A

Lab Record

On

Python Programming With Problem

Master Of Computer Application - I Sem

RUNGTA INTERNATIONAL SKILLS UNIVERSITY

SESSION: 2025-26

Submitted To:-
Ms. KAVITA KANWAR

Submitted By:-
TANU CHAUDHARY
Ref No. RU-25-11704

RUNGTA INTERNATIONAL SKILLS UNIVERSITY,CG
SCHOOL OF INFORMATION TECHNOLOGY

INDEX

S. No	Name of Practical	Submission Date	Remarks
10	Write a python program using math module to calculate: a) Square root of a number b) Factorial of a number c) Power of a number		
11	Write a python program to find the area and circumference of a circle using math module, take input from the user.		
12	Create a list of student and select CR randomly, then shuffle the list and print it.		
13	Write a python program to generate a random password of 12 characters using UPPERCASE, lowercase, digits and special symbols(!, @, #, $, *). Password should start with Capital letter or digits.		
14	Create a package named shapes that contains two modules: circle.py and rectangle.py. Each module should compute area and perimeter of the shape. Write the complete directory structure and sample code.		
15	Write a program that asks the user for an index and prints the element at that index from a predefined list. Handle: • IndexError (index out of range) • TypeError (if user enters a non-integer index) • Print the specific exception message using except Exception as e.		
16	Implement a supervised learning model to classify flowers in the Iris dataset using decision tree classifier. Print the accuracy of the model.		
17	Use Support Vector Machine (SVM) on the breast cancer datasets to classify malignant vs benign tumors.		
18	Perform Principal Component Analysis (PCA) on the digits dataset and reduce the dimension to 2. Print the explained variance ratio.		
19	Apply K-Means clustering on the Iris Dataset and print cluster labels.		

AIM 10:

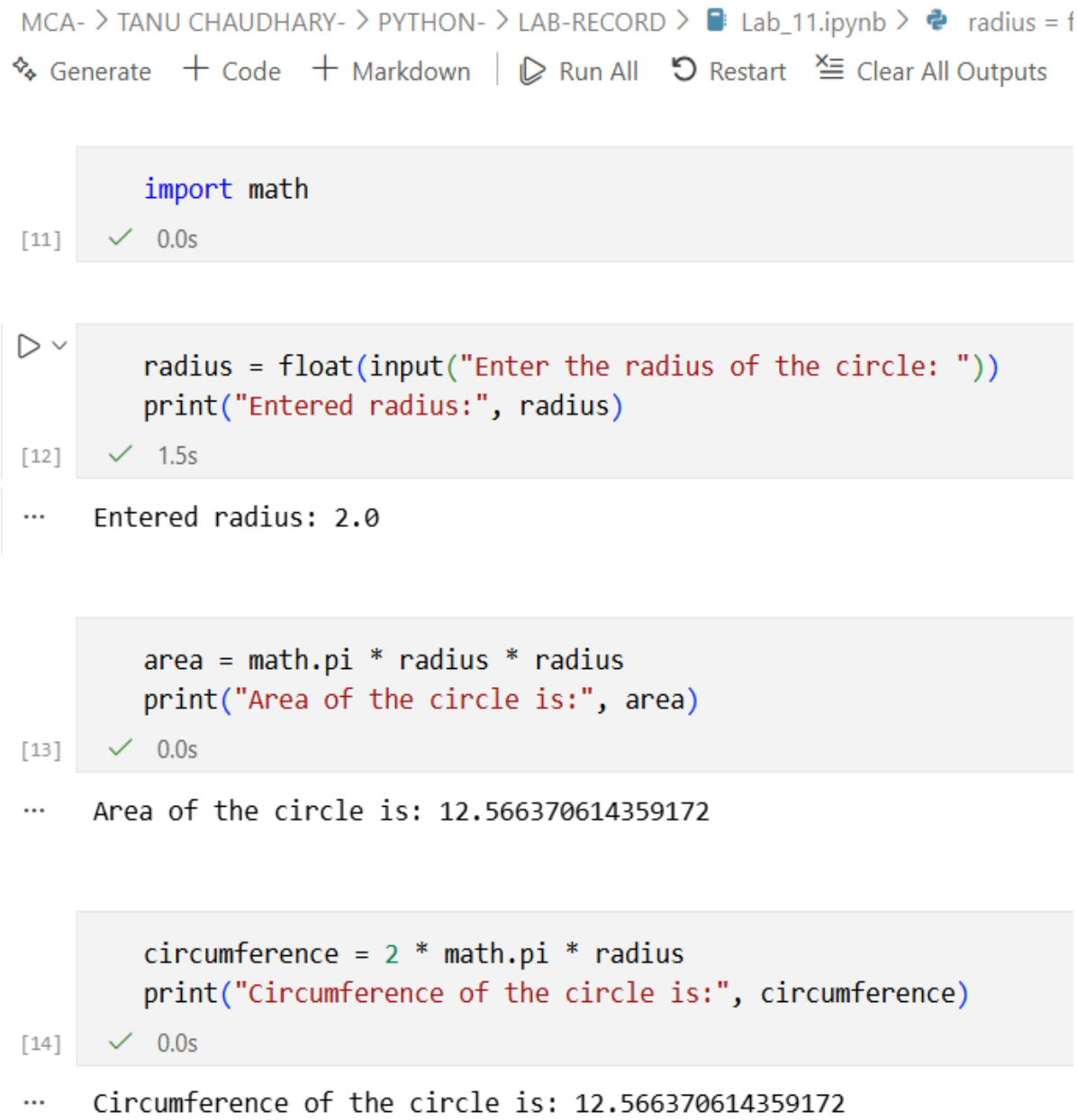
Write a python program using math module to calculate:
- Square root of a number
- Factorial of a number
- Power of a number (take input from the user)

