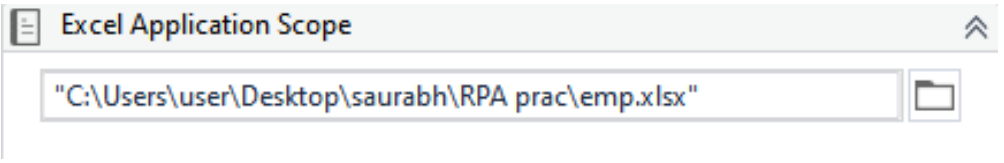
6B. Create automate the process to extract data from an excel file into a data table and vice versa.
Aim: Using one excel sheet data to copy and writing it into other excel sheet.
(Using excel data given below)

Emp_Id	F_name	L_name	Salary	Expenses	Saving
1001	Saurabh	Yadav	100000	10000	90000
1002	Pankaj	Gavali	200000	20000	180000
1003	Bharat	Bhagat	300000	30000	270000
1004	Virat	Kohli	400000	40000	360000
1005	Mahendra	Dhoni	500000	50000	450000
106	Raju	Pal	60000	1500	58500
107	Mihir	Gandhi	70000	9600	60400

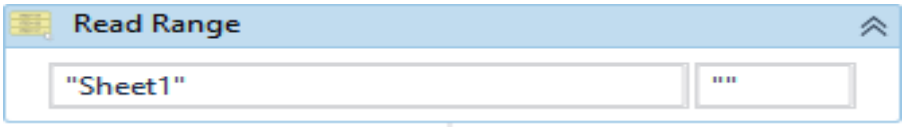
Step:

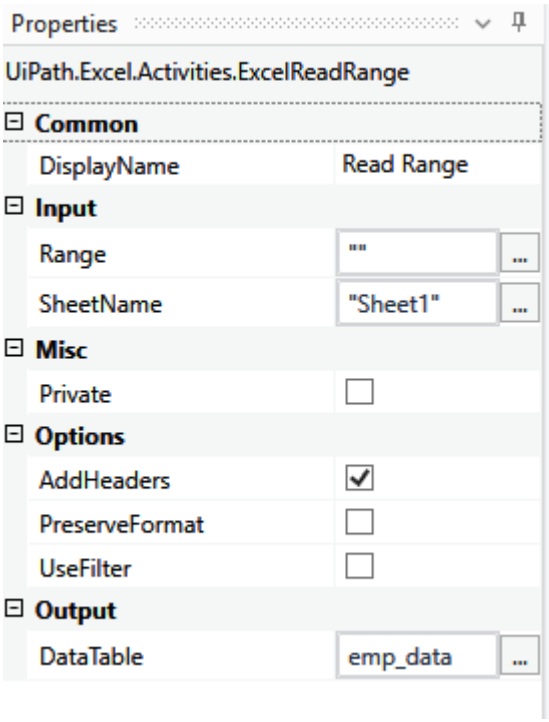
1. Start project with sequence.

2. Add “Excel Application Scope” and enter the path of excel file.

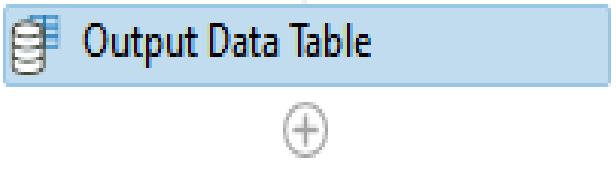


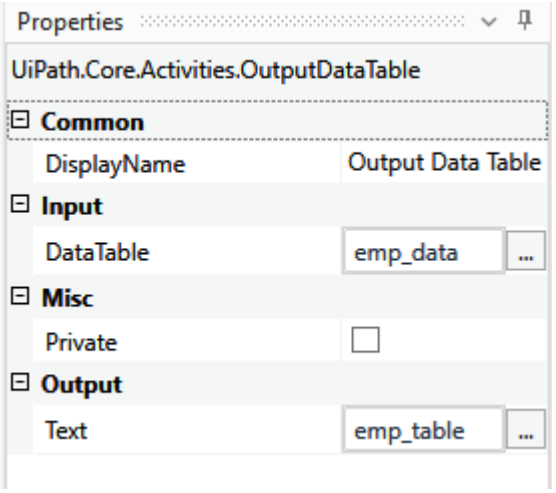
3. Inside body add “Read Range” variable name and datatype as datatable(emp_data) and range to extract data from excel (blank means entire sheet).



