KLS Vishwanathrao Deshpande Institute of Technology Haliyal– 581329

Machine learning Lab [BCSL606]

for

VI Semester B.E.

As prescribed by

VISVESVARAYA TECHNOLOGICAL UNIVERSITY,BELAGAVI–590014

(For the Academic Year 2025-2026)

Prepared by

Prof.Amarnath Swami

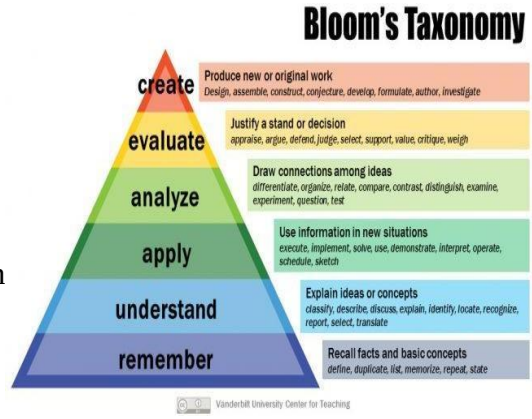
Department of Computer Science & Engineering(AIML)

Vision(College)

PROGRAMOUTCOMES(POs)

Program Outcomes as defined by NBA(PO)Engineering Graduates will be able to:

1. **Engineeringknowledge**:Applytheknowledgeofmathematics,science,engineeringfundam`entals,andanengineeringspecializationtothesolutionofcomplexengineeringproblems.
2. **Problemanalysis**:Identify,formulate,reviewresearchliterature,andanalyzecomplexengineering problems reaching substantiated conclusions using first principles of mathematics,naturalsciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems anddesignsystemcomponentsorprocessesthatmeetthespecifiedneedswithappropriateconsideration for the public health and safety, and the cultural, societal, and environmentalconsiderations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and researchmethods including design of experiments, analysis and interpretation of data, and synthesis oftheinformation to provide validconclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modernengineering and IT tools including prediction and modeling to complex engineering activitieswithan understanding of thelimitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environmentandsustainability**:Understandtheimpactoftheprofessionalengineeringsolutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**:Communicateeffectivelyoncomplexengineeringactivitieswiththeengineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management andfinance**:Demonstrateknowledgeandunderstanding of the engineering andmanagementprinciplesandapplythesetoone’sownwork,asamemberandleaderinateam,tomanageprojectsandinmultidisciplinaryenvironments.
12. **Lifelonglearning**:Recognizetheneedfor,andhavethepreparationand ability to engage in independentand life-long learning in the broadestcontextof technologicalchange.



To nurture talent & enrich society through excellence in technical education, research & innovation.
Mission(College)
To augment innovative Pedagogy & kindle quest for interdisciplinary learning & to enhance conceptual understanding. To build competence, professional ethics & develop entrepreneurial thinking. To strengthen Industry Institute Partnership & explore global collaborations. To inculcate culture of socially responsible citizenship. To focus on Holistic & Sustainable development.
Vision(Dept)
To lead the way in the creation and application of cutting-edge Computer science and engineering (AIML) technologies, advancing the frontiers of knowledge, and empowering future generations to drive innovation and transformation on a global scale.
Mission(Dept)
To train students with a strong conceptual understanding using innovative pedagogies, empowering them to excel in the dynamic fields of Artificial Intelligence and Machine Learning. To imbibe professional, research, and entrepreneurial skills with a commitment to the nation's development at large. To strengthen the industry-institute Interaction. To promote life–long learning with a sense of societal & ethical responsibilities.

ProgramEducationalObjectives(PEO)	
PEO1	TodevelopanabilitytoidentifyandanalyzetherequirementsofComputer ScienceandEngineeringin designandproviding novel engineeringsolutions.
PEO2	Todevelopabilitiestoworkinteamonmultidisciplinaryprojectswitheffective communicationskills,ethicalqualitiesandleadershiproles.
PEO3	Todevelop abilitiesforsuccessfulComputer ScienceEngineer andachievehigher career goals.
ProgramSpecific Outcomes(PSO)	
PSO1	To develop abilities to model real world problems using appropriate algorithms, computational theories and programming languages in the area of AI and ML..
PSO2	To develop software applications and products in specialized areas of Artificial Intelligence and Machine Learning.

Evaluation:

	Particulars	**Marks**	**Total**
CIA	Performance	20	**30**
	Journal	05	
	Viva-voce	05	
	Add-on Course	05	**05**
	Virtual Lab Experiments	05	**05**
	LabIA	10	**10**

Mappingof Experimentswith CO, PO and PSO

S.No.	ExperimentDetails	CO	PO	PSO
1	Develop a program to create histograms for all numerical features and analyze the distribution of each feature. Generate box plots for all numerical features and identify any outliers. Use California Housing dataset.	1	1,2,4,5	1
2	Develop a program to Compute the correlation matrix to understand the relationships between pairs of features. Visualize the correlation matrix using a heatmap to know which variables have strong positive/negative correlations. Create a pair plot to visualize pairwise relationships between features. Use California Housing dataset.	1	1,2,3	1
3	Develop a program to implement Principal Component Analysis (PCA) for reducing the dimensionality of the Iris dataset from 4 features to 2	2	1,2,3,	1
4	For a given set of training data examples stored in a .CSV file, implement and demonstrate the Find-S algorithm to output a description of the set of all hypotheses consistent with the training examples.	2	1,2,3	1
5	Develop a program to implement k-Nearest Neighbour algorithm to classify the randomly generated 100 values of x in the range of [0,1]. Perform the following based on dataset generated. 1. Label the first 50 points $\{x_1,\ldots\ldots,x_{50}\}$ as follows: if (xi ≤ 0.5), then $x_i \in$ $Class_1$, else $x_i \in Class_1$ 2. Classify the remaining points, $x_{51},\ldots\ldots,x_{100}$ using KNN. Perform this for $k=1,2,3,4,5,20,30$	3	1,3,4,5	1
6	Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs	3,4	1,2,5	1
7	Develop a program to demonstrate the working of Linear Regression and Polynomial Regression. Use Boston Housing Dataset for Linear Regression and Auto MPG Dataset (for vehicle fuel efficiency prediction) for Polynomial Regression.	4	1,2,5	1
8	Develop a program to demonstrate the working of the decision tree algorithm. Use Breast Cancer Data set for building the decision tree and apply this knowledge to classify a new sample.	4	1,2,5	1
9	Develop a program to implement the Naive Bayesian classifier considering Olivetti Face Data set for training. Compute the accuracy of the classifier, considering a few test data sets.	4,5	1,2,5	1
10	Develop a program to implement k-means clustering using Wisconsin Breast Cancer data set and visualize the clustering result.	4,5	1,2,5	1

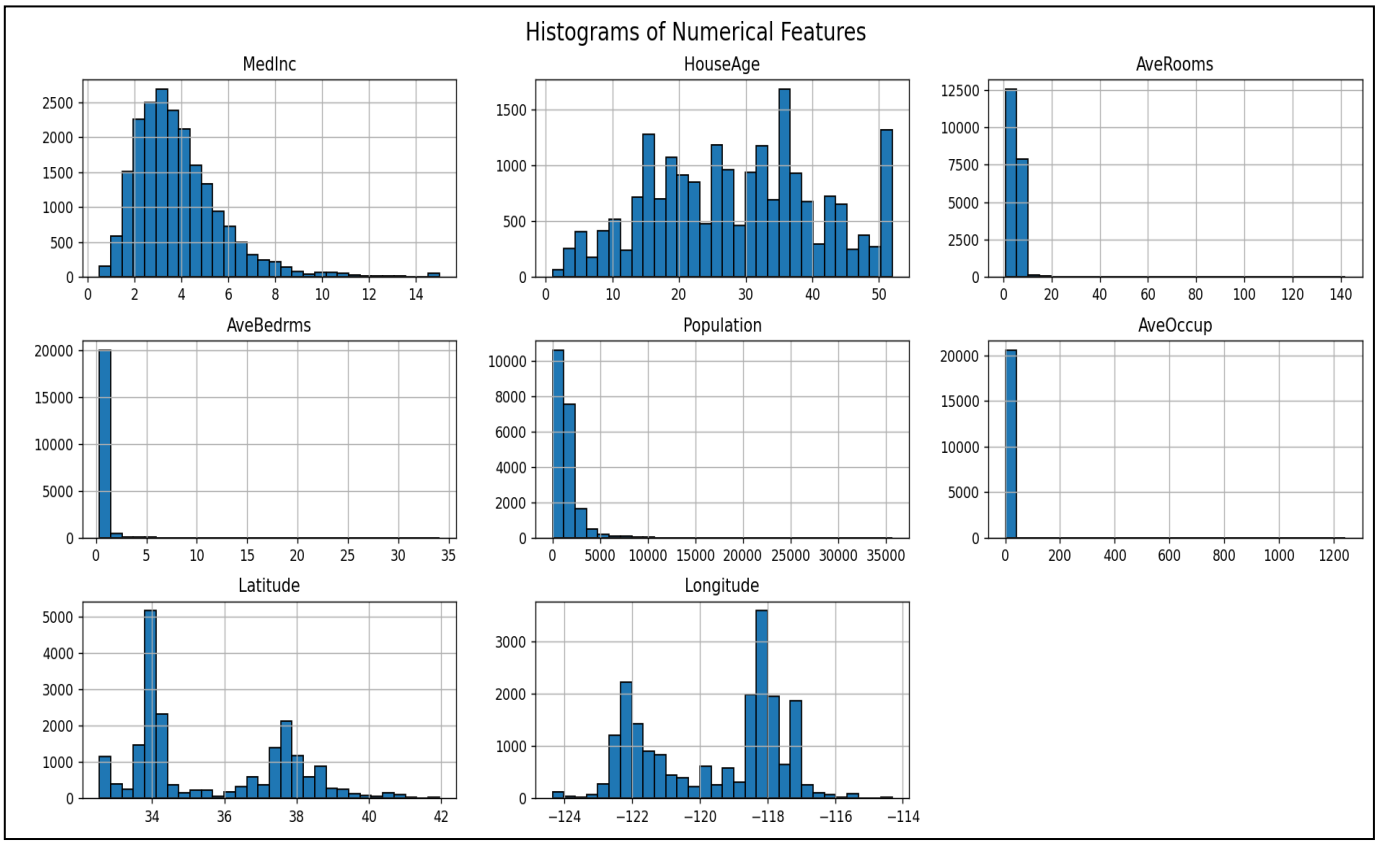
Sl.NO	Experiments
1	Develop a program to create histograms for all numerical features and analyze the distribution of each feature. Generate box plots for all numerical features and identify any outliers. Use California Housing dataset.
2	Develop a program to Compute the correlation matrix to understand the relationships between pairs of features. Visualize the correlation matrix using a heatmap to know which variables have strong positive/negative correlations. Create a pair plot to visualize pairwise relationships between features. Use California Housing dataset.
3	Develop a program to implement Principal Component Analysis (PCA) for reducing the dimensionality of the Iris dataset from 4 features to 2.
4	For a given set of training data examples stored in a .CSV file, implement and demonstrate the Find-S algorithm to output a description of the set of all hypotheses consistent with the training examples.
5	Develop a program to implement k-Nearest Neighbour algorithm to classify the randomly generated 100 values of *x* in the range of [0,1]. Perform the following based on dataset generated. 3. Label the first 50 points $\{x_1,\ldots\ldots,x_{50}\}$ as follows: if (xi ≤ 0.5), then $x_i \in \text{Class}_1$, else $x_i \in \text{Class}_1$ 4. Classify the remaining points, $x_{51},\ldots\ldots,x_{100}$ using KNN. Perform this for $k=1,2,3,4,5,20,30$
6	Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs
7	Develop a program to demonstrate the working of Linear Regression and Polynomial Regression. Use Boston Housing Dataset for Linear Regression and Auto MPG Dataset (for vehicle fuel efficiency prediction) for Polynomial Regression.
8	Develop a program to demonstrate the working of the decision tree algorithm. Use Breast Cancer Data set for building the decision tree and apply this knowledge to classify a new sample.
9	Develop a program to implement the Naive Bayesian classifier considering Olivetti Face Data set for training. Compute the accuracy of the classifier, considering a few test data sets.
10	Develop a program to implement k-means clustering using Wisconsin Breast Cancer data set and visualize the clustering result.
	Virtual Labs
11	To learn about extended argument syntax and how we can use it to make python code much simpler
12	This experiment aims at learning the concept of lambda functions, a very useful concept taken from functional programming

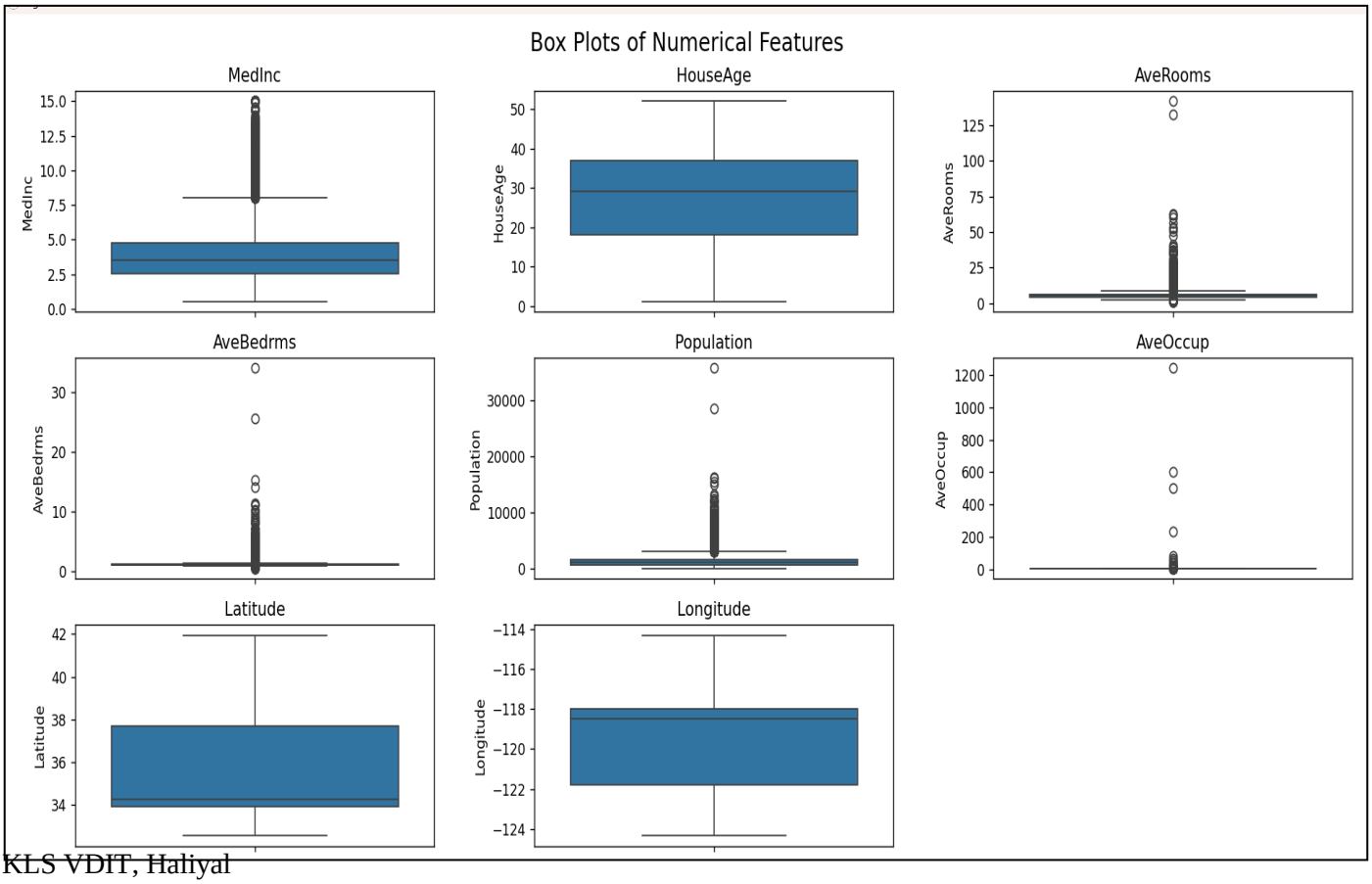
PROGRAM 1

Develop a program to create histograms for all numerical features and analyze the distribution of each feature. Generate box plots for all numerical features and identify any outliers. Use California Housing dataset.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import fetch_california_housing

# Load the California Housing dataset
data = fetch_california_housing()
df = pd.DataFrame(data.data, columns=data.feature_names)

# Create histograms for all numerical features
plt.figure(figsize=(12, 8))
df.hist(bins=30, figsize=(12, 8), layout=(3, 3), edgecolor='black')
plt.suptitle("Histograms of Numerical Features", fontsize=16)
plt.tight_layout()
plt.show()