Source Code:

AIM 11:

Write a python program to find the area and circumference of a circle using math module, take input from the user.

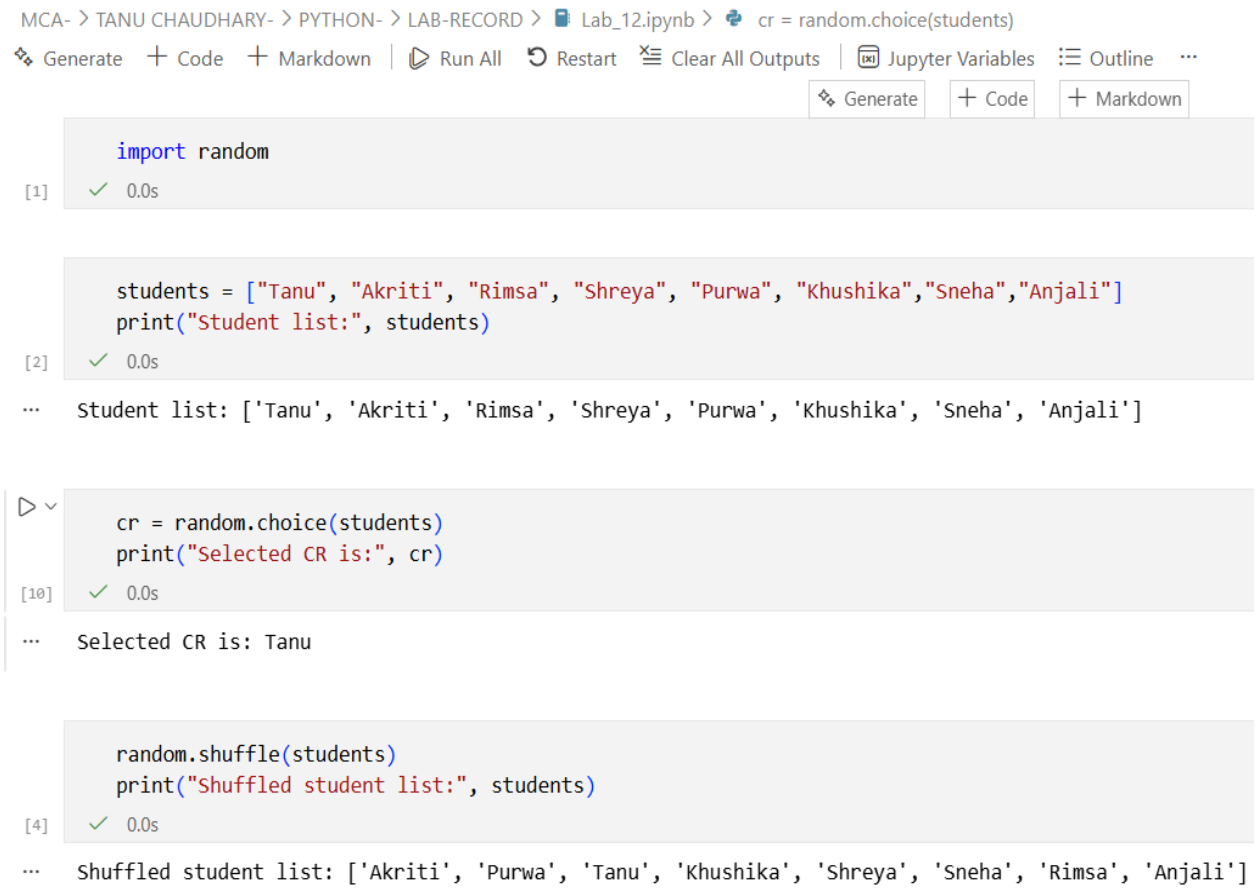
Source Code:



AIM 12:

Create a list of student and select CR randomly, then shuffle the list and print it.

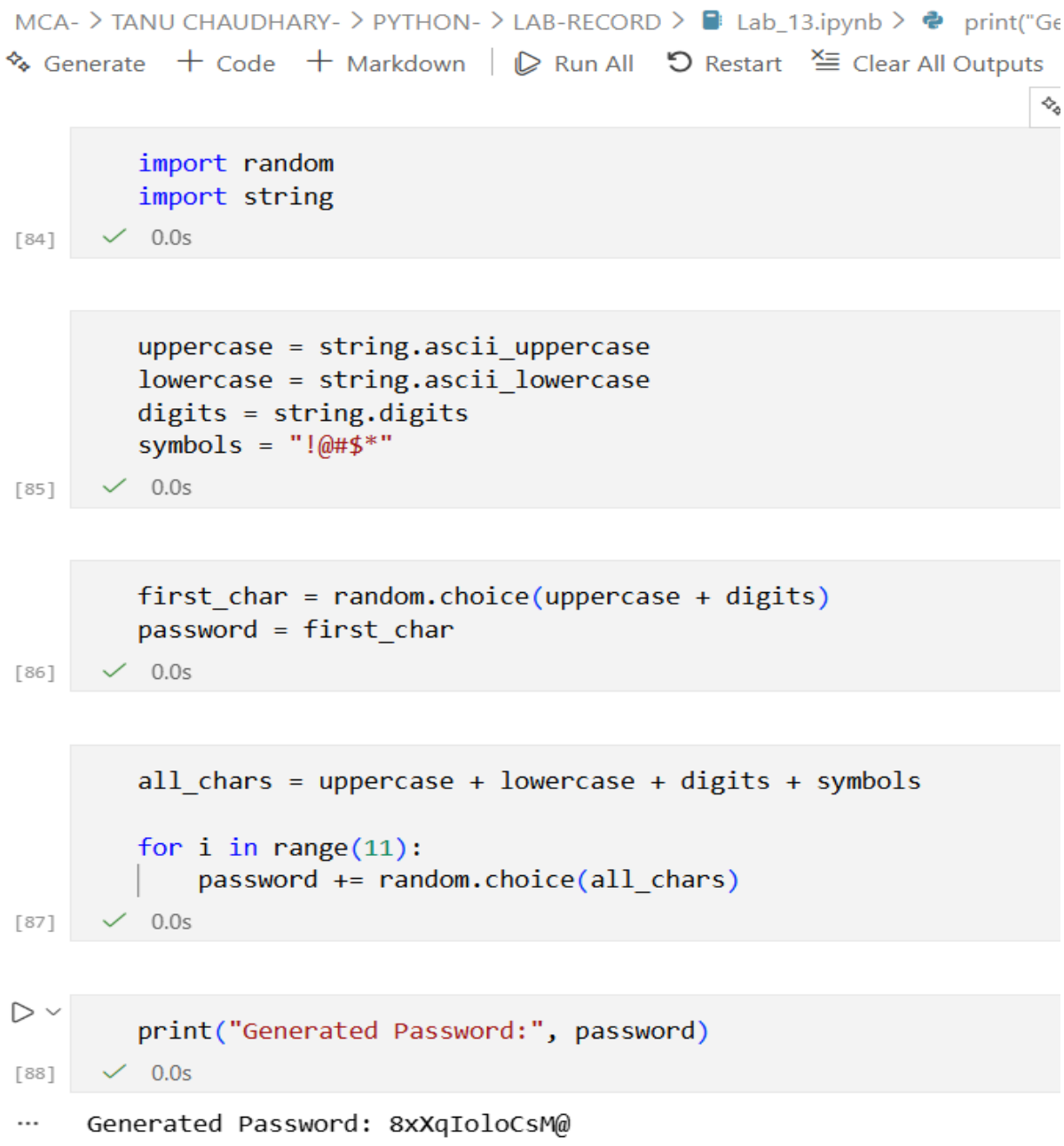
Source Code:



AIM 13:

Write a python program to generate a random password of 12 characters using UPPERCASE, lowercase, digits and special symbols(!, @, #, $, *). Password should start with Capital letter or digits.