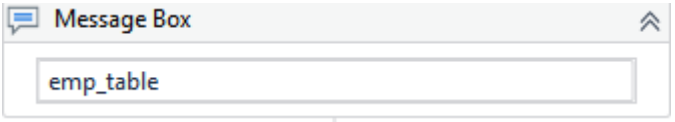


4. Add “Output Data Table” input as “emp_data” output as “emp_table”.

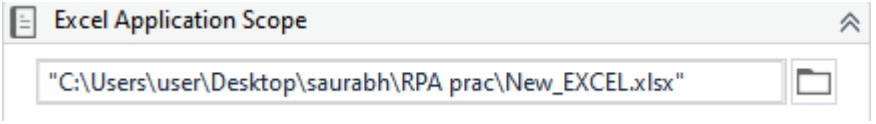




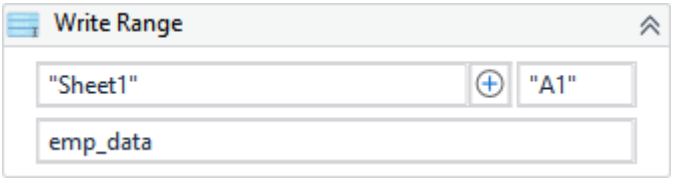
5. “Message Box” to print the result adding variable “emp_table” (here extracted data will be printed).



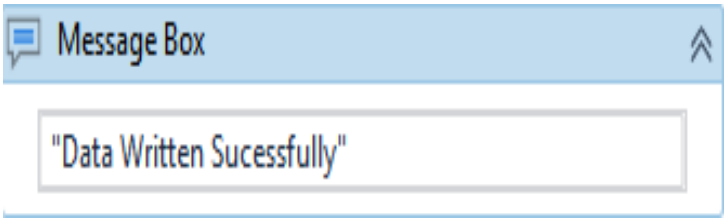
6. Insert “excel application scope” and new excel sheet path.



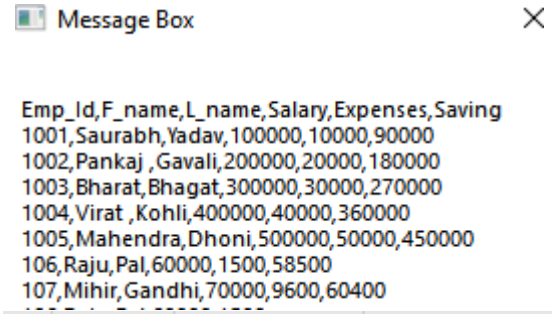
7. Insert “Write Range” range from where the data should be pasted and variable “emp_data”.

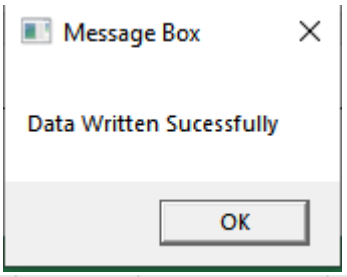


8. Message Box to print confirmation



Output:





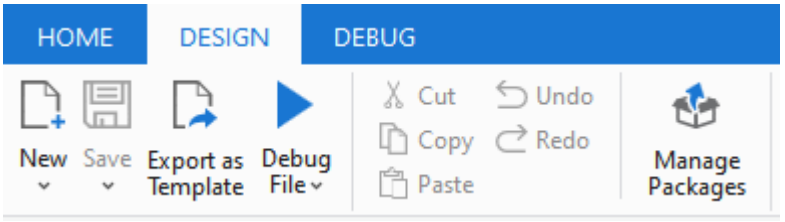
	A	B	C	D	E	F	G
1	1001	Saurabh	Yadav	100000	10000	90000	
2	1002	Pankaj	Gavali	200000	20000	180000	
3	1003	Bharat	Bhagat	300000	30000	270000	
4	1004	Virat	Kohli	400000	40000	360000	
5	1005	Mahendra	Dhoni	500000	50000	450000	
6	106	Raju	Pal	60000	1500	58500	
7	107	Mihir	Gandhi	70000	9600	60400	
8	106	Raju	Pal	60000	1500		
9	107	Mihir	Gandhi	70000	9600		
10							

Practical no 7

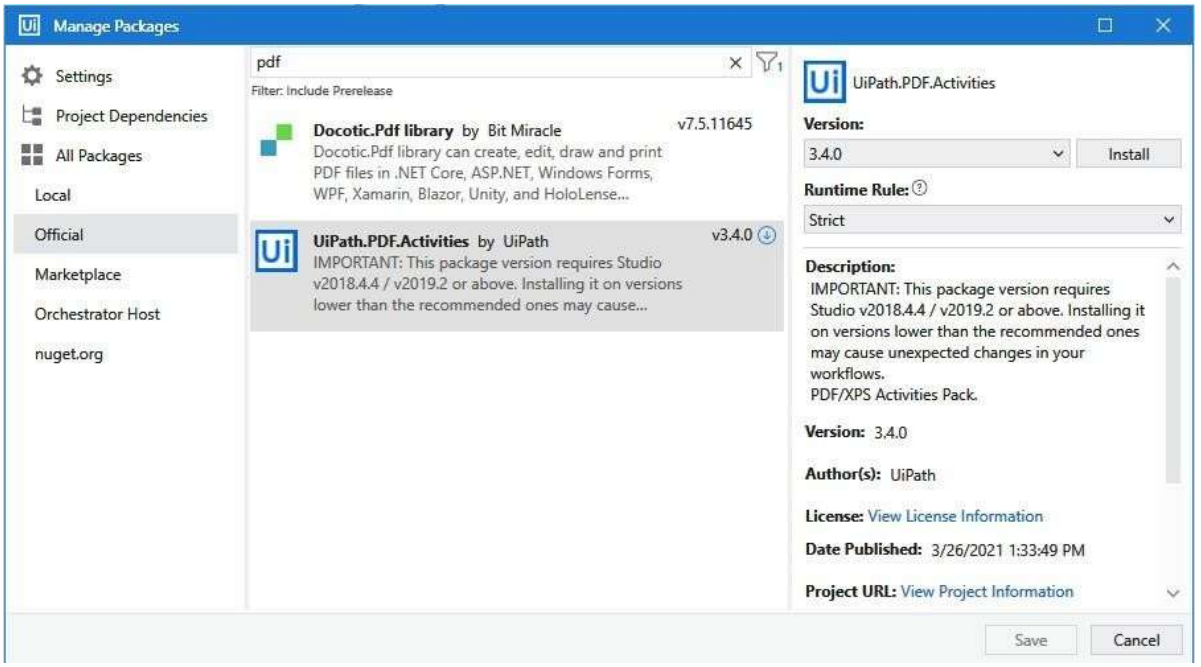
7A: Install and automate any process using UiPath.
Aim: Install and automate any process using UiPath with the following plug-ins: Any two.