# Generate box plots for all numerical features to identify outliers
plt.figure(figsize=(12, 8))
for i, column in enumerate(df.columns):
    plt.subplot(3, 3, i+1)
    sns.boxplot(y=df[column])
    plt.title(column)
plt.suptitle("Box Plots of Numerical Features", fontsize=16)
plt.tight_layout()
plt.show()
```

OUTPUT


Histograms of Numerical Features
MedInc
HouseAge
AveRooms
AveBedrms
Population
AveOccup
Latitude
Longitude

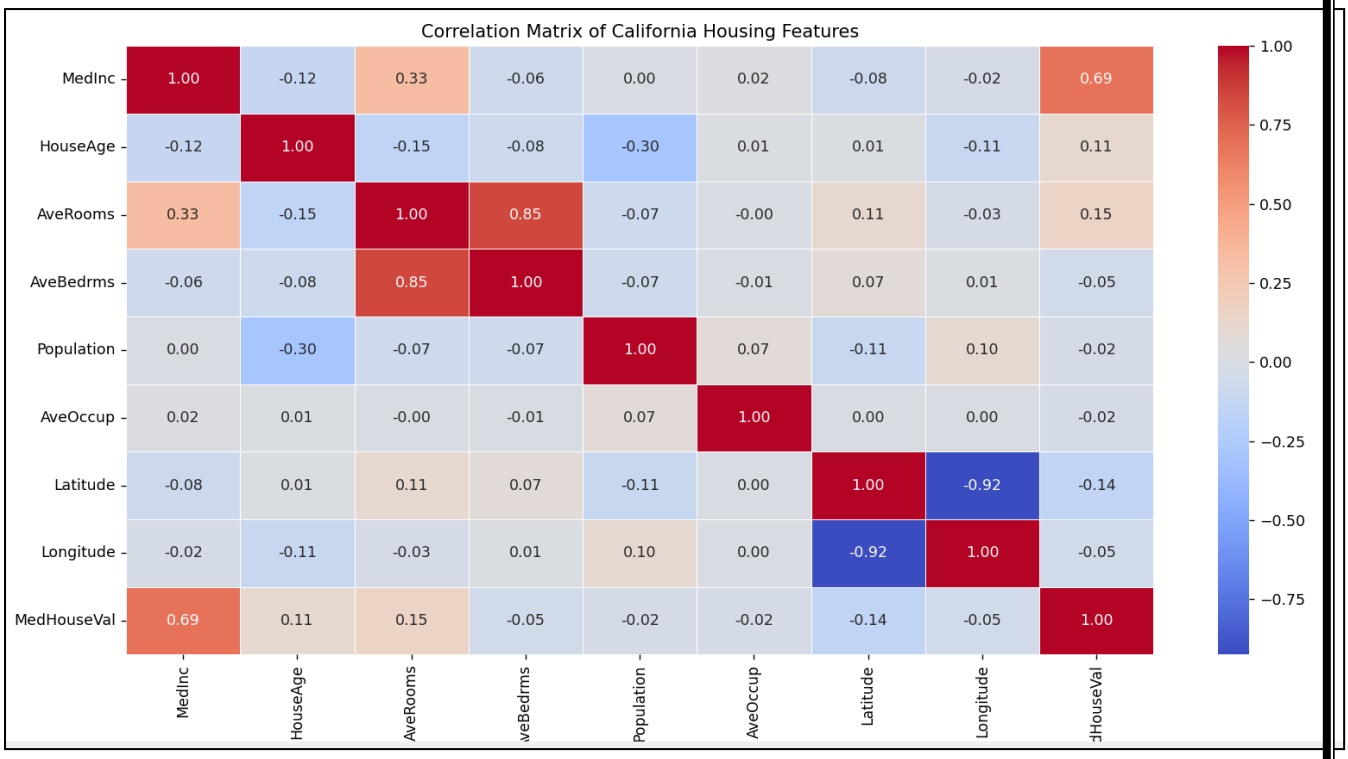

Box Plots of Numerical Features
MedInc
HouseAge
AveRooms
AveBedrms
Population
AveOccup
Latitude
Longitude

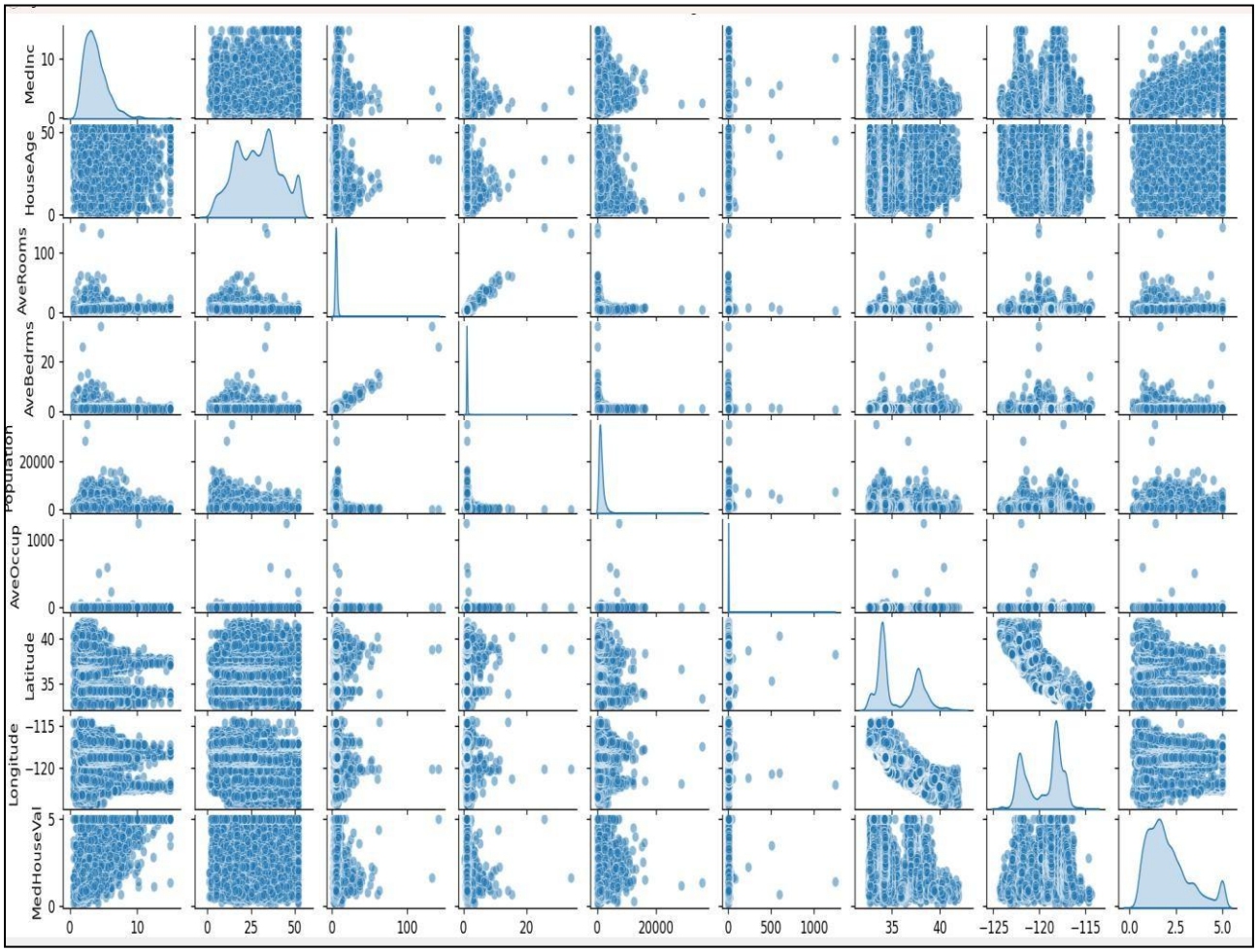
PROGRAM 2

Develop a program to Compute the correlation matrix to understand the relationships between pairs of features. Visualize the correlation matrix using a heatmap to know which variables have strong positive/negative correlations. Create a pair plot to visualize pairwise relationships between features. Use California Housing dataset.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_california_housing

# Step 1: Load the California Housing Dataset
california_data = fetch_california_housing(as_frame=True)
data = california_data.frame

# Step 2: Compute the correlation matrix
correlation_matrix = data.corr()

# Step 3: Visualize the correlation matrix using a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Correlation Matrix of California Housing Features')
plt.show()

# Step 4: Create a pair plot to visualize pairwise relationships
sns.pairplot(data, diag_kind='kde', plot_kws={'alpha': 0.5})
plt.suptitle('Pair Plot of California Housing Features', y=1.02)
plt.show()
```

OUTPUT


Correlation Matrix of California Housing Features
MedInc
HouseAge
AveRooms
AveBedrms
Population
AveOccup
Latitude
Longitude
MedHouseVal
1.00 -0.12 0.33 -0.06 0.00 0.02 -0.08 -0.02 0.69
-0.12 1.00 -0.15 -0.08 -0.30 0.01 0.01 -0.11 0.11
0.33 -0.15 1.00 0.85 -0.07 -0.00 0.11 -0.03 0.15
-0.06 -0.08 0.85 1.00 -0.07 -0.01 0.07 0.01 -0.05
0.00 -0.30 -0.07 -0.07 1.00 0.07 -0.11 0.10 -0.02
0.02 0.01 -0.00 -0.01 0.07 1.00 0.00 0.00 -0.02
-0.08 0.01 0.11 0.07 -0.11 0.00 1.00 -0.92 -0.14
-0.02 -0.11 -0.03 0.01 0.10 0.00 -0.92 1.00 -0.05
0.69 0.11 0.15 -0.05 -0.02 -0.02 -0.14 -0.05 1.00
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75


MedInc
HouseAge
AveRooms
AveBedrms
Population
AveOccup
Latitude
Longitude
MedHouseVal

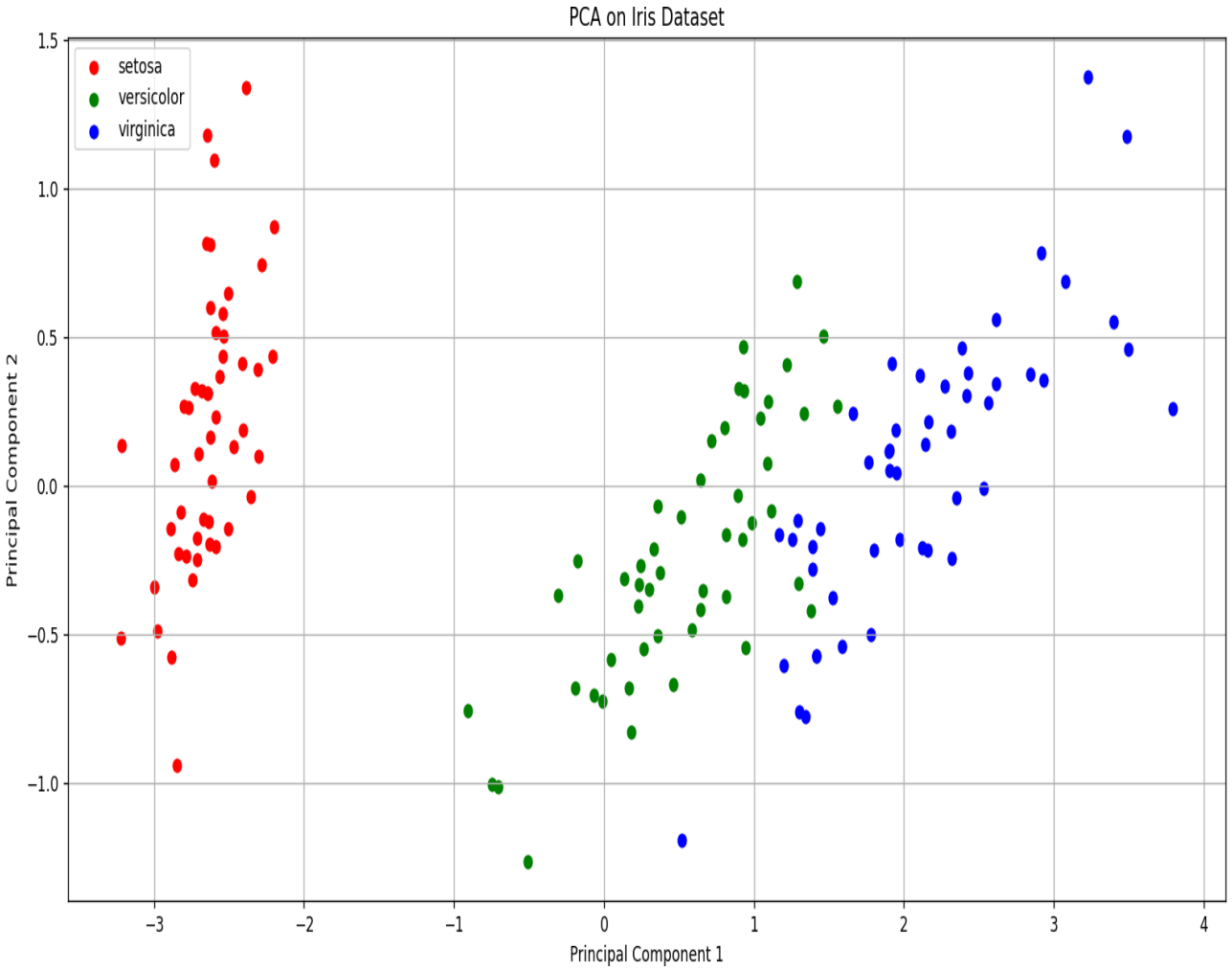
PROGRAM -3

Develop a program to implement Principal Component Analysis (PCA) for reducing the dimensionality of the Iris dataset from 4 features to 2.

```
Import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
# Load the Iris dataset
iris = load_iris()
data = iris.data
labels = iris.target
label_names = iris.target_names
# Convert to a DataFrame for better visualization
iris_df = pd.DataFrame(data, columns=iris.feature_names)
# Perform PCA to reduce dimensionality to 2
pca = PCA(n_components=2)
data_reduced = pca.fit_transform(data)
# Create a DataFrame for the reduced data
reduced_df = pd.DataFrame(data_reduced, columns=['Principal Component 1', 'Principal Component 2'])
reduced_df['Label'] = labels
# Plot the reduced data
plt.figure(figsize=(8, 6))
colors = ['r', 'g', 'b']
```

```
for I, label in enumerate(np.unique(labels)):
    plt.scatter(
        reduced_df[reduced_df['Label'] == label]['Principal Component 1'],
        reduced_df[reduced_df['Label'] == label]['Principal Component 2'],
        label=label_names[label],
        color=colors[i]
    )
plt.title('PCA on Iris Dataset')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend()
plt.grid()
plt.show()
```

OUTPUT


PCA on Iris Dataset
setosa
versicolor
virginica
Principal Component 2
Principal Component 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−3
−2
−1
0
1
2
3
4

PROGRAM-4

For a given set of training data examples stored in a .CSV file, implement and demonstrate the Find-S algorithm to output a description of the set of all hypotheses consistent with the training examples.

```
import pandas as pd
def find_s_algorithm(file_path):
    data = pd.read_csv(file_path)
    print("Training data:")
    print(data)
    attributes = data.columns[:-1]
    class_label = data.columns[-1]
    hypothesis = ['?' for _ in attributes]
    for index, row in data.iterrows():
        if row[class_label] == 'Yes':
            for i, value in enumerate(row[attributes]):
                if hypothesis[i] == '?' or hypothesis[i] == value:
                    hypothesis[i] = value
                else:
                    hypothesis[i] = '?'
    return hypothesis
file_path = 'training_data.csv'
hypothesis = find_s_algorithm(file_path)
print("\nThe final hypothesis is:", hypothesis)
```

OUTPUT

Training data:

	Outlook	Temperature	Humidity	Windy	PlayTennis
0	Sunny	Hot	High	False	No
1	Sunny	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Rain	Cold	High	False	Yes
4	Rain	Cold	High	True	No
5	Overcast	Hot	High	True	Yes
6	Sunny	Hot	High	False	No

The final hypothesis is: ['Overcast', 'Hot', 'High', '?']

PROGRAM=5

Develop a program to implement k-Nearest Neighbour algorithm to classify the randomly generated 100 values

of x in the range of [0,1]. Perform the following based on dataset generated.

1. Label the first 50 points {x1,……,x50} as follows: if (xi ≤ 0.5), then xi ∈ Class1, else xi ∈ Class1

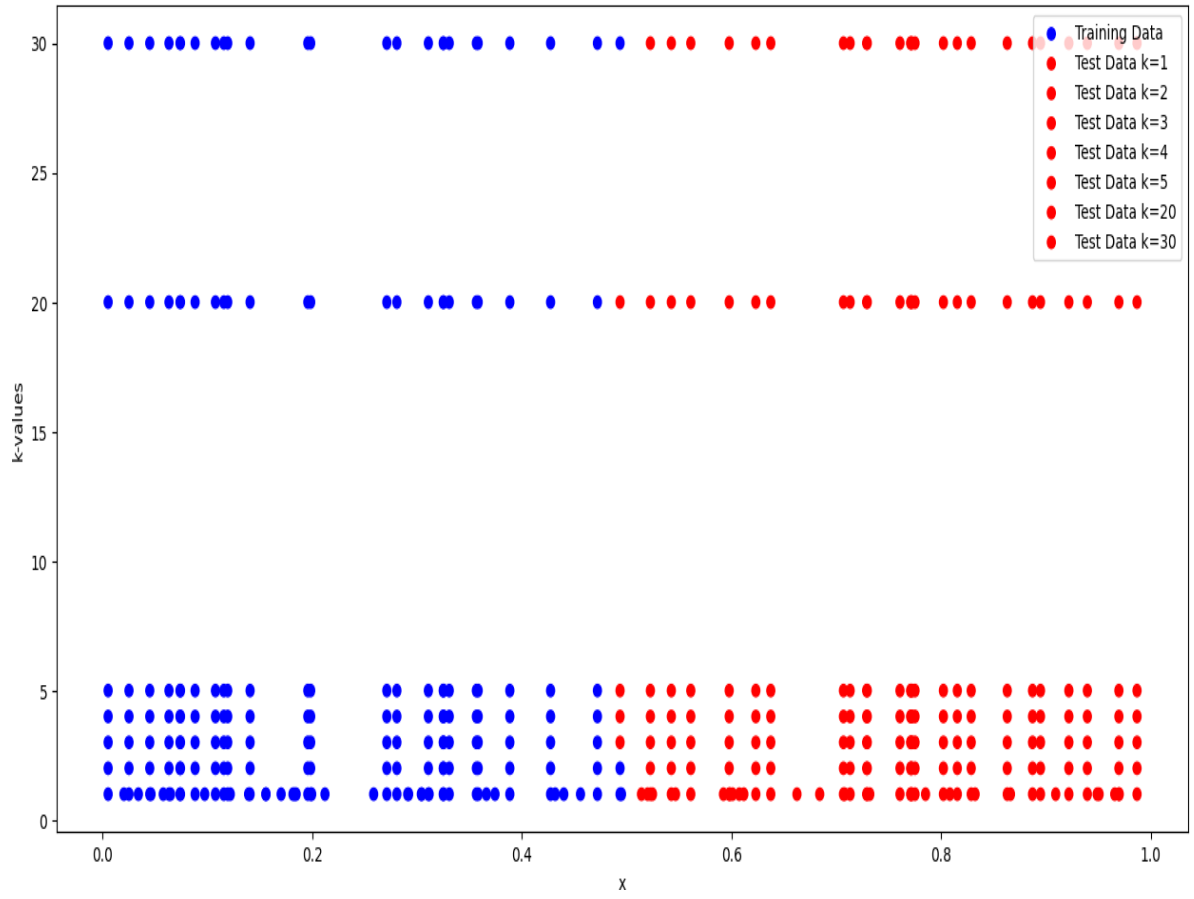
2. Classify the remaining points, x51,……,x100 using KNN. Perform this for k=1,2,3,4,5,20,30

```
import numpy as np
import matplotlib.pyplot as plt
from collections import Counter
def generate_data():
    np.random.seed(42)  # For reproducibility
    x = np.random.rand(100)  # Generate 100 random values in [0,1]
    labels = np.array(["Class1" if xi <= 0.5 else "Class2" for xi in x[:50]]) # Label first 50 points
```

```
    return x, labels


def knn_classification(train_x, train_labels, test_x, k):
    predictions = []
    for x_test in test_x:
        distances = np.abs(train_x - x_test) # Compute absolute distance
        nearest_indices = np.argsort(distances)[:k] # Get k nearest neighbors
        nearest_labels = train_labels[nearest_indices] # Get corresponding labels
        most_common = Counter(nearest_labels).most_common(1)[0][0] # Majority voting
        predictions.append(most_common)
    return np.array(predictions)
def main():
    x, labels = generate_data()
    train_x, test_x = x[:50], x[50:]
    train_labels = labels
        k_values = [1, 2, 3, 4, 5, 20, 30]
        results = {}
    for k in k_values:
        predictions = knn_classification(train_x, train_labels, test_x, k)
        results[k] = predictions
        for k, preds in results.items():
        print(f"Results for k={k}: {preds}")
        plt.scatter(train_x, [1] * 50, c=["blue" if lbl == "Class1" else "red" for lbl in train_labels], label="Training Data")
    for k, preds in results.items():
        plt.scatter(test_x, [k] * 50, c=["blue" if lbl == "Class1" else "red" for lbl in preds], label=f"Test Data k={k}")
```

```
    plt.xlabel("x")
    plt.ylabel("k-values")
    plt.legend()
    plt.show()
if __name__== "__main__":
    main()
```