Source Code:



AIM 14:

Create a package named shapes that contains two modules: circle.py and rectangle.py. Each module should compute area and perimeter of the shape. Write the complete directory structure and sample code.

Source Code:

MCA- > TANU CHAUDHARY- > PYTHON- > LAB-RECORD > Shapes > circle.py > ...

```python
import math

def area(radius):
    return math.pi * radius * radius

def perimeter(radius):
    return 2 * math.pi * radius
```

MCA- > TANU CHAUDHARY- > PYTHON- > LAB-RECORD > Shapes > rectangle.py >

```python
def area(length, breadth):
    return length * breadth

def perimeter(length, breadth):
    return 2 * (length + breadth)
```

MCA- > TANU CHAUDHARY- > PYTHON- > LAB-RECORD > Lab_14.py > ...

```python
from Shapes import circle, rectangle

# Circle
r = float(input("Enter radius of the circle: "))

print("Circle Area:", circle.area(r))
print("Circle Perimeter:", circle.perimeter(r))

# Rectangle
l = float(input("Enter length of the rectangle: "))
b = float(input("Enter breadth of the rectangle: "))

print("Rectangle Area:", rectangle.area(l, b))
print("Rectangle Perimeter:", rectangle.perimeter(l, b))
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```
PS C:\Users\ASUS\Documents> & C:/Users/ASUS/AppData/Local/Microsoft
Enter radius of the circle: 2
Circle Area: 12.566370614359172
Circle Perimeter: 12.566370614359172
Enter length of the rectangle: 3
Enter breadth of the rectangle: 4
Rectangle Area: 12.0
Rectangle Perimeter: 14.0
```

AIM 15:

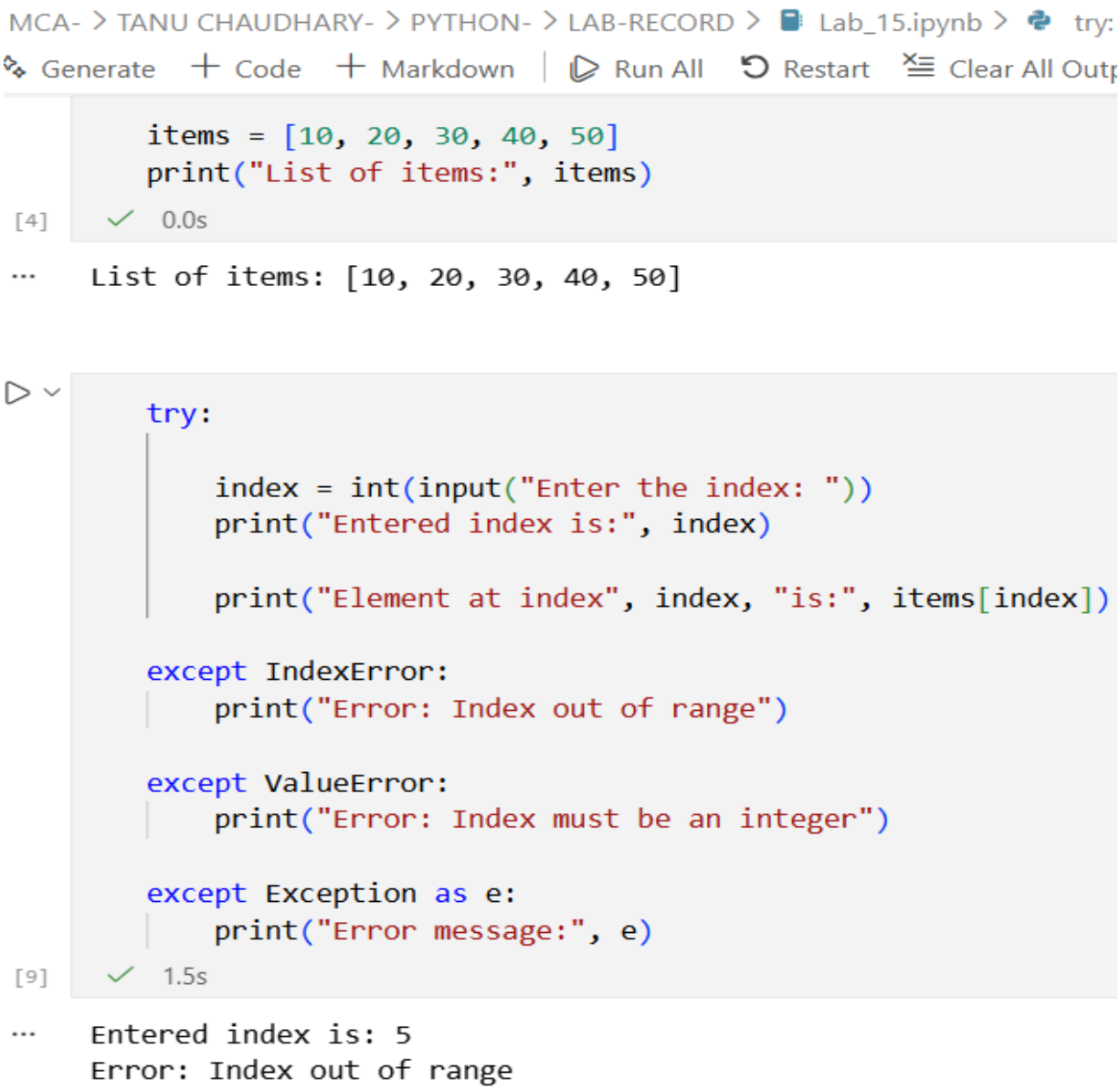
Write a program that asks the user for an index and prints the element at that index from a predefined list. Handle:

• IndexError (index out of range)

• TypeError (if user enters a non-integer index)

• Print the specific exception message using except Exception as e.

Source Code:



AIM 16:

Implement a supervised learning model to classify flowers in the Iris dataset using decision tree classifier. Print the accuracy of the model.

Source Code:

MCA- > TANU CHAUDHARY- > PYTHON- > LAB-RECORD > Lab_16.ipynb > X_train, X_test, y_train, y_test = train_test_split(

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline

```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
```

[1] ✓ 0.9s

```
iris = load_iris()
X = iris.data
y = iris.target

df = pd.DataFrame(X, columns=iris.feature_names)
df['target'] = y
df.head()
```

[2] ✓ 0.0s

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
X_train.shape, X_test.shape
```

[3] ✓ 0.0s

```
((120, 4), (30, 4))
```

```
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
```

[5] ✓ 0.0s

```
DecisionTreeClassifier
▶ Parameters
```

```
y_pred = dt_model.predict(X_test)
y_pred
```

[6] ✓ 0.0s

```
array([1, 0, 2, 1, 1, 0, 1, 2, 1, 1, 2, 0, 0, 0, 0, 1, 2, 1, 1, 2, 0, 2,
       0, 2, 2, 2, 2, 2, 0, 0])
```

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy of Decision Tree Classifier:", accuracy)
```

[7] ✓ 0.0s

```
Accuracy of Decision Tree Classifier: 1.0
```

AIM 17:

Use Support Vector Machine (SVM) on the breast cancer datasets to classify malignant vs benign tumors.

Source Code:

MCA- > TANU CHAUDHARY- > PYTHON- > LAB-RECORD > Lab_17.ipynb > svm_model = SVC(kernel='linear', random_state=42)

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | ... | Python 3.11.9

Generate | Code | Markdown

```
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

[1] ✓ 1.0s Python

```
cancer = load_breast_cancer()
X = cancer.data
y = cancer.target

df = pd.DataFrame(X, columns=cancer.feature_names)
df['target'] = y
df.head()
```

[2] ✓ 0.0s Python

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	target
	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890	0
	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902	0
	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758	0
	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300	0
	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678	0

rows × 31 columns

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
X_train.shape, X_test.shape
```

[3] ✓ 0.0s Python

```
((455, 30), (114, 30))
```

```
svm_model = SVC(kernel='linear', random_state=42)
svm_model.fit(X_train, y_train)
```

[4] ✓ 0.8s

```
SVC
▸ Parameters
```

```
y_pred = svm_model.predict(X_test)
y_pred
```

[5] ✓ 0.0s

```
array([1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 0, 0])
```