Steps:

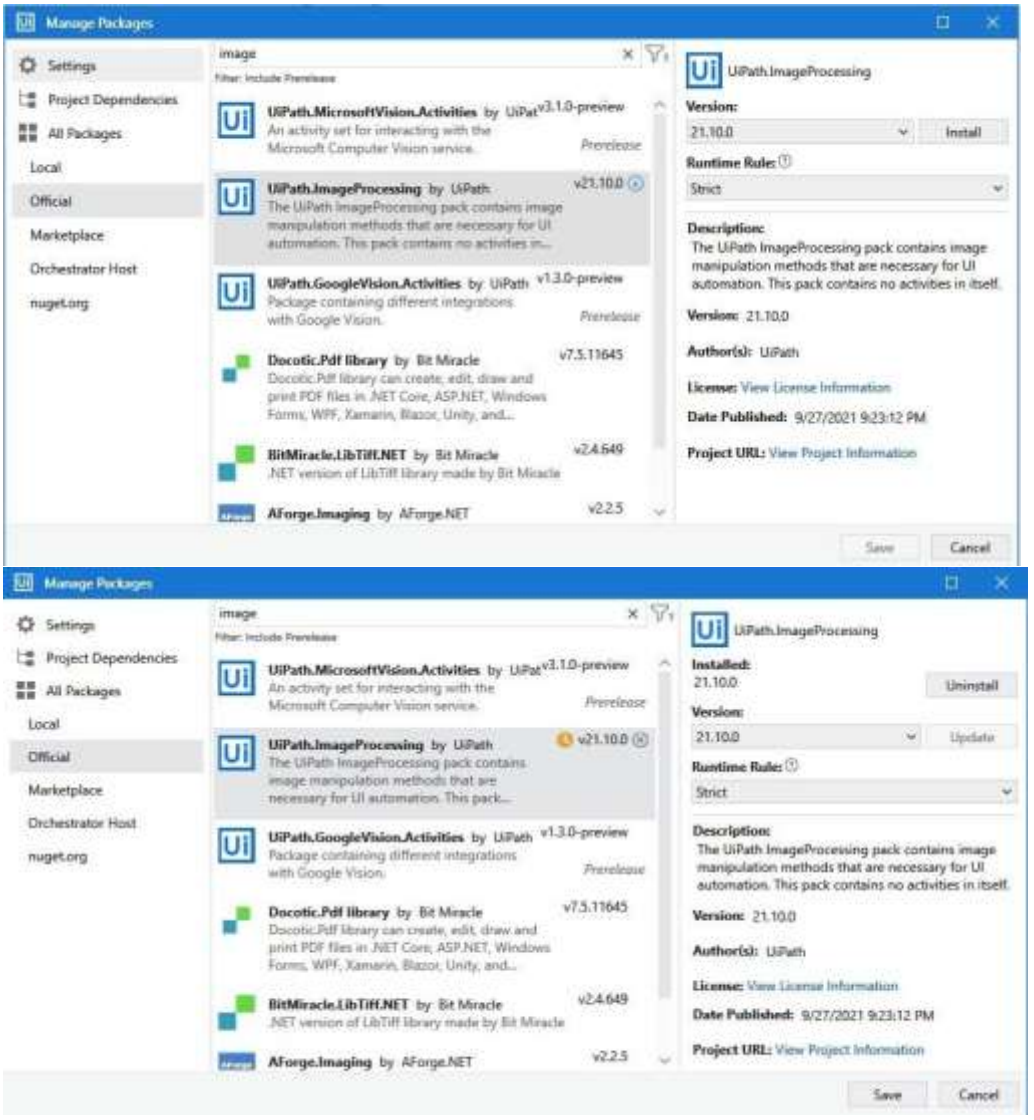
1. Create a Process→Open Ui path→Click on manage package.



2. Click on All Packages on Manage Package Pop-up widow and Search the Required Activity or Package you want to install→Select that particular activity or package and click on install→ Click on it will Automatically will save the changes.



3. Click on All Packages on Manage Package Pop-up widow and Search the Required Activity or Package you want to install→Select that particular activity or package and click on install→ Click on it will Automatically will save the changes.