OUTPUT



Results for k=1: ['Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class1'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class1' 'Class1' 'Class2'

'Class1' 'Class2' 'Class1' 'Class2' 'Class2' 'Class1' 'Class1' 'Class2'

'Class2' 'Class2' 'Class2' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class2' 'Class1'

'Class1' 'Class1']

Results for k=2: ['Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class1'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class1' 'Class1' 'Class2'

'Class1' 'Class2' 'Class1' 'Class2' 'Class2' 'Class1' 'Class1' 'Class2'

'Class2' 'Class2' 'Class2' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class2' 'Class1'

'Class1' 'Class1']

Results for k=3: ['Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class1'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class1' 'Class1' 'Class2'

'Class1' 'Class2' 'Class1' 'Class2' 'Class2' 'Class1' 'Class1' 'Class2'

'Class2' 'Class2' 'Class2' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class1']

Results for k=4: ['Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class1'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class1' 'Class1' 'Class2'

'Class1' 'Class2' 'Class1' 'Class2' 'Class2' 'Class1' 'Class1' 'Class2'

'Class2' 'Class2' 'Class2' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class1']

Results for k=5: ['Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class1'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class1' 'Class1' 'Class2'

'Class1' 'Class2' 'Class1' 'Class2' 'Class2' 'Class1' 'Class1' 'Class2'

'Class2' 'Class2' 'Class2' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class1']

Results for k=20: ['Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class1'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class1' 'Class1' 'Class2'

'Class1' 'Class2' 'Class1' 'Class2' 'Class2' 'Class1' 'Class1' 'Class2'

'Class2' 'Class2' 'Class2' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class1']

Results for k=30: ['Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class1'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class1' 'Class1' 'Class2'

'Class1' 'Class2' 'Class1' 'Class2' 'Class2' 'Class1' 'Class1' 'Class2'

'Class2' 'Class2' 'Class2' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2'

'Class1' 'Class1' 'Class1' 'Class1' 'Class2' 'Class2' 'Class2' 'Class1'

'Class1' 'Class2' 'Class2' 'Class2' 'Class2' 'Class1' 'Class2' 'Class1'

'Class1' 'Class1']

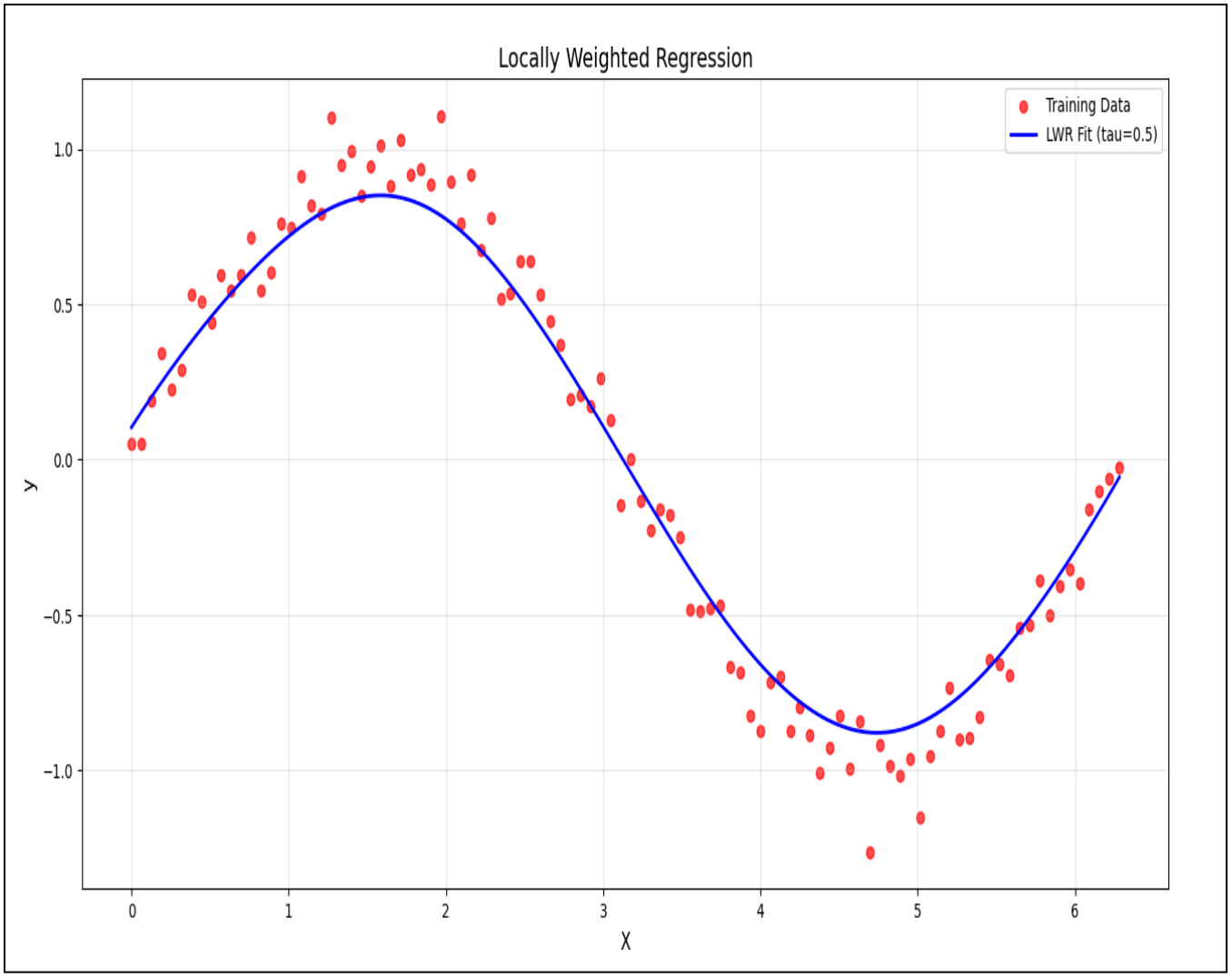
PROGRAM -6

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 gaussian_kernel(x, xi, tau):
    return np.exp(-np.sum((x - xi) ** 2) / (2 * tau ** 2))
def locally_weighted_regression(x, X, y, tau):
    m = X.shape[0]
    weights = np.array([gaussian_kernel(x, X[i], tau) for i in range(m)])
    W = np.diag(weights)
    X_transpose_W = X.T @ W
    theta = np.linalg.inv(X_transpose_W @ X) @ X_transpose_W @ y
    return x @ theta
np.random.seed(42)
X = np.linspace(0, 2 * np.pi, 100)
y = np.sin(X) + 0.1 * np.random.randn(100)
X_bias = np.c_[np.ones(X.shape), X]
x_test = np.linspace(0, 2 * np.pi, 200)
x_test_bias = np.c_[np.ones(x_test.shape), x_test]
tau = 0.5
y_pred = np.array([locally_weighted_regression(xi, X_bias, y, tau) for xi in
x_test_bias])
plt.figure(figsize=(10, 6))
plt.scatter(X, y, color='red', label='Training Data', alpha=0.7)
plt.plot(x_test, y_pred, color='blue', label=f'LWR Fit (tau={tau})', linewidth=2)
plt.xlabel('X', fontsize=12)
```

```
plt.ylabel('y', fontsize=12)
plt.title('Locally Weighted Regression', fontsize=14)
plt.legend(fontsize=10)
plt.grid(alpha=0.3)
plt.show()
```

OUTPUT



PROGRAM -7

Develop a program to demonstrate the working of Linear Regression and Polynomial Regression. Use Boston Housing Dataset for Linear Regression and Auto MPG Dataset (for vehicle fuel efficiency prediction) for Polynomial Regression.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures, StandardScaler
from sklearn.pipeline import make_pipeline
from sklearn.metrics import mean_squared_error, r2_score


def linear_regression_california():
    housing = fetch_california_housing(as_frame=True)
    X = housing.data[["AveRooms"]]
    y = housing.target


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


    model = LinearRegression()
    model.fit(X_train, y_train)


    y_pred = model.predict(X_test)


    plt.scatter(X_test, y_test, color="blue", label="Actual")
```

```
    plt.plot(X_test, y_pred, color="red", label="Predicted")
    plt.xlabel("Average number of rooms (AveRooms)")
    plt.ylabel("Median value of homes ($100,000)")
    plt.title("Linear Regression - California Housing Dataset")
    plt.legend()
    plt.show()

    print("Linear Regression - California Housing Dataset")
    print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
    print("R^2 Score:", r2_score(y_test, y_pred))


def polynomial_regression_auto_mpg():
    url = "https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data"
    column_names = ["mpg", "cylinders", "displacement", "horsepower", "weight", "acceleration", "model_year", "origin"]
    data = pd.read_csv(url, sep='\s+', names=column_names, na_values="?")
    data = data.dropna()

    X = data["displacement"].values.reshape(-1, 1)
    y = data["mpg"].values

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

    poly_model = make_pipeline(PolynomialFeatures(degree=2), StandardScaler(), LinearRegression())
    poly_model.fit(X_train, y_train)
```

```
    y_pred = poly_model.predict(X_test)

    plt.scatter(X_test, y_test, color="blue", label="Actual")
    plt.scatter(X_test, y_pred, color="red", label="Predicted")
    plt.xlabel("Displacement")
    plt.ylabel("Miles per gallon (mpg)")
    plt.title("Polynomial Regression - Auto MPG Dataset")
    plt.legend()
    plt.show()

    print("Polynomial Regression - Auto MPG Dataset")
    print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
    print("R^2 Score:", r2_score(y_test, y_pred))

if __name__== "__main__":
    print("Demonstrating Linear Regression and Polynomial Regression\n")
    linear_regression_california()
    polynomial_regression_auto_mpg()
```

OUTPUT

Demonstrating Linear Regression and Polynomial Regression

Linear Regression - California Housing Dataset

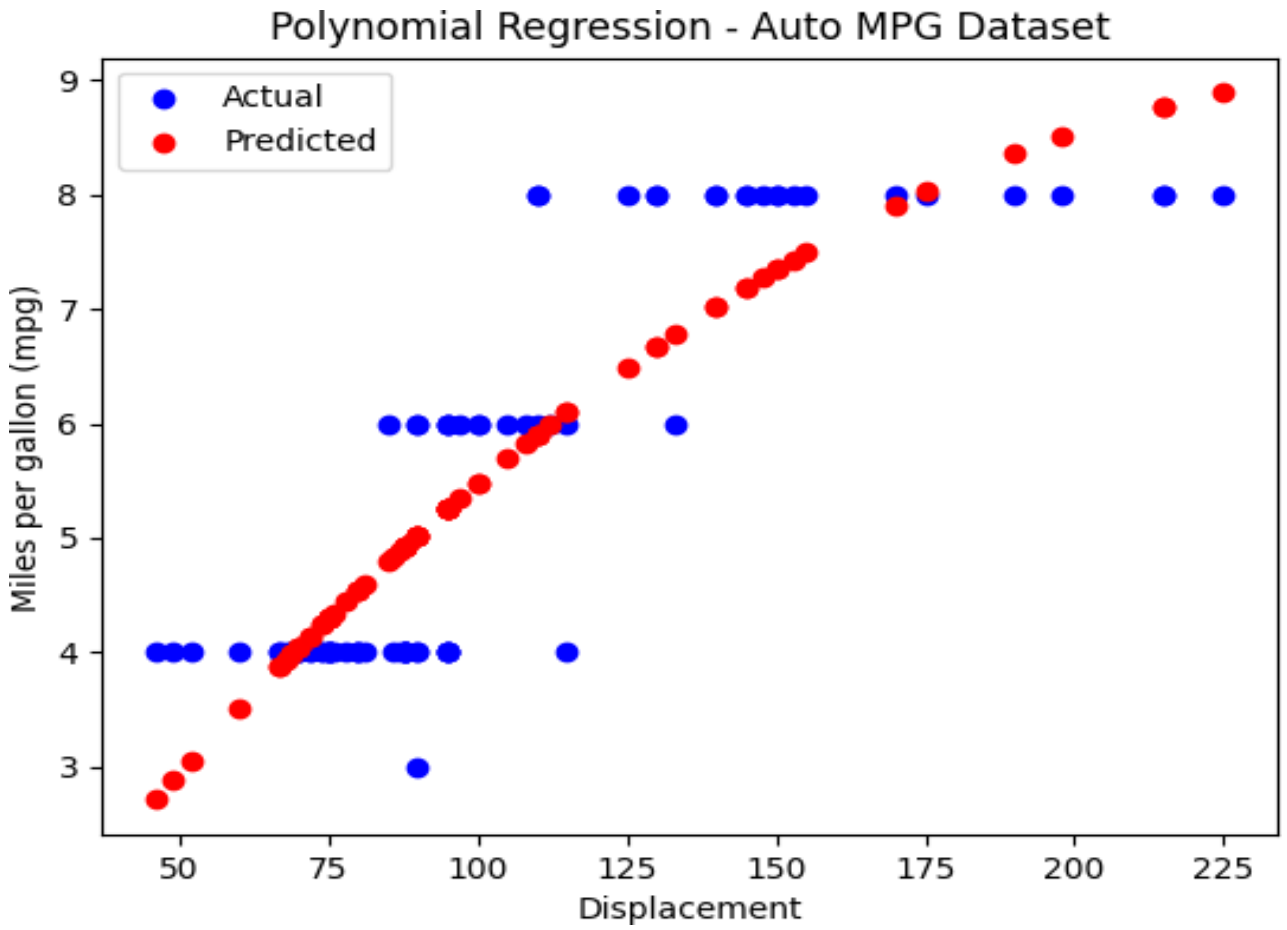
Mean Squared Error: 1.2923314440807299

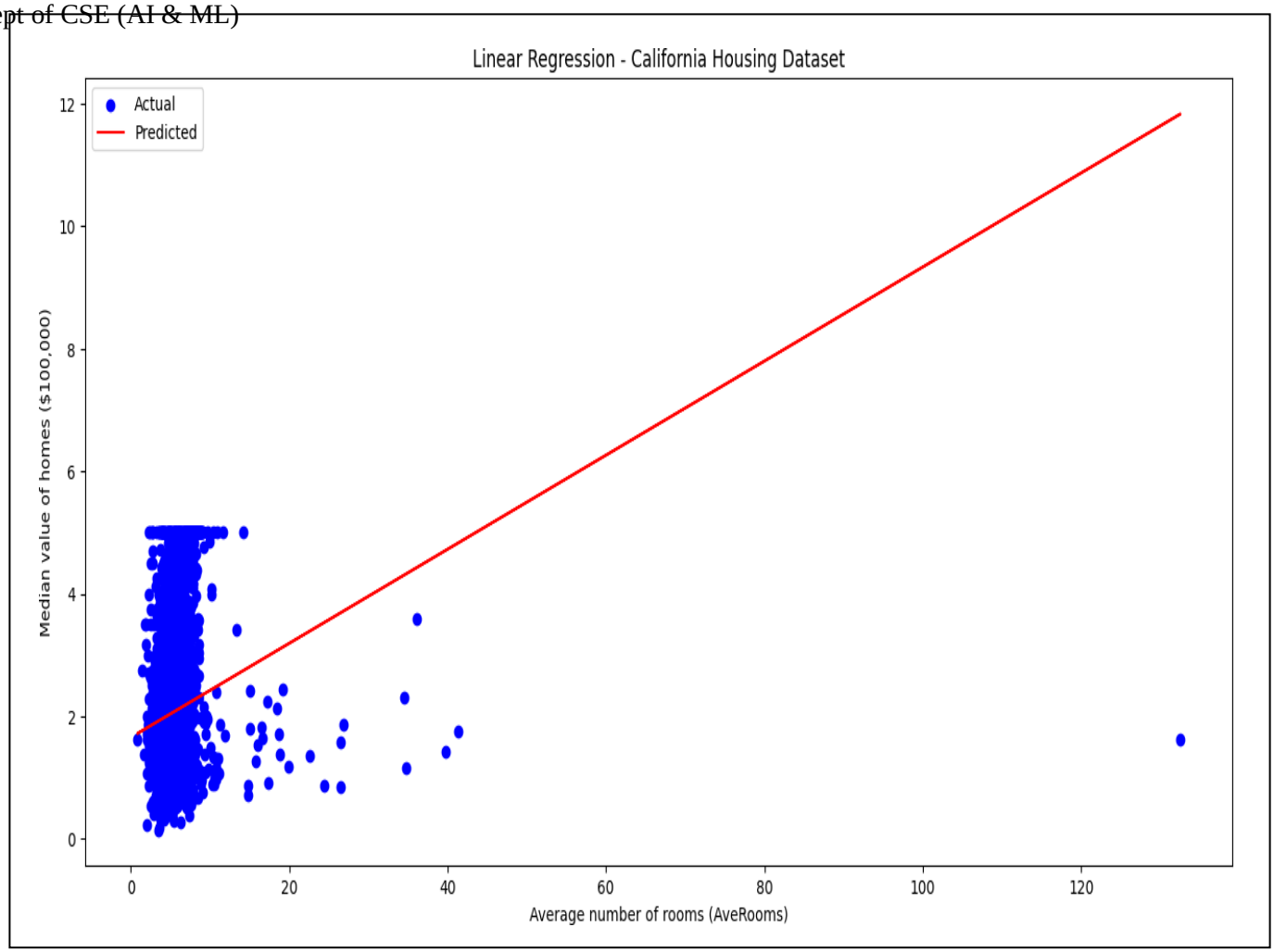
R^2 Score: 0.013795337532284901

Polynomial Regression - Auto MPG Dataset

Mean Squared Error: 0.743149055720586

R^2 Score: 0.7505650609469626




Linear Regression - California Housing Dataset
Actual
Predicted
Median value of homes ($100,000)
Average number of rooms (AveRooms)
0
2
4
6
8
10
12
0
20
40
60
80
100
120

