**Note:**

**You need to attempt all MCQ questions and any 5 programming questions out of 10 in each section like Pandas, Numpy and ML,DL**

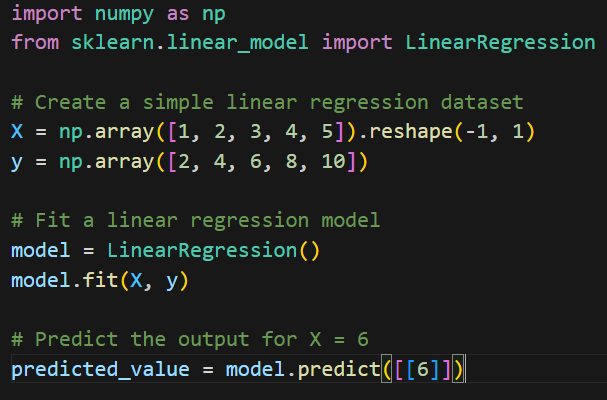
**1. # What is the predicted value for X = 6?**

**A. 11**

**B. 12**

**C. 13**

**D. 14**

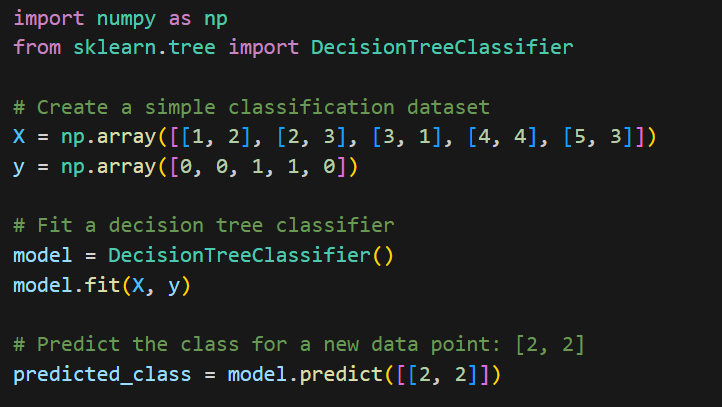


**2. What is the predicted class for the data point [2, 2]?**

**A. 0**

**B. 1**

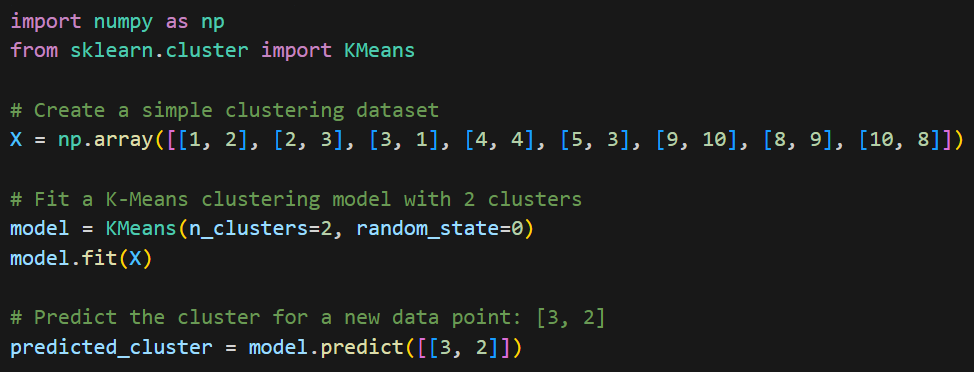
**C. It cannot be determined from the given information.**



**3. Which cluster is the data point [3, 2] assigned to?**

**A. Cluster 0**

**B. Cluster 1**



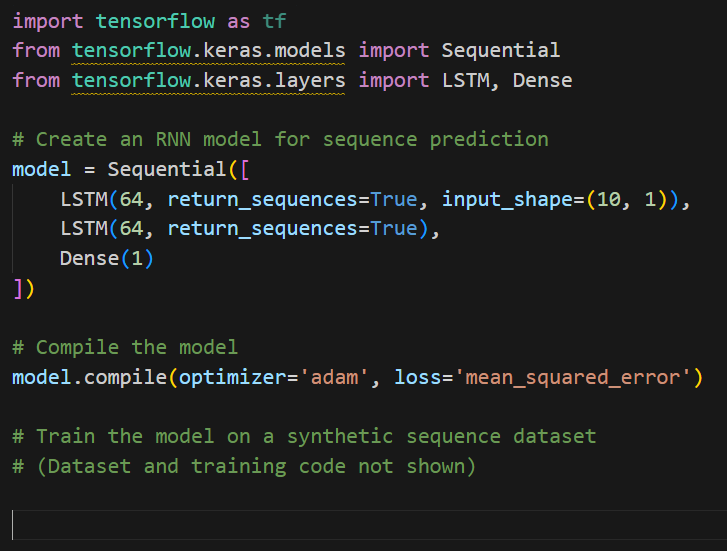
**4. After training, what type of problem is this RNN model suitable for?**

**A. Image classification**

**B. Text generation**

**C. Time series prediction**

**D. Reinforcement learning**



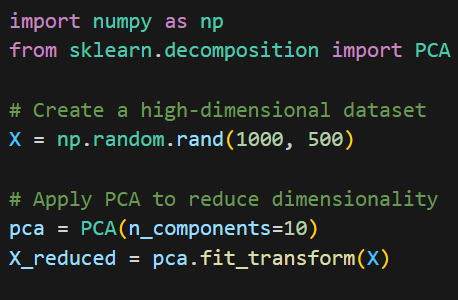
**5. What does the value of n\_components=10 represent in PCA?**