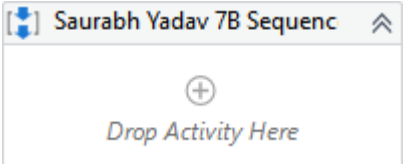
7 B) Automate the following screen scraping methods using UiPath

i. Full Test (Invoice PDF) 3 PDS (invoice)

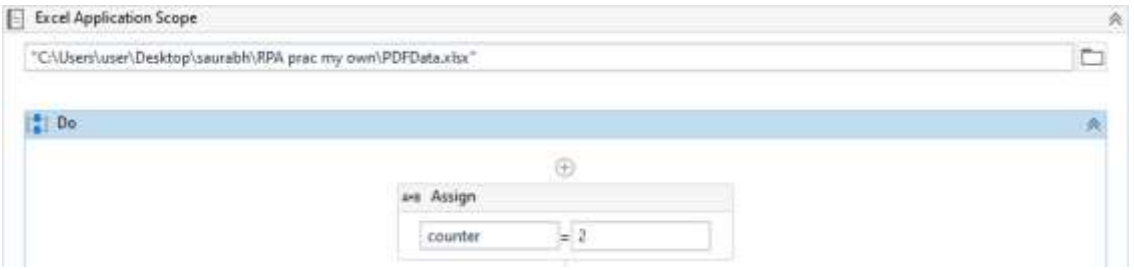
Extract and put value in Excel

Steps:

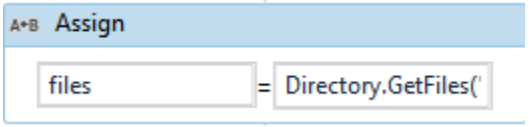
1. Create a new Blank Project and give it an appropriate name. Drag a Sequence activity from Activity tab.



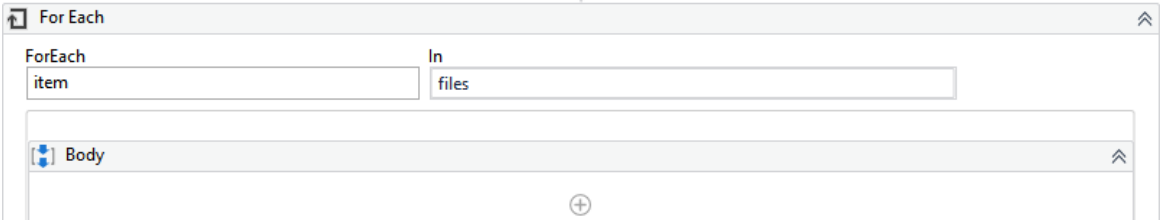
2. Add Excel Application Scope Activity give path and in an assign activity start a counter from 2.



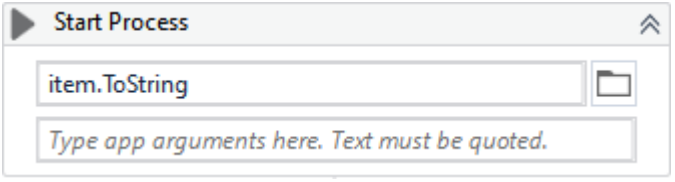
3. Add a assign activity to get the files in the given folder path.



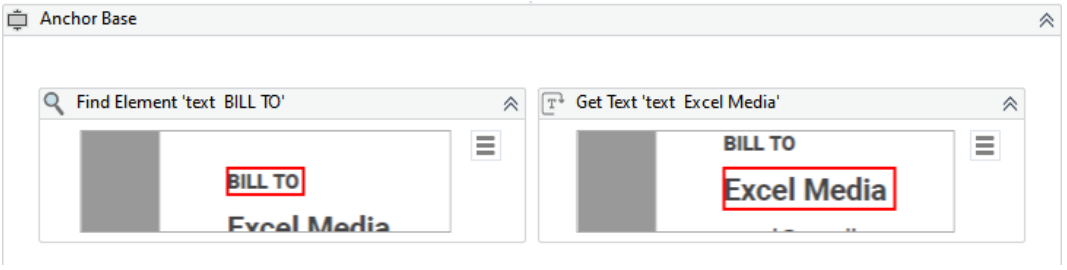
4. Add a for each activity to iterate over the files.