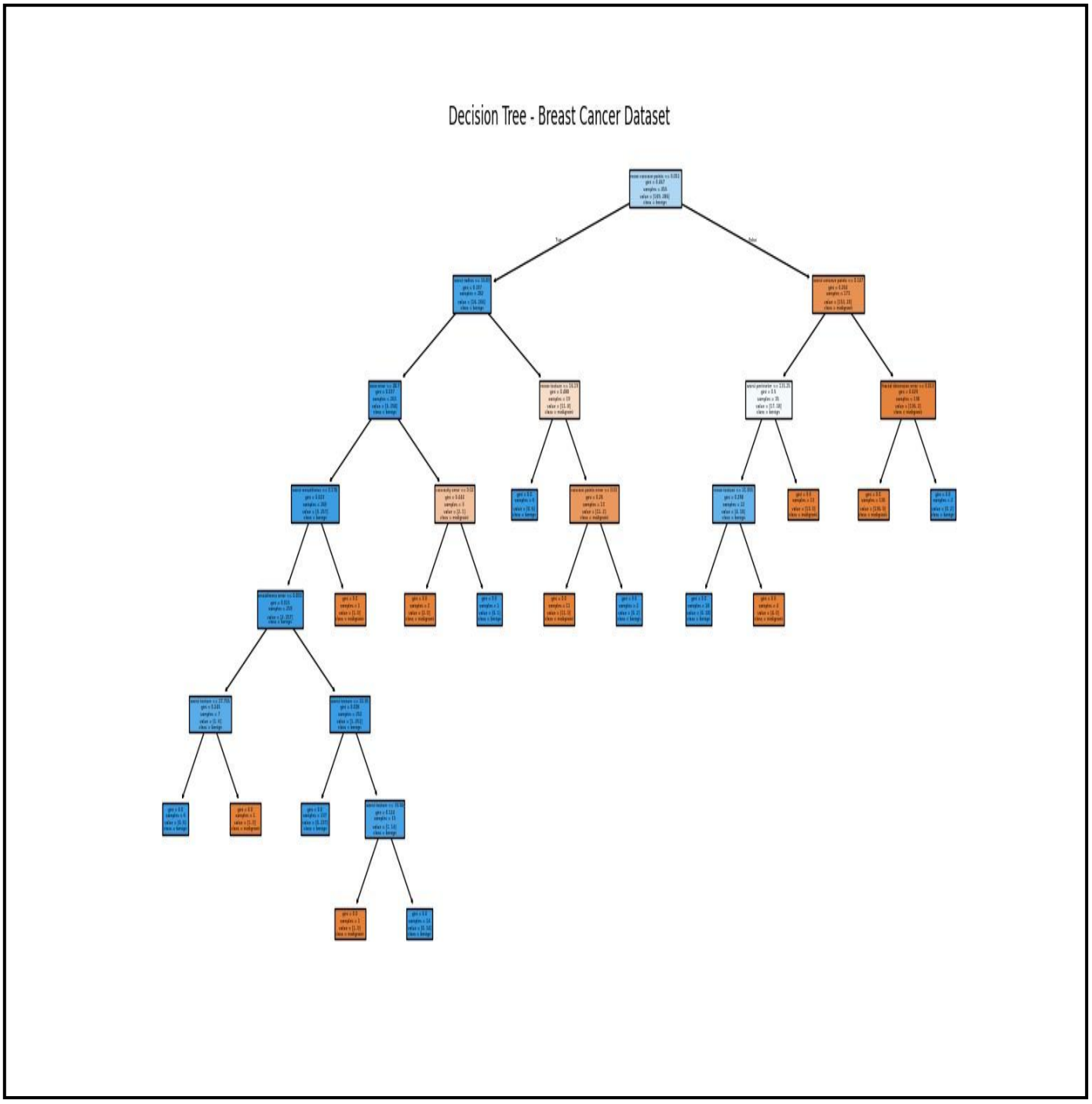
PROGRAM 8

Develop a program to demonstrate the working of the decision tree algorithm. Use Breast Cancer Data set for building the decision tree and apply this knowledge to classify a new sample.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn import tree
data = load_breast_cancer()
X = data.data
y = data.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy * 100:.2f}%")
new_sample = np.array([X_test[0]])
prediction = clf.predict(new_sample)
prediction_class = "Benign" if prediction == 1 else "Malignant"
print(f"Predicted Class for the new sample: {prediction_class}")
plt.figure(figsize=(12,8))
tree.plot_tree(clf, filled=True, feature_names=data.feature_names,
class_names=data.target_names)
```

```
plt.title("Decision Tree - Breast Cancer Dataset")
plt.show()
```

OUTPUT



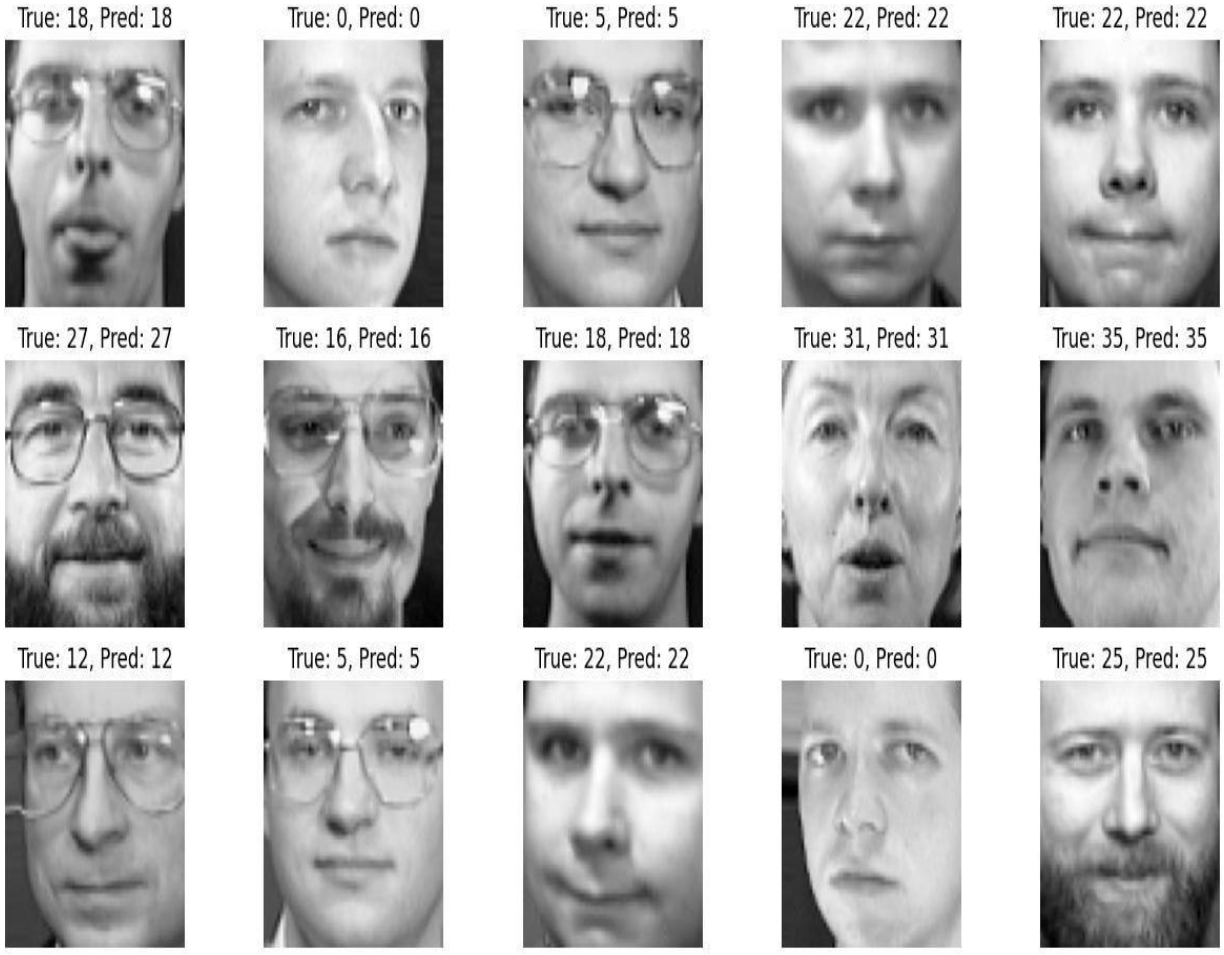
PROGRAM 9

Develop a program to implement the Naive Bayesian classifier considering Olivetti Face Data set for training. Compute the accuracy of the classifier, considering a few test data sets.

```
import numpy as np

from sklearn.datasets import fetch_olivetti_faces

from sklearn.model_selection import train_test_split, cross_val_score

from sklearn.naive_bayes import GaussianNB

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

import matplotlib.pyplot as plt


data = fetch_olivetti_faces(shuffle=True, random_state=42)

X = data.data

y = data.target


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)


gnb = GaussianNB()

gnb.fit(X_train, y_train)

y_pred = gnb.predict(X_test)


accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy * 100:.2f}%')


print("\nClassification Report:")

print(classification_report(y_test, y_pred, zero_division=1))
```

```
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))

cross_val_accuracy = cross_val_score(gnb, X, y, cv=5, scoring='accuracy')
print(f'\nCross-validation accuracy: {cross_val_accuracy.mean() * 100:.2f}%')

fig, axes = plt.subplots(3, 5, figsize=(12, 8))
for ax, image, label, prediction in zip(axes.ravel(), X_test, y_test, y_pred):
    ax.imshow(image.reshape(64, 64), cmap=plt.cm.gray)
    ax.set_title(f"True: {label}, Pred: {prediction}")
    ax.axis('off')
plt.show()
```

OUTPUT



Accuracy: 80.83%

Classification Report:

	precision	recall	f1-score	support
0	0.67	1.00	0.80	2
1	1.00	1.00	1.00	2
2	0.33	0.67	0.44	3
3	1.00	0.00	0.00	5
4	1.00	0.50	0.67	4
5	1.00	1.00	1.00	2
7	1.00	0.75	0.86	4
8	1.00	0.67	0.80	3
9	1.00	0.75	0.86	4
10	1.00	1.00	1.00	3
11	1.00	1.00	1.00	1
12	0.40	1.00	0.57	4
13	1.00	0.80	0.89	5
14	1.00	0.40	0.57	5
15	0.67	1.00	0.80	2
16	1.00	0.67	0.80	3
17	1.00	1.00	1.00	3
18	1.00	1.00	1.00	3
19	0.67	1.00	0.80	2
20	1.00	1.00	1.00	3
21	1.00	0.67	0.80	3
22	1.00	0.60	0.75	5
23	1.00	0.75	0.86	4

24	1.00	1.00	1.00	3
25	1.00	0.75	0.86	4
26	1.00	1.00	1.00	2
27	1.00	1.00	1.00	5
28	0.50	1.00	0.67	2
29	1.00	1.00	1.00	2
30	1.00	1.00	1.00	2
31	1.00	0.75	0.86	4
32	1.00	1.00	1.00	2
34	0.25	1.00	0.40	1
35	1.00	1.00	1.00	5
36	1.00	1.00	1.00	3
37	1.00	1.00	1.00	1
38	1.00	0.75	0.86	4
39	0.50	1.00	0.67	5
accuracy			0.81	120
macro avg	0.89	0.85	0.83	120
weighted avg	0.91	0.81	0.81	120

Confusion Matrix:

```
[[2 0 0 ... 0 0 0]
 [0 2 0 ... 0 0 0]
 [0 0 2 ... 0 0 1]
 ...
 [0 0 0 ... 1 0 0]
 [0 0 0 ... 0 3 0]
 [0 0 0 ... 0 0 5]]
```

Cross-validation accuracy: 87.25%

PROGRAM 10

Develop a program to implement k-means clustering using Wisconsin Breast Cancer data set and visualize the clustering result.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_breast_cancer
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import confusion_matrix, classification_report

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

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

kmeans = KMeans(n_clusters=2, random_state=42)
y_kmeans = kmeans.fit_predict(X_scaled)

print("Confusion Matrix:")
print(confusion_matrix(y, y_kmeans))
print("\nClassification Report:")
print(classification_report(y, y_kmeans))
```

```
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

df = pd.DataFrame(X_pca, columns=['PC1', 'PC2'])
df['Cluster'] = y_kmeans
df['True Label'] = y

plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='PC1', y='PC2', hue='Cluster', palette='Set1', s=100, edgecolor='black', alpha=0.7)
plt.title('K-Means Clustering of Breast Cancer Dataset')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend(title="Cluster")
plt.show()

plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='PC1', y='PC2', hue='True Label', palette='coolwarm', s=100, edgecolor='black', alpha=0.7)
plt.title('True Labels of Breast Cancer Dataset')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend(title="True Label")
plt.show()

plt.figure(figsize=(8, 6))
```

```
sns.scatterplot(data=df, x='PC1', y='PC2', hue='Cluster', palette='Set1', s=100,
edgecolor='black', alpha=0.7)

centers = pca.transform(kmeans.cluster_centers_)

plt.scatter(centers[:, 0], centers[:, 1], s=200, c='red', marker='X',
label='Centroids')

plt.title('K-Means Clustering with Centroids')

plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')

plt.legend(title="Cluster")

plt.show()
```

OUTPUT

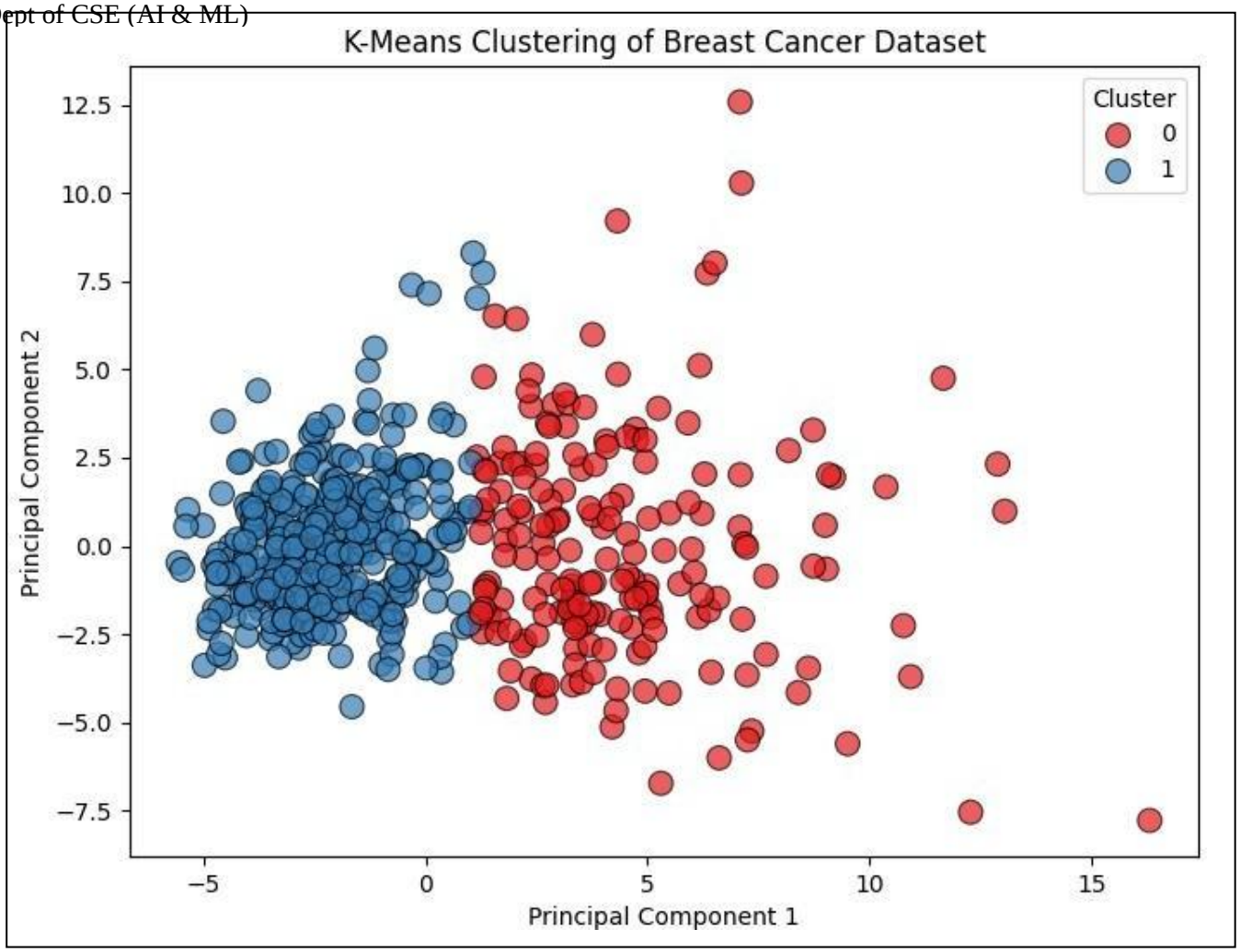
Confusion Matrix:

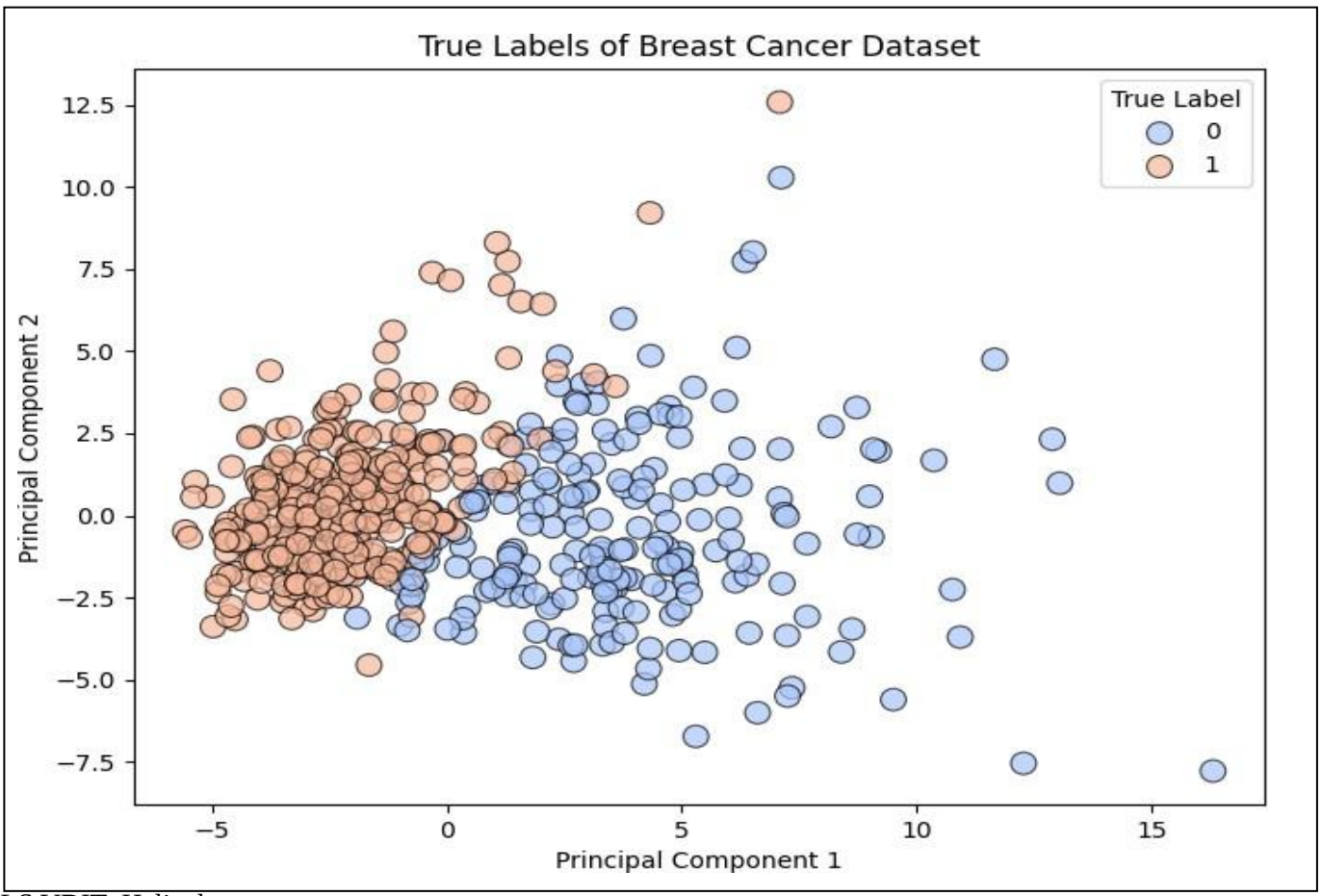
[[175 37]