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy of SVM Classifier:", accuracy)

print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

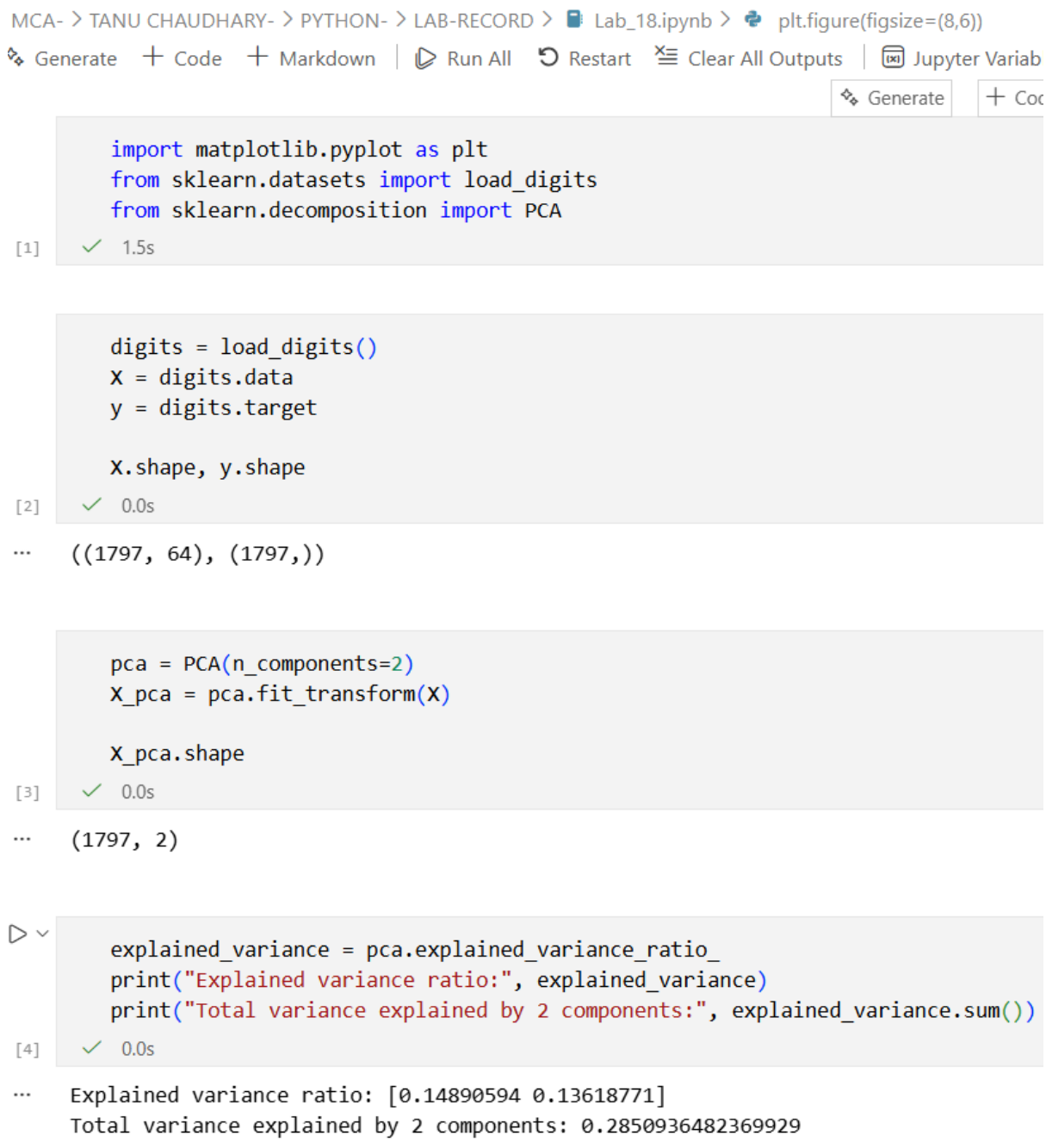
[6] ✓ 0.0s

```
Accuracy of SVM Classifier: 0.956140350877193

Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.91      0.94        43
           1       0.95      0.99      0.97        71

    accuracy                           0.96       114
   macro avg       0.96      0.95      0.95       114
weighted avg       0.96      0.96      0.96       114


Confusion Matrix:
 [[39  4]
 [ 1 70]]
```

AIM 18:

Perform Principal Component Analysis (PCA) on the digits dataset and reduce the dimension to 2. Print the explained variance ratio.