5. Add an Start process activity and pass the iterator variable.



6. Open one pdf and using anchor base, find element and get text extract the elements you want to get.



7 C) Automate the process of send mail event (use gmail setting).

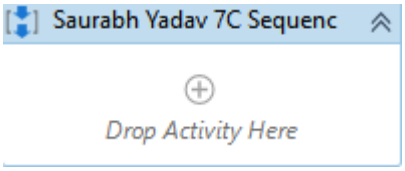
SMTP server address: smtp.gmail.com.

Gmail SMTP port (TLS): 587.

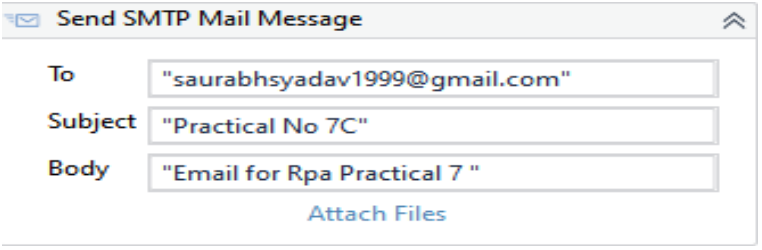
Add 3 different attachments as input.

Step:

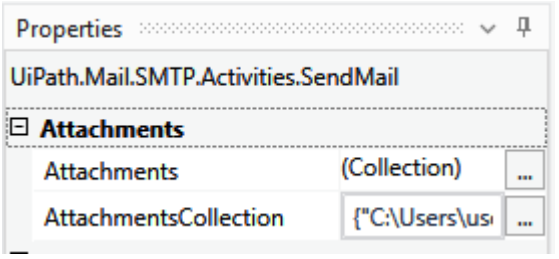
1. Create a new Blank Project and give it an appropriate name. Drag a Sequence activity from Activity tab.