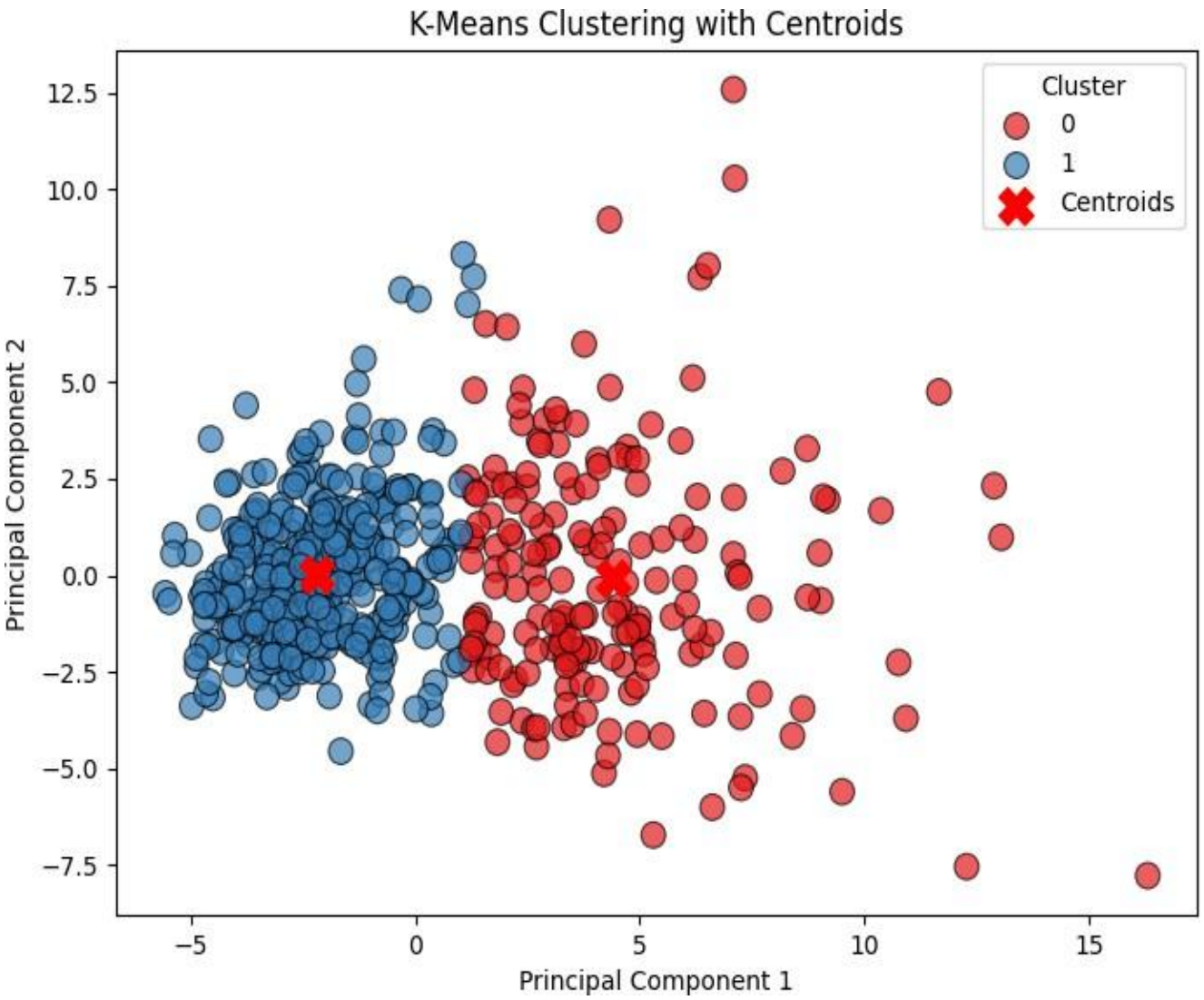
[13 344]]

Classification Report:

	precision	recall	f1-score	support
0	**0.93**	**0.83**	**0.88**	**212**
1	**0.90**	**0.96**	**0.93**	**357**
accuracy			**0.91**	**569**
macro avg	**0.92**	**0.89**	**0.90**	**569**
weighted avg	**0.91**	**0.91**	**0.91**	**569**


K-Means Clustering of Breast Cancer Dataset
Cluster
0
1
Principal Component 2
Principal Component 1
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−5
0
5
10
15


True Labels of Breast Cancer Dataset
True Label
0
1
Principal Component 2
Principal Component 1
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−5
0
5
10
15


K-Means Clustering with Centroids
Cluster
0
1
Centroids
Principal Component 2
Principal Component 1
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−5
0
5
10
15

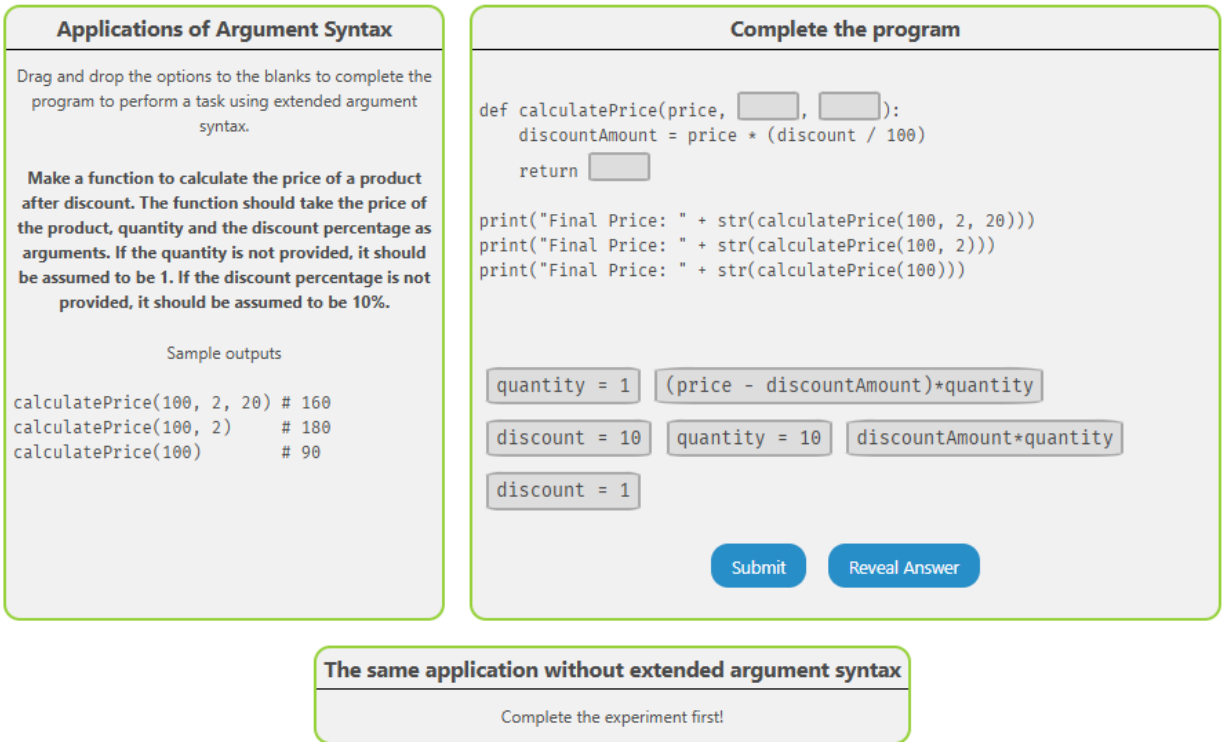
Program 11:

To learn about extended argument syntax and how we can use it to make python code much simpler

Procedure:

There are two tabs in this experiment -

1. Learn - This tab aims at testing your coding skills with extended argument syntax and aims at teaching you about the same. You have to drag and drop the correct code in the given space and you will be shown code which does the same thing but without using argument syntax.
2. Practice - This tab shows you how argument syntax works through an interactive animation. Just fill in the blanks with correct input and you will be shown an animation according to that.

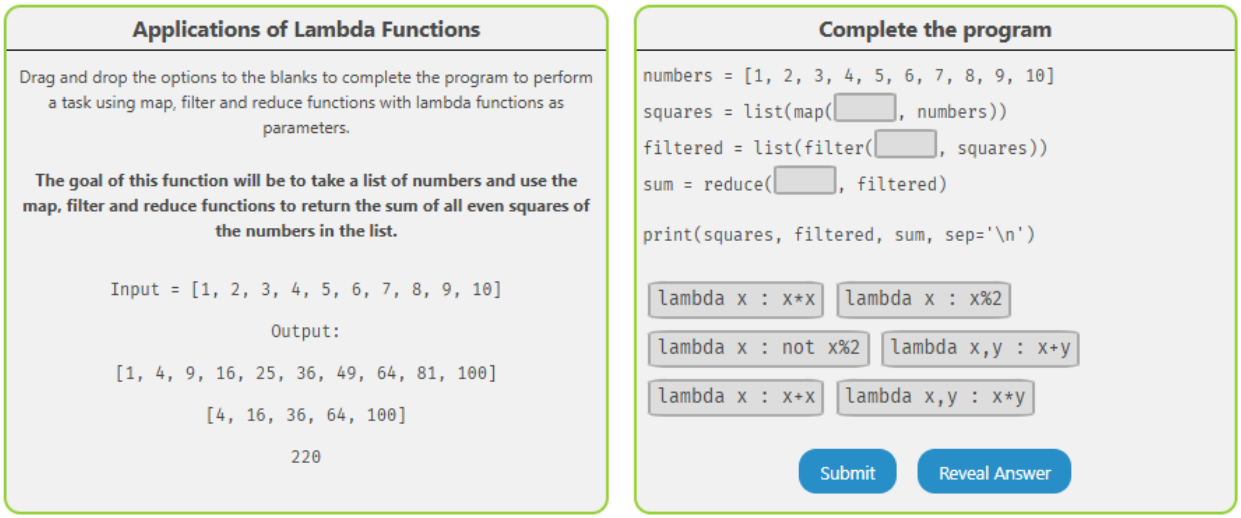


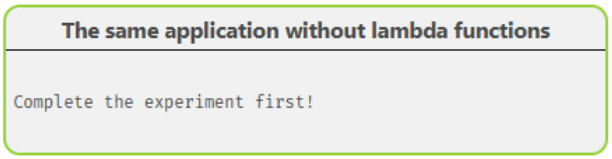
Program 12

This experiment aims at learning the concept of lambda functions, a very useful concept taken from functional programming.

Procedure:

- To perform this experiment, you just need to observe how different lambda functions works on different inputs.
- You can change the input values and see how the output changes. Select a value from the dropdown to change the lambda function as well





KLS

Vishwanathrao Deshpande Institute of Technology

Haliyal – 581329

Microcontrollers & Embedded Systems [BCO601]

for

VI Semester B.E.

As prescribed by

VISVESVARAYA TECHNOLOGICAL UNIVERSITY,BELAGAVI–590014

(For Academic Year: 2025-2026)

Prepared by

Prof. Kanchan Khokale

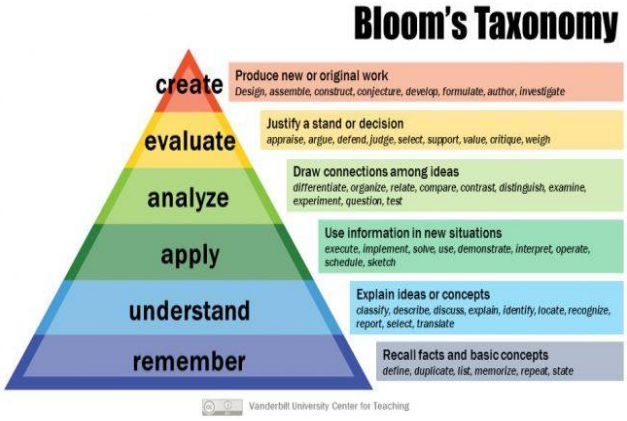
Department of Computer Science & Engineering(AIML)

Index

PROGRAM OUTCOMES(POs)

Program Outcomes as defined by NBA (PO) Engineering Graduates will be able to:

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.



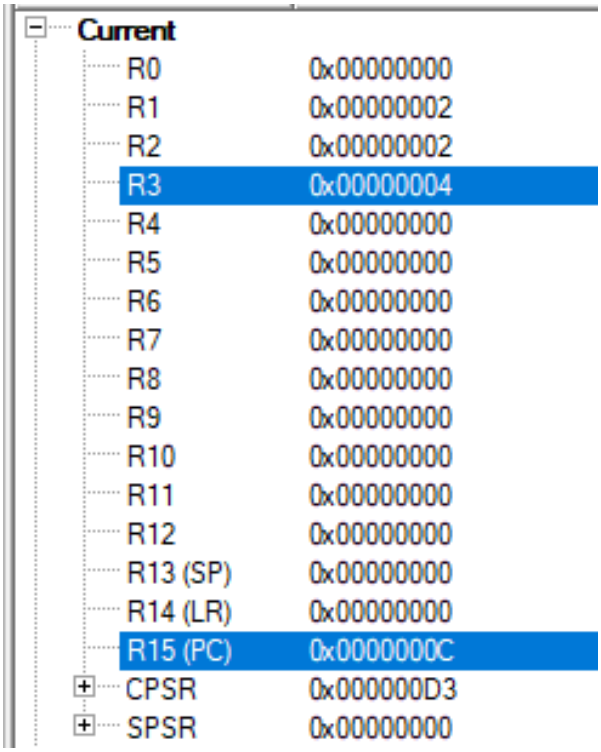
Vision (College)
To nurture talent and enrich society through excellence in technical education, research and innovation.
Mission (College)
1. To augment innovative Pedagogy, kindle quest for interdisciplinary learning & to enhance conceptual understanding. 2. To build competence, professional ethics & develop entrepreneurial thinking. 3. To strengthen Industry Institute Partnership & explore global collaborations. 4. To inculcate culture of socially responsible citizenship. 5. To focus on Holistic & Sustainable development.
Vision (Dept)
To lead the way in the creation and application of cutting-edge Computer science and engineering (AIML) technologies, advancing the frontiers of knowledge, and empowering future generations to drive innovation and transformation on a global scale.
Mission (Dept)
To train students with a strong conceptual understanding using innovative pedagogies, empowering them to excel in the dynamic fields of Artificial Intelligence and Machine Learning. To imbibe professional, research, and entrepreneurial skills with a commitment to the nation's development at large. To strengthen the industry-institute Interaction. To promote life–long learning with a sense of societal & ethical responsibilities.

Program Educational Objectives (PEO)	
PEO1	To develop an ability to identify and analyze the requirements of Computer Science and Engineering in design and providing novel engineering solutions.
PEO2	To develop abilities to work in team on multidisciplinary projects with effective communication skills, ethical qualities and leadership roles.

PEO3	To develop abilities for successful Computer Science Engineer and achieve higher career goals.
Program Specific Outcomes (PSO)	
PSO 1	To develop abilities to model real world problems using appropriate algorithms, computational theories and programming languages in the area of AI and ML..
PSO 2	To develop software applications and products in specialized areas of Artificial Intelligence and Machine Learning.