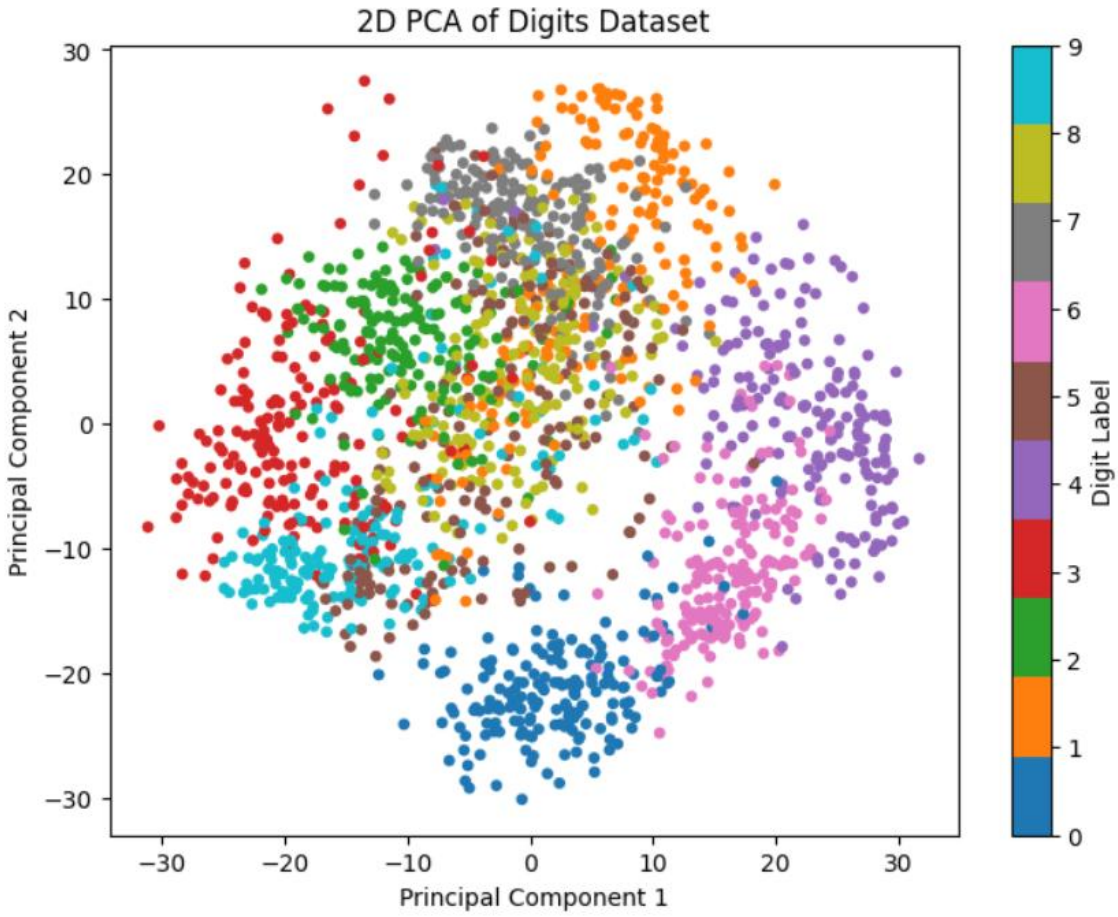
Source Code:



```
plt.figure(figsize=(8,6))
plt.scatter(X_pca[:,0], X_pca[:,1], c=y, cmap='tab10', s=15)
plt.colorbar(label='Digit Label')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title('2D PCA of Digits Dataset')
plt.show()
```