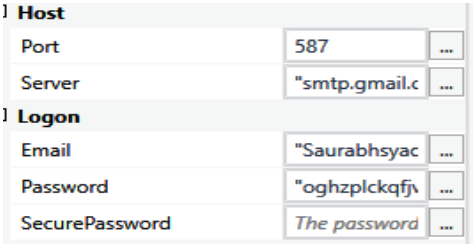


2. Add Send SMTP Mail Message Activity enter the recipient email, subject and body of the email to be sent.

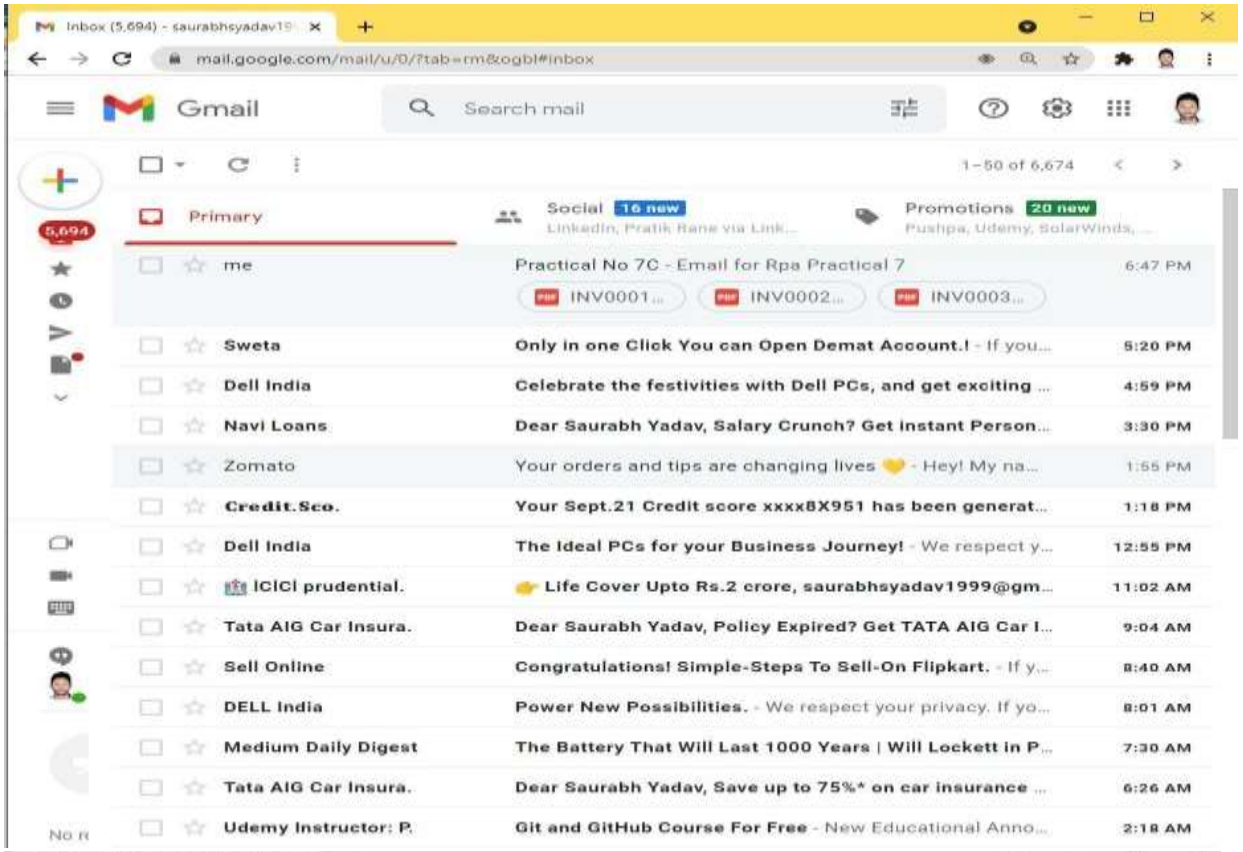


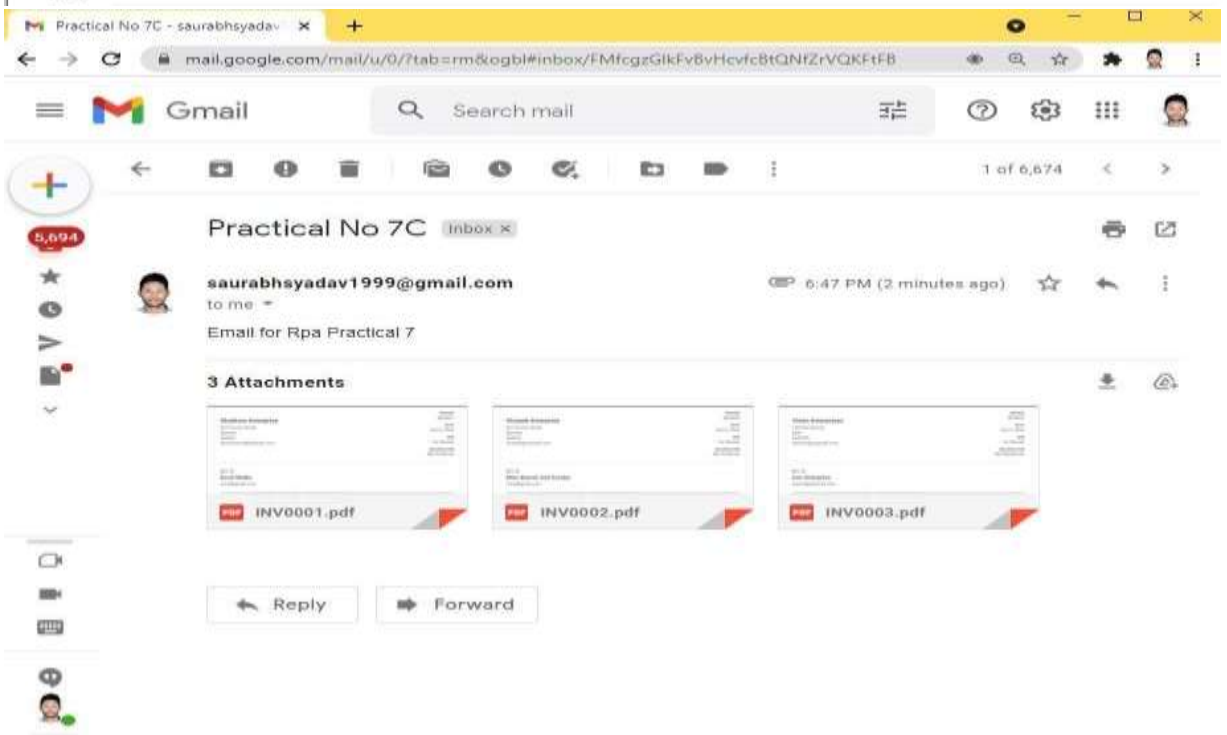
3. Enter the files variable in AttachmentsCollection Attribute. Enter smtp port number in port attribute of Host and the hostname in server field. Enter the email and password of the sender in the Logon email and password.