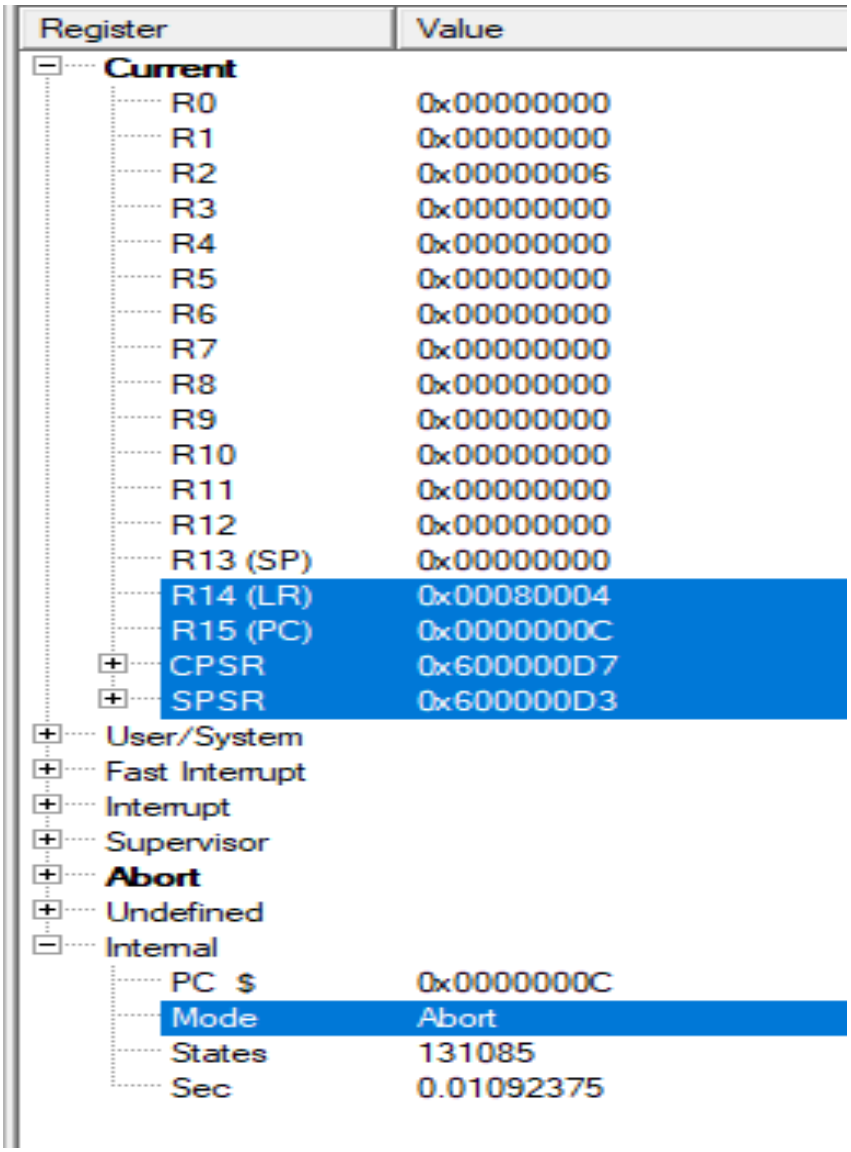
1. WRITE AN ALP TO MULTIPLY TWO 16 BIT NUMBERS

```
AREA MULTIPLY, CODE, READONLY
ENTRY
START
MOV R1,#2          ;MOVE 20 VALUE TO R1 REGISTER
MOV R2,#2          ; MOVE 20 VALUE TO R2 REGISTER
MUL R3,R1,R2       ;MULTIPLY R1 & R2 AND STORE RESULT IN R3 EGISTER
NOP
NOP
NOP
END
```



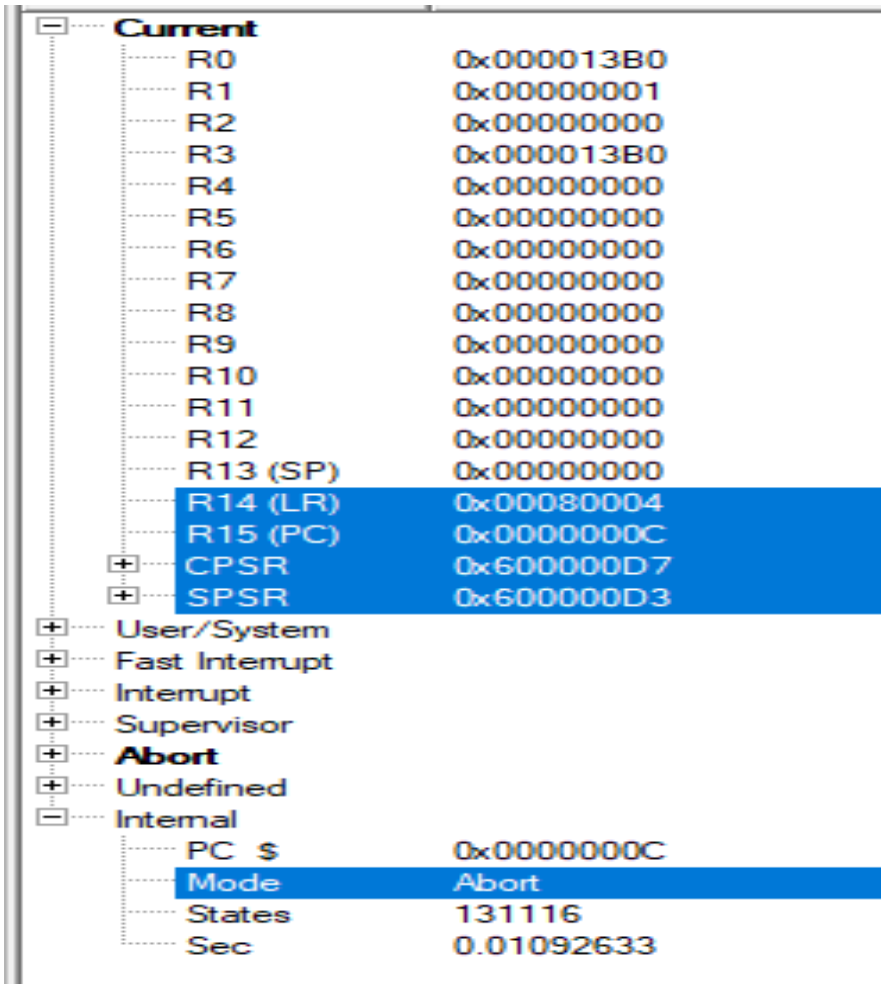
2. WRITE AN ALP TO FIND THE SUM OF FIRST 10 INTEGER NUMBERS

```
        AREA SUM, CODE, READONLY
        ENTRY
        MOV R1,#10          ; LOAD 10 TO REGISTER
        MOV R2,#0           ; EMPTY THE REGISTER TO STORE RESULT
LOOP ADD R2, R2, R1         ; ADD THE CONTENT OF R1 WITH RESULT AT R2
        SUBS R1,#0x01       ; DECREMENT R1 BY 1
        BNE LOOP            ; REPEAT TILL R1 GOES 0
        NOP                 ; JUMPS BACK TO C CODE
        NOP
        END
```


Register
Value
Current
R0 0x00000000
R1 0x00000000
R2 0x00000006
R3 0x00000000
R4 0x00000000
R5 0x00000000
R6 0x00000000
R7 0x00000000
R8 0x00000000
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x00000000
R14 (LR) 0x00080004
R15 (PC) 0x0000000C
CPSR 0x600000D7
SPSR 0x600000D3
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
PC $ 0x0000000C
Mode Abort
States 131085
Sec 0.01092375

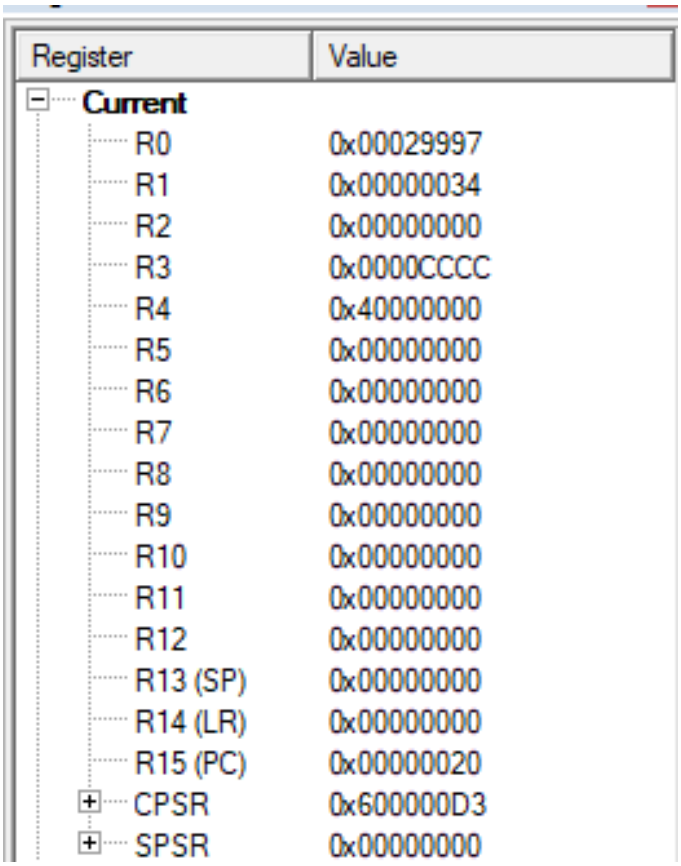
3. WRITE A PROGRAM TO FIND FACTORIAL OF A NUMBER.

```
        AREA FACTORIAL, CODE, READONLY
        ENTRY           ; MARK FIRST INSTRUCTION TO EXECUTE START
        MOV R0, #3      ; STORE FACTORIAL NUMBER IN R0
        MOV R1, R0      ; MOVE THE SAME NUMBER IN R1
FACT SUBS R1, R1, #1    ; SUBTRACTION
        MUL R3, R0,R1           ; MULTIPLICATION
        MOV R0, R3      ; RESULT R3 MOVED TO R0
        CMP R1, #1      ; COMPARISON R1 WITH 1
        BNE FACT        ; BRANCH TO THE FACT IF NOT EQUAL
        NOP
        NOP
        NOP
        END             ; MARK END OF FILE
```


Current
R0 0x000013B0
R1 0x00000001
R2 0x00000000
R3 0x000013B0
R4 0x00000000
R5 0x00000000
R6 0x00000000
R7 0x00000000
R8 0x00000000
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x00000000
R14 (LR) 0x00080004
R15 (PC) 0x0000000C
CPSR 0x600000D7
SPSR 0x600000D3
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
PC $ 0x0000000C
Mode Abort
States 131116
Sec 0.01092633

4. WRITE A PROGRAM TO ADD AN ARRAY OF 16 BIT NUMBERS AND STORE THE 32 BIT RESULT IN INTERNAL RAM.

```
AREA ADDITION, CODE, READONLY

ENTRY

MOV     R5, #6          ; Initialize counter (N=6)

MOV     R0, #0          ; Initialize sum to zero

LDR     R1, =VALUE1     ; Load address of first value

LDR     R4, =RESULT     ; Load address to store result

LOOP    LDRH    R3, [R1], #2    ; Load 16-bit data and increment R1 by 2

ADD     R0, R0, R3      ; Add R3 to sum (R0)

SUBS    R5, R5, #1      ; Decrement counter

BNE     LOOP            ; Repeat until counter is zero

STR     R0, [R4]        ; Store 32-bit sum in RESULT

B       .               ; Infinite loop to stop execution


VALUE1  DCW     0x1111, 0x2222, 0x3333, 0xAAAA, 0xBBBB, 0xCCCC  ; 16-bit values array


AREA    DATA2, DATA, READWRITE  ; Data section for result storage

RESULT  DCD     0x0             ; 32-bit space for result


END
```

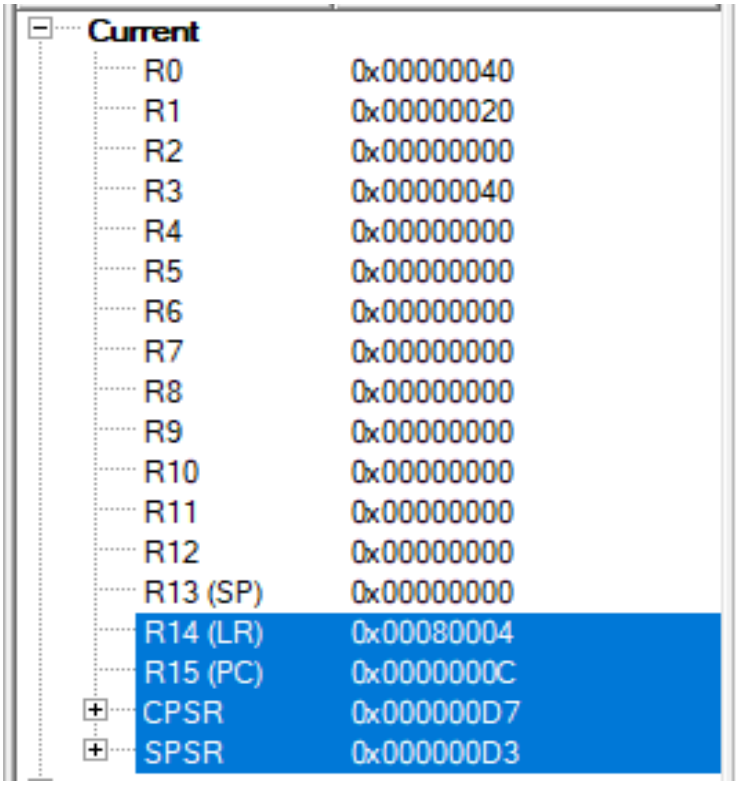


Register	Value
Current	
R0	0x00029997
R1	0x00000034
R2	0x00000000
R3	0x0000CCCC
R4	0x40000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000020
CPSR	0x600000D3
SPSR	0x00000000

5. WRITE A PROGRAM TO FIND THE SQUARE OF A NUMBER (1 TO 10) USING LOOK-UP TABLE.

```
        AREA SQUARE, CODE, READONLY
        ENTRY   ; Mark first instruction to execute
START   LDR     R0, =TABLE1     ; Load start address of lookup table
      LDR     R1, =8          ; Load number whose square is to be found
      MOV     R1, R1, LSL #2  ; Generate address corresponding to square of given number
      ADD     R0, R0, R1      ; Load address of element in lookup table
      LDR     R3, [R0]        ; Get square of given number in R3
      NOP
      NOP
      NOP

; Lookup table contains squares of numbers from 0 to 10 (in hex)
TABLE1  DCD     0x00000000  ; Square of 0 = 0
      DCD     0x00000001  ; Square of 1 = 1
      DCD     0x00000004  ; Square of 2 = 4
      DCD     0x00000009  ; Square of 3 = 9
      DCD     0x00000010  ; Square of 4 = 16
      DCD     0x00000019  ; Square of 5 = 25
      DCD     0x00000024  ; Square of 6 = 36
      DCD     0x00000031  ; Square of 7 = 49
```

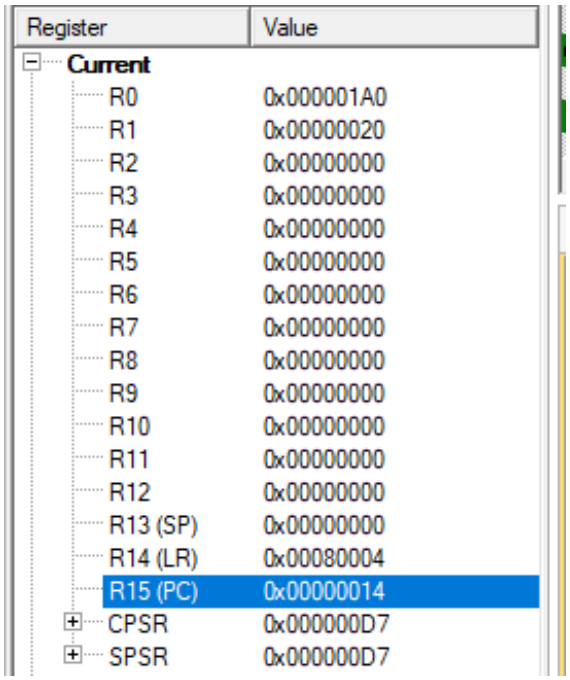
```
DCD    0x00000040  ; Square of 8 = 64

DCD    0x00000051  ; Square of 9 = 81

DCD    0x00000064  ; Square of 10 = 100


END    ; Mark end of file
```



6. WRITE A PROGRAM TO FIND THE LARGEST/SMALLEST NUMBER IN AN ARRAY OF 32 NUMBERS.

```
AREA SMALLEST, CODE, READONLY
    ENTRY  ; Mark first instruction to execute
START  MOV    R5, #6        ; Initialize counter to 6 (N=7 total elements)
    LDR    R1, =VALUE1    ; Load the address of the first value
    LDR    R2, [R1], #4   ; Load the first element into R2
LOOP   LDR    R4, [R1], #4   ; Load next array element
    CMP    R2, R4         ; Compare R2 with R4
    BLS    LOOP1          ; If R2 <= R4, continue
    MOV    R2, R4         ; If R4 < R2, update R2 to hold the smaller value
LOOP1  SUBS   R5, R5, #1     ; Decrement counter
    CMP    R5, #0         ; Compare counter to 0
    BNE    LOOP           ; Loop back until all elements are checked
    LDR    R4, =RESULT    ; Load the address of RESULT
    STR    R2, [R4]       ; Store the smallest value in RESULT
    NOP
    NOP
    NOP
; Array of 32-bit numbers (N=7)
VALUE1  DCD    0x44444444
    DCD    0x22222222
    DCD    0x11111111
```

```
    DCD    0x33333333

    DCD    0xAAAAAAAA

    DCD    0x88888888

    DCD    0x99999999


    AREA DATA2, DATA, READWRITE   ; To store the result

RESULT  DCD    0x0              ; 32-bit space for result

    END    ; Mark end of file
```



7. DISPLAY "HELLO WORLD" MESSAGE USING INTERNAL UART.

```
#include <LPC213X.H>

#include <stdint.h>


void UART0_init(void)

{

   PINSEL0 |= 0x00000005;  /* Enable UART0 RxD0 and TxD0 pins */


   U0LCR = 0x83;  /* DLAB = 1, 1 stop bit, 8-bit character length */

   U0DLM = 0x00;

   U0DLL = 0x61;  /* Baud rate = 9600 with PCLK = 15MHz */

   U0LCR = 0x03;  /* DLAB = 0 */

}


unsigned char UART0_RxChar(void)  /* Function to receive a byte on UART0 */

{

   while ((U0LSR & 0x01) == 0);  /* Wait till RDR bit becomes 1 (valid data) */

   return U0RBR;

}
```

```
void UART0_TxChar(char ch)                    /* Function to send a byte on UART0 */
{
   U0THR = ch;
   while ((U0LSR & 0x20) == 0);  /* Wait until THRE bit becomes 1 (transmission complete) */
}
void UART0_SendString(char *p)  /* Function to send a string via UART0 */
{
   while (*p != '\0')
   {
      UART0_TxChar(*p);
      p++;
   }
}
int main(void)
{
   UART0_init();  /* Initialize UART0 */

   UART0_SendString("Hello World\r\n");  /* Send "Hello World" via UART */

   while (1);  /* Infinite loop */
}
```

Using Keil uVision

Create a new project in Keil uVision.

Add this .c file and select LPC213X as the target.

Compile and build the project.

Load it onto an LPC213X development board.

Open a Serial Monitor (e.g., Tera Term, PuTTY, or Keil Debugger) with 9600 baud rate.

Observe the output:

Copy

Edit

Output:

Hello World

8. Interface a Stepper motor and rotate it in clockwise and anti-clockwise direction

```
#include <LPC21xx.H>

void clock_wise(void);

void anti_clock_wise(void);

unsigned long int var1, var2;

unsigned int i = 0, j = 0, k = 0;

int main(void)

{

   PINSEL0 = 0x00FFFFFF;  // Set P0.12 to P0.15 as GPIO

   IO0DIR |= 0x0000F000;  // Set P0.12 to P0.15 as output

   while (1)

   {

      // Rotate Clockwise 50 times

      for (j = 0; j < 50; j++)

      {

         clock_wise();

      }

      // Delay

      for (k = 0; k < 65000; k++);

      // Rotate Anti-Clockwise 50 times

      for (j = 0; j < 50; j++)

      {
```

```
            anti_clock_wise();
        }
        // Delay
        for (k = 0; k < 65000; k++);
    }
} // End of main
void clock_wise(void)
{
    var1 = 0x00001000;  // Initialize first step
    for (i = 0; i < 4; i++)  // Perform 4 steps (A → B → C → D)
    {
        IO0CLR = 0x0000F000;  // Clear P0.12 - P0.15
        IO0SET = var1;  // Set next step
        for (k = 0; k < 3000; k++);  // Delay for speed control
        var1 <<= 1;  // Shift left for next step
        if (var1 > 0x00008000) var1 = 0x00001000;  // Reset after 4 steps
    }
}
void anti_clock_wise(void)
{
    var1 = 0x00008000;  // Initialize last step
    for (i = 0; i < 4; i++)  // Perform 4 steps (D → C → B → A)
```

```
  {
    IO0CLR = 0x0000F000;  // Clear P0.12 - P0.15
    IO0SET = var1;  // Set next step
    for (k = 0; k < 3000; k++);  // Delay for speed control
    var1 >>= 1;  // Shift right for next step
    if (var1 < 0x00001000) var1 = 0x00008000;  // Reset after 4 steps
  }
}
```

- **Create a new Keil project** and select **LPC21xx** as the target device.
- **Add the above .c file** and compile it.
- **Simulate using Debug Mode** to check GPIO changes.

9. Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between

Port0 Connected to data lines of all 7 segement displays

a = P0.16

b = P0.17

c = P0.18

d = P0.19

e = P0.20

f = P0.21

g = P0.22

dot = P0.23

Select lines for four 7 Segments DIS1 P0.28

DIS2 P0.29 DIS3 P0.30 DIS4 P0.31

Values Correspoinding to Alphabets 1, 2, 3 and 4

```
#include <LPC21XX.h>

unsigned int delay;

unsigned int Switchcount=0;

unsigned int Disp[16]={0x003F0000, 0x00060000, 0x005B0000, 0x004F0000,
0x00660000,0x006D0000, 0x007D0000, 0x00070000, 0x007F0000, 0x006F0000,

0x00770000,0x007C0000,  0x00390000, 0x005E0000, 0x00790000, 0x00710000 };

#define SELDISP1 0x10000000      //P0.28

#define SELDISP2 0x20000000      //P0.29

#define SELDISP3 0x40000000      //P0.30
```

```
#define SELDISP4 0x80000000      //P0.31

#define ALLDISP 0xF0000000       //Select all display

#define DATAPORT 0x00FF0000   //P0.16 to P0.23 Data lines connected to drive Seven Segments

int main (void)

{

    PINSEL0 = 0x00000000; PINSEL1 = 0x00000000; IO0DIR = 0xF0FF0000; IO1DIR = 0x00000000;

    while(1)

    {

    IO0SET |= ALLDISP;    // select all digits

    IO0CLR = 0x00FF0000;          // clear the data lines to 7-segment displays IO0SET =
    Disp[Switchcount];    // get the 7-segment display value from the array

    if(!(IO1PIN & 0x00800000))   // if the key is pressed

    {

            for(delay=0;delay<100000;delay++); // delay

            if((IO1PIN & 0x00800000))  // check to see if key has been released

            {

                    Switchcount++;

                    if(Switchcount == 0x10)        // 0 to F has been displayed ? go back to 0

                    {

                            Switchcount = 0; IO0CLR =0xF0FF0000;

                    }

            } } } }
```

10. Interface a 4x4 keyboard and display the key code on an LCD.

```
/*Program to demonstrate keyboard operation Takes a key from key board and displays it on LCD screen*/

#include<lpc21xx.h> #include<stdio.h>


/******* FUNCTION PROTOTYPE*******/ void lcd_init(void);

void clr_disp(void); void lcd_com(void); void lcd_data(void); void wr_cn(void); void wr_dn(void); void scan(void); void get_key(void); void display(void);

void delay(unsigned int); void init_port(void);


unsigned long int scan_code[16]= {0x00EE0000,0x00ED0000,0x00EB0000,0x00E70000,
0x00DE0000,0x00DD0000,0x00DB0000,0x00D70000,

0x00BE0000,0x00BD0000,0x00BB0000,0x00B70000,

0x007E0000,0x007D0000,0x007B0000,0x00770000};


unsigned char ASCII_CODE[16]= {'0','1','2','3',

'4','5','6','7',

'8','9','A','B',

'C','D','E','F'};

unsigned char row,col;

unsigned char temp,flag,i,result,temp1; unsigned int r,r1;

unsigned long int var,var1,var2,res1,temp2,temp3,temp4; unsigned char *ptr,disp[] = "4X4 KEYPAD";
```

```
unsigned char disp0[] = "KEYPAD TESTING"; unsigned char disp1[] = "KEY = ";

int main()

{

//  ARMLIB_enableIRQ(); init_port();        //port intialisation

delay(3200);          //delay

lcd_init();    //lcd intialisation

delay(3200);          //delay

clr_disp();   //clear display

delay(500); //delay


//.....LCD DISPLAY TEST    //

ptr = disp;

temp1 = 0x81;        // Display starting address lcd_com();

delay(800);


while(*ptr!='\0')

{

temp1 = *ptr; lcd_data(); ptr ++;

}
```

```
//.....KEYPAD Working.    //

while(1)

{

get_key();


display();

}


} //end of main()


void get_key(void)  //get the key from the keyboard

{

unsigned int i; flag = 0x00;

IO1PIN=0x000f0000;

while(1)

{

for(row=0X00;row<0X04;row++)   //Writing one for col's

{

if( row == 0X00)

{

temp3=0x00700000;

}
```

```
else if(row == 0X01)
{ temp3=0x00B00000;
}
else if(row == 0X02)
{
temp3=0x00D00000;
}
else if(row == 0X03)
{
temp3=0x00E00000;
}
var1 = temp3;
IO1PIN = var1;                  // each time var1 value is put to port1 IO1CLR =~var1;

// Once again Conforming (clearing all other bits)
scan();
delay(100); //delay
if(flag == 0xff)
break;
}                                       // end of for if(flag == 0xff) break;
}                                      // end of while
```

```
for(i=0;i<16;i++)
{
if(scan_code[i] == res1)      //equate the scan_code with res1
{
result = ASCII_CODE[i];      //same position value of ascii code break;     //is assigned to result
}
}
}// end of get_key();
void scan(void)
{
unsigned long int t;
temp2 = IO1PIN;     // status of port1
temp2 = temp2 & 0x000F0000;                 // Verifying column key if(temp2 != 0x000F0000)
// Check for Key Press or Not
{
delay(1000);           //delay(100)//give debounce delay check again temp2 = IO1PIN;
temp2 = temp2 & 0x000F0000;     //changed condition is same

if(temp2 != 0x000F0000)    // store the value in res1
{
flag = 0xff;
```

```
res1 = temp2;

t = (temp3 & 0x00F00000);          //Verfying Row Write res1 = res1 | t;//final scan value is stored in
res1

}

else

{

flag = 0x00;

}

}

} // end of scan()


void display(void)

{

ptr = disp0;

temp1 = 0x80;       // Display starting address of first line lcd_com();

while(*ptr!='\0')

{

temp1 = *ptr; lcd_data(); ptr ++;

}

ptr = disp1;

temp1 = 0xC0;       // Display starting address of second line lcd_com();
```

```
while(*ptr!='\0')
{
temp1 = *ptr; lcd_data();
ptr ++;
}
temp1 = 0xC6;        //display address for key value lcd_com();
temp1 = result; lcd_data();
}
void lcd_init (void)
{
temp = 0x30; wr_cn(); delay(3200);
temp = 0x30; wr_cn(); delay(3200);
temp = 0x30; wr_cn(); delay(3200);
temp = 0x20; wr_cn(); delay(3200);
// load command for lcd function setting with lcd in 4 bit mode,
// 2 line and 5x7 matrix display
temp = 0x28; lcd_com(); delay(3200);
// load a command for display on, cursor on and blinking off
 temp1 = 0x0C; lcd_com(); delay(800);
// command for cursor increment after data dump temp1 = 0x06;
lcd_com(); delay(800);
temp1 = 0x80; lcd_com(); delay(800);
```

```
}
void lcd_data(void)
{
temp = temp1 & 0xf0; wr_dn();
temp= temp1 & 0x0f; temp= temp << 4; wr_dn();
delay(100);
}
void wr_dn(void)    ////write data reg
{
IO0CLR = 0x000000FC;       // clear the port lines.
IO0SET = temp;             // Assign the value to the PORT lines IO0SET = 0x00000004;          // set
bit RS = 1
IO0SET = 0x00000008;       // E=1
delay(10);
IO0CLR = 0x00000008;
}
void lcd_com(void)
 {
temp = temp1 & 0xf0; wr_cn();
temp = temp1 & 0x0f; temp = temp << 4; wr_cn();
delay(500);
}
```

```
void wr_cn(void)    //write command reg
{
IO0CLR = 0x000000FC;       // clear the port lines.
IO0SET       = temp;       // Assign the value to the PORT lines IO0CLR = 0x00000004;
                           // clear bit RS = 0
IO0SET = 0x00000008;       // E=1
delay(10);
IO0CLR = 0x00000008;
}
void clr_disp(void)
{
temp1 = 0x01;       // command to clear lcd display lcd_com();
delay(500);
}
void delay(unsigned int r1)
{
for(r=0;r<r1;r++);
}
void init_port()
{
IO0DIR = 0x000000FC;       //configure o/p lines for lcd IO1DIR = 0XFFF0FFFF;
}
```

Virtual Lab Experiments

1. To write the Embedded C Program to Blinking the LED with time delay intervals using LPC2148 ARM Micro Controller.

COMPONENTS REQUIRED:

1. LPC 2148 ARM Microcontroller Development board.
2. Keil µVision version 5
3. Flash Magic

```
#include <lpc214x.h>

unsignedint delay;

int main(void)

{

 IO1DIR = (1<<20);

while(1)

{

IO1CLR = (1<<20); // CLEAR (0) P1.20 to turn LED ON

for(delay=0; delay<500000; delay++); // delay

IO1SET = (1<<20); // SET (1) P1.10 to turn LEDs OFF

for(delay=0; delay<500000; delay++); // delay

}

}
```

2. INTERFACE THE BUZZER

COMPONENTS REQUIRED:

1. LPC 2148 ARM Microcontroller Development board.
2. Keil µVision version 5
3. Flash Magic

```
#include <LPC214x.h>

#include <stdio.h>

#define BUZZ 7

void Delay(void);

void Wait(void);

void Delay()

{

        unsignedinti,j;

        for(i=0;i<1000;i++)

        for(j=0;j<700;j++);

}

void main()

{

         PINSEL0 = 0x00; //Configure Port0.7 as GPIO

        IODIR0 = 3 << BUZZ; //Configure Port0.7 as OutPut pin
```

```
        while(1)
        {
                IOSET0 = 1 << BUZZ;
                Delay();
                IOCLR0 = 1 << BUZZ;
                Delay();
        }
}
```

KLS

Vishwanathrao Deshpande Institute of Technology

Haliyal – 581329

GENERETIVE AI

for

VI Semester B.E.

As prescribed by

VISVESVARAYA TECHNOLOGICAL UNIVERSITY,

BELAGAVI – 590014

(From Academic Year: 2025-26)

Prepared by

Prof. Bheerappa Sasanoor

Department of Computer Science & Engineering (AIML)

Index

Index

PROGRAM OUTCOMES(POs)

Program Outcomes as defined by NBA (PO) Engineering Graduates will be able to:

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

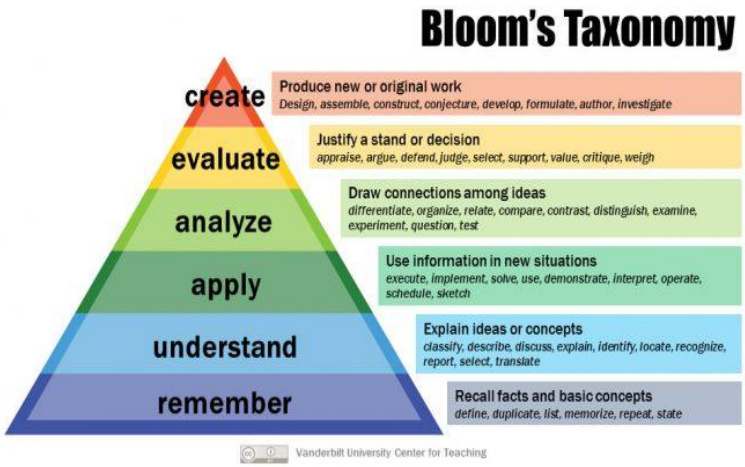
8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.



Vision (College)
To nurture talent and enrich society through excellence in technical education, research and innovation.
Mission (College)
1.To augment innovative Pedagogy, kindle quest for interdisciplinary learning & to enhance conceptual understanding. 2.To build competence,professional ethics & develop entrepreneurial thinking. 3.To strengthen Industry Institute Partnership & explore global collaborations. 4.To inculcate culture of socially responsible citizenship. 5.To focus on Holistic & Sustainable development.
Vision (Dept)
To achieve excellence in technical education, research, and innovation in computer science and Engineering by emphasizing on global trending technologies.
Mission (Dept)
To train students with conceptual understanding through innovative pedagogies. To imbibe professional, research, and entrepreneurial skills with a commitment to the nation's development at large. To strengthen the industry-institute Interaction. To promote life–long learning with a sense of societal & ethical responsibilities.

Program Educational Objectives (PEO)	
PEO1	To develop an ability to identify and analyze the requirements of Computer Science and Engineering in design and providing novel engineering solutions.
PEO2	To develop abilities to work in team on multidisciplinary projects with effective communication skills, ethical qualities and leadership roles.
PEO3	To develop abilities for successful Computer Science Engineer and achieve higher career goals.
Program Specific Outcomes (PSO)	
PSO 1	To develop the ability to model real-world problems using appropriate data structure and suitable algorithms in the area of Data Processing, System Engineering, and Networking for varying complexity.
PSO 2	To develop an ability to use modern computer languages, environments and platforms in creating innovative career.

CO's And PO's Mapping Chart

Subject with code: Generative AI [BAIL657C] **AY: 2025-26**

Semester: VI Sem

SL. No.	Description	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO1	PSO2
1	Develop the ability to explore and analyze word embeddings, perform vector arithmetic to investigate word relationships, visualize embeddings using dimensionality reduction techniques	3													
2	Apply prompt engineering skills to real-world scenarios, such as information retrieval, text generation.		3											2	
3	Utilize pre-trained Hugging Face models for real-world applications, including sentiment analysis and text summarization		3												
4	Apply different architectures used in large language models, such as transformers, and understand their advantages and limitations.		3												2

Degree of compliance Low:1 Medium:2 High:3

Evaluation:

	Particulars	Marks	Total
CIA	Performance	10	15
	Journal	03	
	Viva-voce	02	
	Lab IA	05	10
	Grand Total		**25**

Mapping of Experiments with CO, PO and PSO

Sl.No.	Content	CO	PO
1	Explore pre-trained word vectors. Explore word relationships using vector arithmetic. Perform arithmetic operations and analyze results.	1	3
2	Use dimensionality reduction (e.g., PCA or t-SNE) to visualize word embeddings for Q 1. Select 10 words from a specific domain (e.g., sports, technology) and visualize their embeddings. Analyze clusters and relationships. Generate contextually rich outputs using embeddings. Write a program to generate 5 semantically similar words for a given input.	1	3
3	Train a custom Word2Vec model on a small dataset. Train embeddings on a domain-specific corpus (e.g., legal, medical) and analyze how embeddings capture domain-specific semantics.	2	3
4	Use word embeddings to improve prompts for Generative AI model. Retrieve similar words using word embeddings. Use the similar words to enrich a GenAI prompt. Use the AI model to generate responses for the original and enriched prompts. Compare the outputs in terms of detail and relevance.	2	3
5	Use word embeddings to create meaningful sentences for creative tasks. Retrieve similar words for a seed word.Create a sentence or story using these words as a starting point. Write a program that: Takes a seed word. Generates similar words. Constructs a short paragraph using these words.	2	3
6	Use a pre-trained Hugging Face model to analyze sentiment in text. Assume a real-world application, Load the sentiment analysis pipeline. Analyze the sentiment by giving sentences to input.	2	3
7	Summarize long texts using a pre-trained summarization model using Hugging face model. Load the summarization pipeline. Take a passage as input and obtain the summarized text.	3	3
8	Install langchain, cohere (for key), langchain-community. Get the api key(By logging into Cohere and obtainingthe cohere key). Load a text document from your google drive . Create a prompt template to display the output in a particular manner.	4	3
9	Take the Institution name as input. Use Pydantic to define the schema for the desired output and create a custom output parser. Invoke the Chain and Fetch Results. Extract the below Institution related details from Wikipedia:**The founder of the Institution. When it was founded. The current branches in the institution . How many employees are working in it. A brief 4-line summary of the institution.**	3	3
10	Build a chatbot for the Indian Penal Code. We'll start by downloading the official Indian Penal Code document,and then we'll create a chatbot that can interact with it. Users will be able to ask questions about the Indian Penal Code and have a conversation with it.	4	3

EXPERIMENT WISE LESSON PLAN

Experiment No.1	
Name	**Exploring Pre-Trained Word Vectors**.
Objectives	Understand word relationships using vector arithmetic
Experiment No.2	
Name	**Visualizing Word Embeddings Using Dimensionality Reduction**
Objectives	Explore word relationships through visualization
Experiment No.3	
Name	**Training a Custom Word2Vec Model**
Objectives	Train a Word2Vec model on a small domain-specific dataset.
Experiment No.4	
Name	**Using Word Embeddings to Improve AI Prompting**
Objectives	Improve AI-generated text using word similarity.
Experiment No. 5	
Name	**Text Augmentation Using Synonyms**
Objectives	Improve dataset quality using AI to suggest synonyms for words
Experiment No.6	
Name	**Sentiment Analysis with Hugging Face Transformers**.
Objectives	Analyze sentiment in text using a pre-trained AI model.
Experiment No.7	
Name	**Summarization of Long Texts Using AI**.
Objectives	Generate summaries of large text documents using AI models.
Experiment No.8	
Name	**Installing and Using Cohere AI for Text Generation**
Objectives	Install and use Cohere API for AI-based text generation
Experiment No.9	
Name	**Extracting Institution Information Using Pyldantic**
Objectives	Use Pyldantic to extract structured data from text.
Experiment No.10	
Name	**Building a Chatbot for the Indian Penal Code**
Objectives	Develop an AI chatbot that interacts with the Indian Penal Code.

INTRODUCTION

Generative AI refers to artificial intelligence models that generate new data, such as text, images, and music, based on patterns learned from existing data. This lab manual aims to provide hands-on experience with Generative AI techniques, including word embeddings, AI-based summarization, chatbot development, and more.

Course Objectives

- Understand the principles and concepts behind Generative AI models.
- Implement generative models using prompt design frameworks.
- Apply various Generative AI applications for productivity enhancement.
- Develop Large Language Model-based applications.

To execute these programs, follow these steps:

General Setup

Install Dependencies

Ensure you have Python installed (preferably version 3.7+).
Install required libraries using:

"pip install gensim transformers cohere sklearn matplotlib wikipedia-api"

Download Pre-trained Models (If Required)

For **Word2Vec (Google News model)**:

wget -c "https://s3.amazonaws.com/dl4j-distribution/GoogleNews-vectors-negative300.bin.gz"

gunzip GoogleNews-vectors-negative300.bin.gz

Hugging Face Transformers and Cohere APIs require an internet connection.