[5] ✓ 3.5s

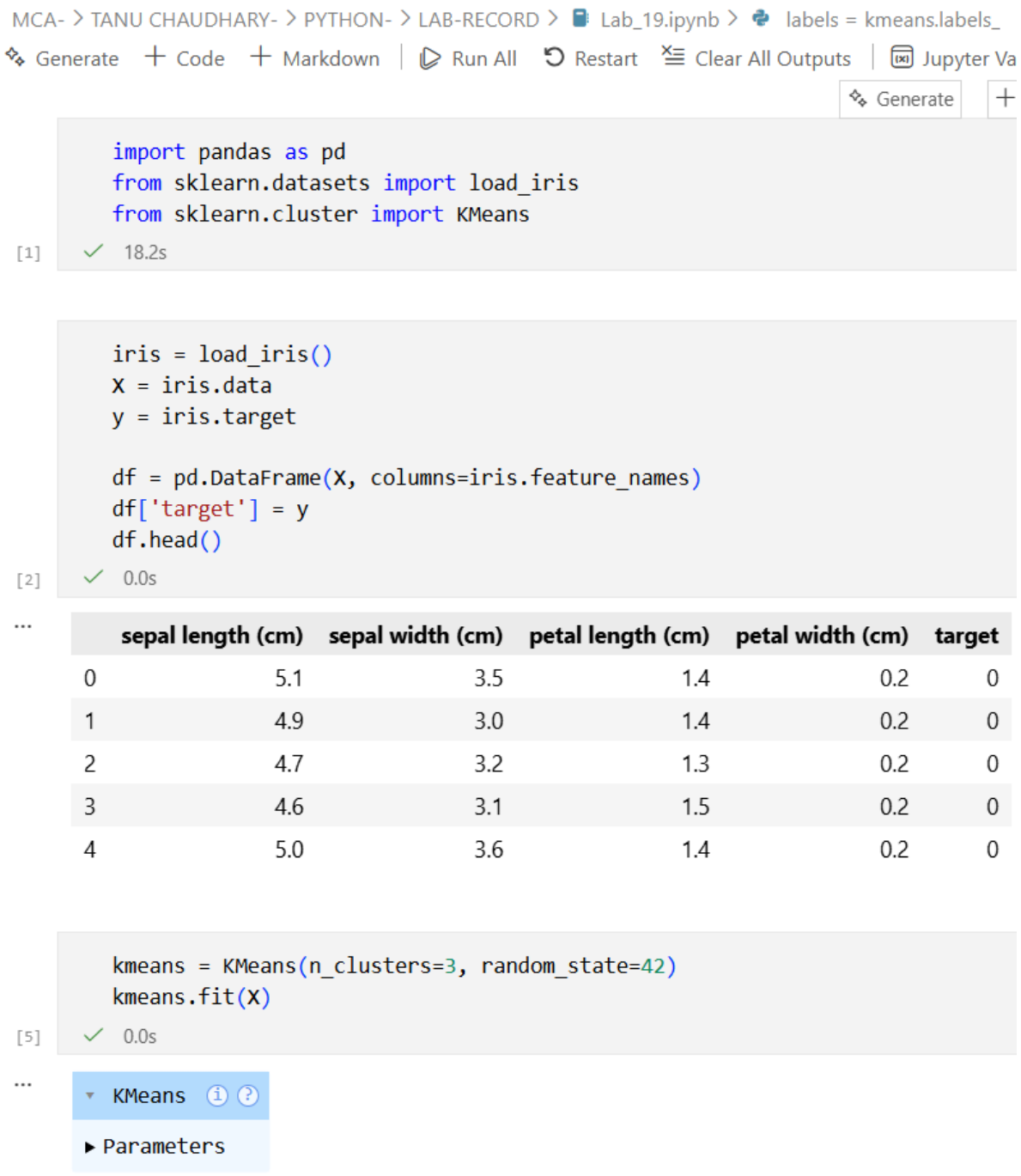
...



AIM 19:

Apply K-Means clustering on the Iris Dataset and print cluster labels.

Source Code:



```
labels = kmeans.labels_
print("Cluster labels for each data point:\n", labels)
```

[6] ✓ 0.0s

```
Cluster labels for each data point:
 [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 0 2 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 2 0 0 0 0 2 0 0 0 0
 0 0 2 2 0 0 0 0 2 0 2 0 2 0 0 2 2 0 0 0 0 0 2 0 0 0 0 2 0 0 0 2 0 0 0 2 0
 0 2]
```

```
df['cluster'] = labels
df.head(10)
```

[7] ✓ 0.0s

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target	cluster
0	5.1	3.5	1.4	0.2	0	1
1	4.9	3.0	1.4	0.2	0	1
2	4.7	3.2	1.3	0.2	0	1
3	4.6	3.1	1.5	0.2	0	1
4	5.0	3.6	1.4	0.2	0	1
5	5.4	3.9	1.7	0.4	0	1
6	4.6	3.4	1.4	0.3	0	1
7	5.0	3.4	1.5	0.2	0	1
8	4.4	2.9	1.4	0.2	0	1
9	4.9	3.1	1.5	0.1	0	1