Output:


Inbox (5,694) - saurabhsyadav19
mail.google.com/mail/u/0/?tab=rm&ogbl#inbox
Gmail
Search mail
1–50 of 6,674
Primary
Social 16 new
LinkedIn, Pratik Rane via Link...
Promotions 20 new
Pushpa, Udemy, SolarWinds, ...
5,694
me
Practical No 7C - Email for Rpa Practical 7
6:47 PM
INV0001...
INV0002...
INV0003...
Sweta
Only in one Click You can Open Demat Account.! - If you...
5:20 PM
Dell India
Celebrate the festivities with Dell PCs, and get exciting ...
4:59 PM
Navi Loans
Dear Saurabh Yadav, Salary Crunch? Get instant Person...
3:30 PM
Zomato
Your orders and tips are changing lives - Hey! My na...
1:55 PM
Credit.Sco.
Your Sept.21 Credit score xxxx8X951 has been generat...
1:18 PM
Dell India
The Ideal PCs for your Business Journey! - We respect y...
12:55 PM
ICICI prudential.
Life Cover Upto Rs.2 crore, saurabhsyadav1999@gm...
11:02 AM
Tata AIG Car Insura.
Dear Saurabh Yadav, Policy Expired? Get TATA AIG Car I...
9:04 AM
Sell Online
Congratulations! Simple-Steps To Sell-On Flipkart. - If y...
8:40 AM
DELL India
Power New Possibilities. - We respect your privacy. If yo...
8:01 AM
Medium Daily Digest
The Battery That Will Last 1000 Years | Will Lockett in P...
7:30 AM
Tata AIG Car Insura.
Dear Saurabh Yadav, Save up to 75%* on car insurance ...
6:26 AM
Udemy Instructor: P.
Git and GitHub Course For Free - New Educational Anno...
2:18 AM


Practical No 7C - saurabhsyadav
mail.google.com/mail/u/0/?tab=rm&ogbl#inbox/FMfcgzGlkFv8vHcvfcBtQNfZrVQKFtFB
Gmail
Search mail
1 of 6,674
5,694
Practical No 7C
Inbox
saurabhsyadav1999@gmail.com
to me
6:47 PM (2 minutes ago)
Email for Rpa Practical 7
3 Attachments
INV0001.pdf
INV0002.pdf
INV0003.pdf
Reply
Forward