Execution Steps for Each Program

Exploring Pre-Trained Word Vectors

Ensure GoogleNews-vectors-negative300.bin is downloaded.

Run the Python script:

python experiment1.py

Visualizing Word Embeddings

Ensure matplotlib and sklearn are installed.

Run the script and check the **graph output**.

Training a Custom Word2Vec Model

Run the script and check the **output for similar words**.

Using Word Embeddings for Prompting

Ensure transformers is installed.

Run the script and check **AI-generated text output**.

Text Augmentation Using Synonyms

Ensure GoogleNews-vectors-negative300.bin is available.

Run the script and check **synonyms list**.

Sentiment Analysis with Hugging Face

Ensure transformers is installed.

Run the script and check **positive/negative sentiment output**.

Summarization of Long Texts

Ensure transformers is installed.

Run the script and check **short summary output**.

Using Cohere for AI Text Generation

Sign up at cohere.com to get an API key.

Replace API_KEY in the script with your actual key.

Run the script and check **AI-generated text output**.

Extracting Institution Information

Ensure wikipedia-api is installed.

Run the script and check **institution details output**.

Building a Chatbot for the Indian Penal Code

Ensure transformers is installed.

Run the script and check **question-answering chatbot output**.

List of Experiments

Experiment 1: Exploring Pre-Trained Word Vectors

Objective: Understand word relationships using vector arithmetic.

Procedure: Load a pre-trained word vector model (e.g., Word2Vec, GloVe) and perform arithmetic operations.

Program:1

```
from gensim.models import KeyedVectors
model = KeyedVectors.load_word2vec_format('GoogleNews-vectors-negative300.bin', binary=True)
print(model.most_similar(positive=['king', 'woman'], negative=['man']))
```

Expected Output: List of words similar to 'queen' in vector space.

Experiment 2: Visualizing Word Embeddings Using Dimensionality Reduction

Objective: Explore word relationships through visualization.

Procedure: Select 10 words from a domain (sports, technology, legal, etc.), visualize embeddings using PCA/t-SNE.

Program:

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import numpy as np
words = ['king', 'queen', 'man', 'woman', 'apple', 'orange', 'cat', 'dog']
word_vectors = np.array([model[word] for word in words])
pca = PCA(n_components=2)
reduced = pca.fit_transform(word_vectors)
plt.scatter(reduced[:,0], reduced[:,1])
for i, word in enumerate(words):
plt.annotate(word, (reduced[i,0], reduced[i,1]))
plt.show()
```

Expected Output: A 2D scatter plot showing the relationship between words.

Experiment 3: Training a Custom Word2Vec Model

Objective: Train a Word2Vec model on a small domain-specific dataset.

Procedure: Use a dataset related to a particular domain (e.g., legal) and generate embeddings.

Program:

```
from gensim.models import Word2Vec
sentences = [['law', 'justice', 'court', 'judge', 'legal'], ['contract', 'agreement', 'lawyer', 'case']]
model = Word2Vec(sentences, vector_size=50, min_count=1)
print(model.wv.most_similar('law'))
```

Expected Output: Words similar to 'law' based on the trained embeddings.

Experiment 4: Using Word Embeddings to Improve AI Prompting

Objective: Improve AI-generated text using word similarity.

Procedure: Utilize embeddings to refine text generation prompts and compare outputs.

Program:

```
import random
SIMILAR_WORDS = {
    "photosynthesis": ["photosynthesis", "plant growth", "sunlight",
          "chlorophyll", "carbon dioxide"],
    "process": ["method", "procedure", "system", "approach",
             "technique"],
    "describe": ["explain", "depict", "illustrate", "outline",
              "narrate"],
    "energy": ["power", "strength", "force", "vitality", "fuel"],
}
def preprocess_prompt(prompt):
    # Tokenize the prompt
    tokens = prompt.lower().split()
    return tokens
def get_similar_words(word, top_n=3):
    return SIMILAR_WORDS.get(word, [word])[:top_n]
def enrich_prompt(prompt):
    tokens = preprocess_prompt(prompt)
enriched_tokens = []
    for token in tokens:
similar_words = get_similar_words(token)
```

```
enriched_tokens.append(random.choice(similar_words))
enriched_prompt = ' '.join(enriched_tokens)
    return enriched_prompt
def generate_response(prompt):
    h some modification
    return f"This is a generated response based on the
            prompt: {prompt}"
def compare_prompts(original_prompt):
enriched_prompt = enrich_prompt(original_prompt)
    print("\nOriginal Prompt: ", original_prompt)
    print("Enriched Prompt: ", enriched_prompt)
original_response = generate_response(original_prompt)
enriched_response = generate_response(enriched_prompt)
    print("\nResponse to Original Prompt:\n", original_response)
    print("\nResponse to Enriched Prompt:\n", enriched_response)
if __name__ == "__main__":
original_prompt = "Describe the process of photosynthesis."
compare_prompts(original_prompt)
```

Expected Output:

Original Prompt: Describe the process of photosynthesis.

Enriched Prompt: explain the system of photosynthesis.

Response to Original Prompt:

This is a generated response based on the prompt: Describe the process of photosynthesis.

Response to Enriched Prompt:

This is a generated response based on the prompt: explain the system of photosynthesis.

Experiment 5: Text Augmentation Using Synonyms

Objective: Improve dataset quality using AI to suggest synonyms for words.

Procedure: Use word embeddings to find synonyms for a word and replace them in a sentence.

Program:

```
import random
similar_words_dict = {
    "king": ["monarch", "ruler", "emperor", "sovereign", "prince"],
    "dog": ["puppy", "canine", "hound", "poodle", "retriever"],
    "computer": ["laptop", "desktop", "machine", "device",
                 "technology"],
    "city": ["town", "metropolis", "capital", "village",
             "settlement"],
    "ocean": ["sea", "water", "waves", "beach", "coast"]
}
def get_similar_words(seed_word, top_n=5):
    return similar_words_dict.get(seed_word, [])[:top_n]
def create_story(seed_word):
similar_words = get_similar_words(seed_word)
    if not similar_words:
        return "No similar words found for the given seed word."
    word1, word2, word3 =random.sample(similar_words,
                                       min(len(similar_words), 3))
       story = f"Once upon a time in a distant land,
        there was a {word1}
    who dreamed of becoming a great {word2}. " \
```

```
    f"Everyone in the village admired the {word3}, and they believed
       that one day the {word1} would make a lasting
       impact on the world."
       return story
if __name__ == "__main__":
seed_word = input("Enter a seed word: ")
       story = create_story(seed_word)
       print("\nGenerated Story:")
       print(story)
```

Expected Output

Enter a seed word: city

Generated Story:

Once upon a time in a distant land, there was a settlement who dreamed

of becoming a great town. Everyone in the village admired the capital,

and they believed that one day the settlement would make a lasting impact on the world.

Experiment 6: Sentiment Analysis with Hugging Face Transformers

Objective: Analyze sentiment in text using a pre-trained AI model.

Procedure: Load a pre-trained sentiment analysis model and analyze user-input text.

Program:

```
# Install necessary libraries if not installed (optional)
import subprocess
import sys
# Function to install required packages
def install(package):
subprocess.check_call([sys.executable, "-m", "pip", "install", package])
# Install transformers and torch if they are not installed
try:
    from transformers import pipeline
    except ImportError:
    install("transformers")
    from transformers import pipeline
    try:
    import torch
    except ImportError:
    install("torch")
    # Load the pre-trained sentiment analysis pipeline
    sentiment_analyzer = pipeline("sentiment-analysis")
    # Get user input for sentiment analysis
    print("Please enter your sentences for sentiment analysis (type 'exit' to quit):")
    user_sentences = []
```

```
while True:
# Get user input sentence
user_input = input("Enter a sentence: ")
# Break the loop if the user types 'exit'
if user_input.lower() == 'exit':
 break
 # Append the sentence to the list
user_sentences.append(user_input)
# Analyze sentiment of the user-provided sentences
if user_sentences:
 results = sentiment_analyzer(user_sentences)
 # Displaying the results
 for sentence, result in zip(user_sentences, results):
 print(f"\nSentence: {sentence}")
print(f"Sentiment: {result['label']}, Confidence: {result['score']:.2f}")
 print("-" * 50)
else:
 print("No sentences provided for analysis.")
```

Expected Output

Please enter your sentences for sentiment analysis (type 'exit' to quit):

Enter a sentence: I recently tried 3M, and it exceeded my expectations! It creates a rich lather that easily lifts dirt and grime without stripping the wax.

The formula is pH-balanced, ensuring a gentle yet thorough clean. Plus, it rinses off easily, leaving no residue—just a spotless, streak-free shine.

The scent is pleasant, and a little goes a long way, making it a great value. Highly recommend for anyone looking to keep their car looking fresh and glossy

Enter a sentence: Shampoo was leaking in the cap and wrapper was just placed on bottle, didn't even have to peal it off. Really makes me question the genuinity of this product.

Enter a sentence: exit

Sentence: I recently tried 3M, and it exceeded my expectations! It creates a rich lather that easily lifts dirt and grime without stripping the wax.

The formula is pH-balanced, ensuring a gentle yet thorough clean. Plus, it rinses off easily, leaving no residue—just a spotless, streak-free shine.

The scent is pleasant, and a little goes a long way, making it a great value. Highly recommend for anyone looking to keep their car looking fresh and glossy

Sentiment: POSITIVE, Confidence: 1.00

Sentence: Shampoo was leaking in the cap and wrapper was just placed on bottle, didn't even have to peal it off. Really makes me question the genuinity of this product.

Sentiment: NEGATIVE, Confidence: 0.99

Experiment 7: Summarization of Long Texts Using AI

Objective: Generate summaries of large text documents using AI models.

Procedure: Use a pre-trained summarization model to generate a concise summary.

Program:

```
# Step 1: Import the Hugging Face pipeline
from transformers import pipeline
# Step 2: Load the summarization pipeline
summarizer = pipeline("summarization")
# Step 3: Input a long passage for summarization
long_text = """
Artificial Intelligence (AI) is transforming various industries by automating tasks, improving
efficiency,
and enabling new capabilities. In the healthcare sector, AI is used for disease diagnosis,
personalized medicine,
and drug discovery. In the business world, AI-powered systems are optimizing customer
service, fraud detection,
and supply chain management. AI's impact on everyday life is significant, from smart
assistants to recommendation
systems in streaming platforms. As AI continues to evolve, it promises even greater
advancements in fields like
education, transportation, and environmental sustainability.
"""
# Step 4: Summarize the input passage
summary = summarizer(long_text, max_length=50, min_length=20,
```

```
do_sample=False)[0]["summary_text"]

# Step 5: Print the summarized text

print("Summarized Text:")

print(summary)
```

Expected Output:

Summarized Text:

Artificial Intelligence (AI) is transforming various industries by automating tasks, improving efficiency and enabling new capabilities . In healthcare sector, AI is used for disease diagnosis, personalized medi cine, and drug discovery . In business world, AI-powered

Experiment 8: Installing and Using Cohere AI for Text Generation

Objective: Install and use Cohere API for AI-based text generation.

Procedure: Install Cohere, generate text using its API.

Program:

```
# Step 1: Install necessary libraries
!pip install langchain cohere langchain-community
# Step 2: Import the required modules from langchain.llms
 import Cohere from langchain.prompts
import PromptTemplate from langchain
import LLMChain from google.colab import drive
# Step 3: Mount Google Drive to access the document drive.mount('/content/drive')
# Step 4: Load the text document from Google Drive
file_path = "/content/drive/MyDrive/sample_text.txt"
# Change this path to your file location with open(file_path, "r") as file:
 text = file.read()
# Step 5: Set up Cohere API key
cohere_api_key = "YOUR_COHERE_API_KEY"
 # Replace with your actual Cohere API key
  # Step 6: Create a prompt template
prompt_template = """
Summarize the following text in three bullet points:
{text} """
# Step 7: Configure the Cohere model with Langchain
```

```
llm = Cohere(cohere_api_key=cohere_api_key)
 prompt = PromptTemplate(input_variables=["text"], template=prompt_template)
  # Step 8: Create an LLMChain with the Cohere model and prompt template
chain = LLMChain(llm=llm, prompt=prompt)
 # Step 9: Run the chain on the loaded text result = chain.run(text)
 # Step 10: Display the formatted output print("Summarized Output in Bullet Points:") print(result)
```

Expected Output

Summarized Output in Bullet Points:

- AI is transforming industries like healthcare, business, and education.
- Smart assistants and recommendation systems are examples of AI's impact on daily life.
- Future advancements will bring improvements in transportation and sustainability.

Experiment 9: Extracting Institution Information Using Pyldantic

Objective: Use Pyldantic to extract structured data from text.

Procedure: Define schema and extract structured data from Wikipedia.

Program:

```
# Step 1: Install necessary libraries
!pip install langchainpydanticwikipedia-api
# Step 2: Import required modules
from langchain.llms import Cohere
from langchain.prompts import PromptTemplate
from langchain import LLMChain
from pydantic import BaseModel
import wikipediaapi
# Step 1: Install necessary libraries
!pip install langchainpydanticwikipedia-api
# Step 2: Import required modules
from langchain.llms import Cohere
from langchain.prompts import PromptTemplate
from langchain import LLMChain
from pydantic import BaseModel
import wikipediaapi
# Step 3: Define a Pydantic schema for the institution's details
class InstitutionDetails(BaseModel):
    founder: str
    founded: str
```

```
    branches: str
    employees: str
    summary: str
# Step 4: Function to fetch details from Wikipedia with user-agent specified
def fetch_wikipedia_summary(institution_name):
wiki_wiki = wikipediaapi.Wikipedia(language='en',
user_agent="InstitutionInfoBot/1.0 (contact: youremail@example.com)")
    page = wiki_wiki.page(institution_name)
    if page.exists():
        return page.text
    else:
        return "No information available on Wikipedia for this institution."
# Step 5: Prompt template for extracting relevant details
prompt_template = """
Extract the following information from the given text:
- Founder
- Founded (year)
- Current branches
- Number of employees
- 4-line brief summary
Text: {text}
Provide the information in the following format:
Founder: <founder>
Founded: <founded>
Branches: <branches>
```

```
Employees: <employees>
Summary: <summary>
# Step 6: Take institution name as input
institution_name = input("Enter the name of the institution: ")
# Step 7: Fetch Wikipedia data for the institution
wiki_text = fetch_wikipedia_summary(institution_name)
# Step 8: Set up Cohere (Replace YOUR_COHERE_API_KEY with your actual key)
cohere_api_key = "YOUR_COHERE_API_KEY"
llm = Cohere(cohere_api_key=cohere_api_key)
# Step 9: Create the Langchain prompt and chain
prompt = PromptTemplate(input_variables=["text"], template=prompt_template)
chain = LLMChain(llm=llm, prompt=prompt)
# Step 10: Run the chain and parse the output
response = chain.run(wiki_text)
# Step 11: Parse the response using Pydantic
try:
    details = InstitutionDetails.parse_raw(response)
    print("Institution Details:")
print(f"Founder: {details.founder}")
print(f"Founded: {details.founded}")
    print(f"Branches: {details.branches}")
    print(f"Employees: {details.employees}")
    print(f"Summary: {details.summary}")
except Exception as e:
    print("Error parsing the response:", e)
```

Expected Output:

Enter the name of the institution: Google

Institution Details:

Founder: Larry Page, Sergey Brin

Founded: 1998

Branches: Global ofÏces in more than 50 countries

Employees: Over 100,000

Summary: Google is a multinational technology company specializing in internet-related services and products. It is known for its search engine, online advertising, cloud computing, and software. Google is one of the Big Five tech companies. It was founded by Larry Page and Sergey Brin in 1998

Experiment 10: Building a Chatbot for the Indian Penal Code

Objective: Develop an AI chatbot that interacts with the Indian Penal Code.

Procedure: Load legal documents and use AI to generate responses.

Program:

```
# Step 1: Install necessary packages
!pip install langchainpydanticwikipedia-apiopenai
# Step 2: Import required modules
from langchain.chains import load_qa_chain
from langchain.docstore.document import Document
```

- from langchain.llms import OpenAI

```
# Step 3: Load the Indian Penal Code text from a file
ipc_file_path = "path_to_your_ipc_file.txt"  # Replace with the actual path to your IPC text file
# Read the IPC document
with open(ipc_file_path, "r", encoding="utf-8") as file:
ipc_text = file.read()
# Step 4: Create a Langchain Document object
ipc_document = Document(page_content=ipc_text)
# Step 5: Set up OpenAI (or any other LLM of your choice)
llm = OpenAI(openai_api_key="YOUR_OPENAI_API_KEY", temperature=0.3)  # Use
temperature=0.3 for more factual responses
# Step 6: Create a simple question-answering chain
qa_chain = load_qa_chain(llm, chain_type="stuff")
# Step 7: Chat with the chatbot
```

```
print("Chatbot for the Indian Penal Code (IPC)")

print("Ask a question about the Indian Penal Code (type 'exit' to stop):")

while True:

user_question = input("\nYour question: ")

    if user_question.lower() == "exit":

        print("Goodbye!")

        break

    # Use the QA chain to answer the question

    response = qa_chain.run(input_documents=[ipc_document], question=user_question)

    print(f"Answer: {response}")
```

Expected Output:

Chatbot for the Indian Penal Code (IPC)

Ask a question about the Indian Penal Code (type 'exit' to stop):

Your question: What is Section 302 of the IPC?

Answer: Section 302 of the Indian Penal Code refers to punishment for murder,

- which is punishable with death or life imprisonment and a fine.
- Your question: exit
- Goodbye!

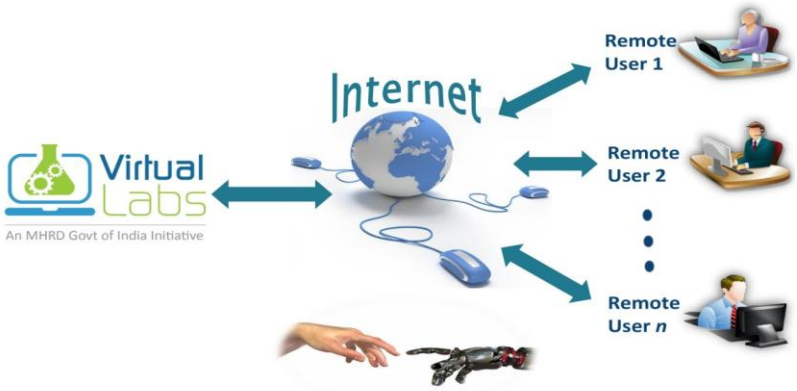
Virtual Lab Experiment

1. Extracting Institution Information Using Pyldantic

Objective: Use Pyldantic to extract structured data from text.

Procedure: Define schema and extract structured data from Wikipedia.

Program:



2. Building a Chatbot for the Indian Penal Code

Objective: Develop an AI chatbot that interacts with the Indian Penal Code.