**A. It reduces the dataset to 10 features.**

**B. It retains the top 10 principal components.**

**C. It computes the explained variance ratio for the top 10 components.**

**D. It selects the first 10 data points.**



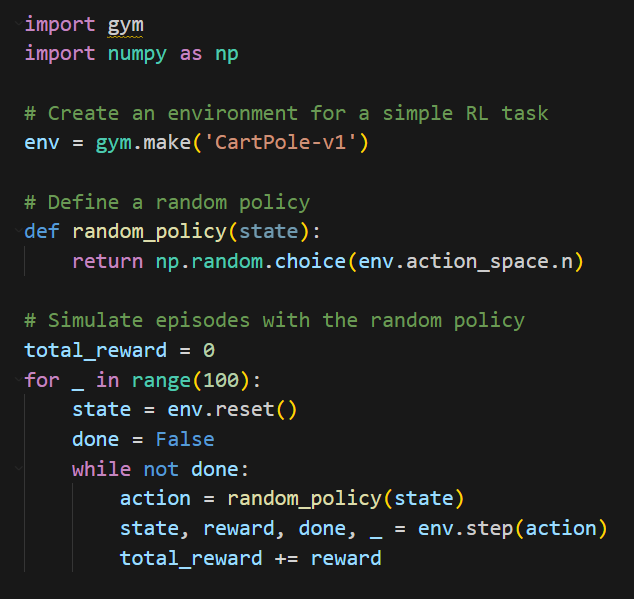
**6. What does the 'total\_reward' represent in this RL simulation?**

**A. The number of episodes completed.**

**B. The total number of time steps taken.**

**C. The cumulative reward obtained from 100 episodes.**

**D. The final state of the environment after 100 episodes.**

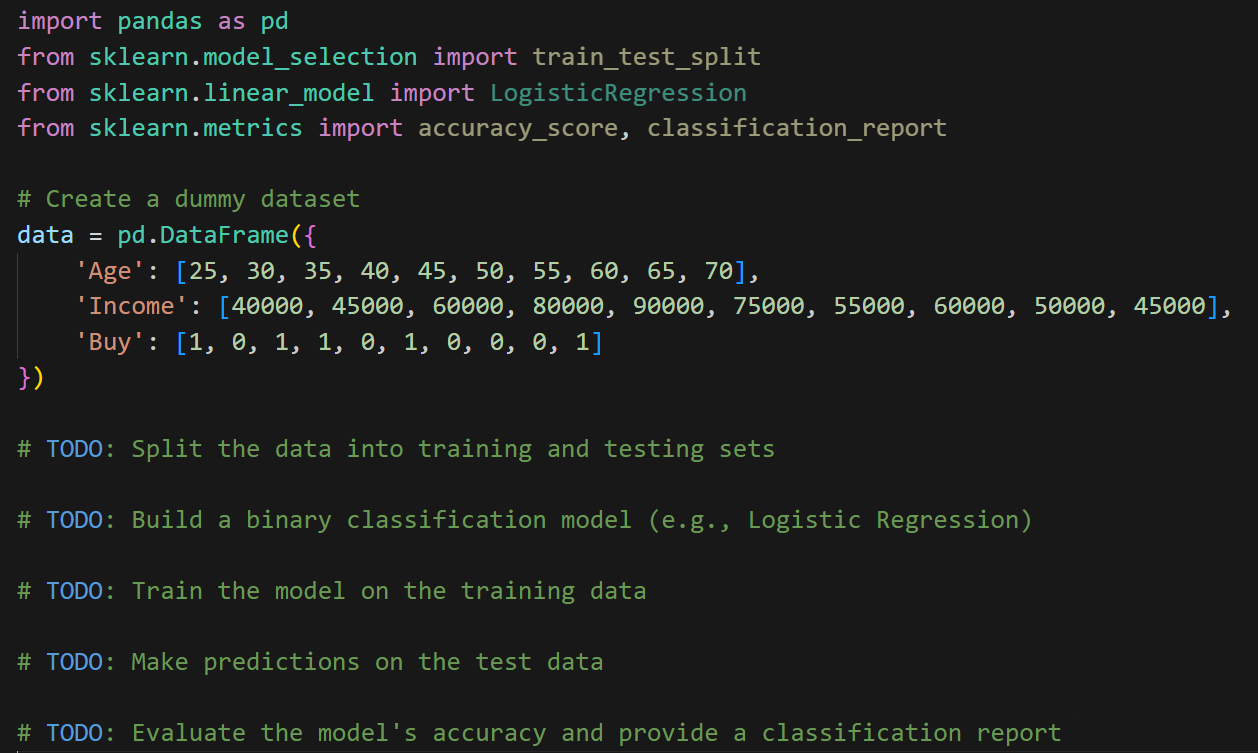


**­­­**

**Question 1: Binary Classification with scikit-learn**

**Problem Statement:**

You need to build a binary classification model using scikit-learn to predict whether a customer will buy a product (1 for yes, 0 for no) based on their age and income. Here's the dummy dataset:



Your tasks are similar to the previous questions: split the data, build a classification model, train it, make predictions, and evaluate the model's performance.

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, classification\_report

# Create a dummy dataset

data = pd.DataFrame({

'Age': [25, 30, 35, 40, 45, 50, 55, 60, 65, 70],

'Income': [40000, 45000, 60000, 80000, 90000, 75000, 55000, 60000, 50000, 45000],

'Buy': [1, 0, 1, 1, 0, 1, 0, 0, 0, 1]

})

# Split the data into training and testing sets

X = data[['Age', 'Income']]

y = data['Buy']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Build a binary classification model (e.g., Logistic Regression)

model = LogisticRegression()

# Train the model on the training data

model.fit(X\_train, y\_train)

# Make predictions on the test data

y\_pred = model.predict(X\_test)

# Evaluate the model's accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy:", accuracy)

# Provide a classification report

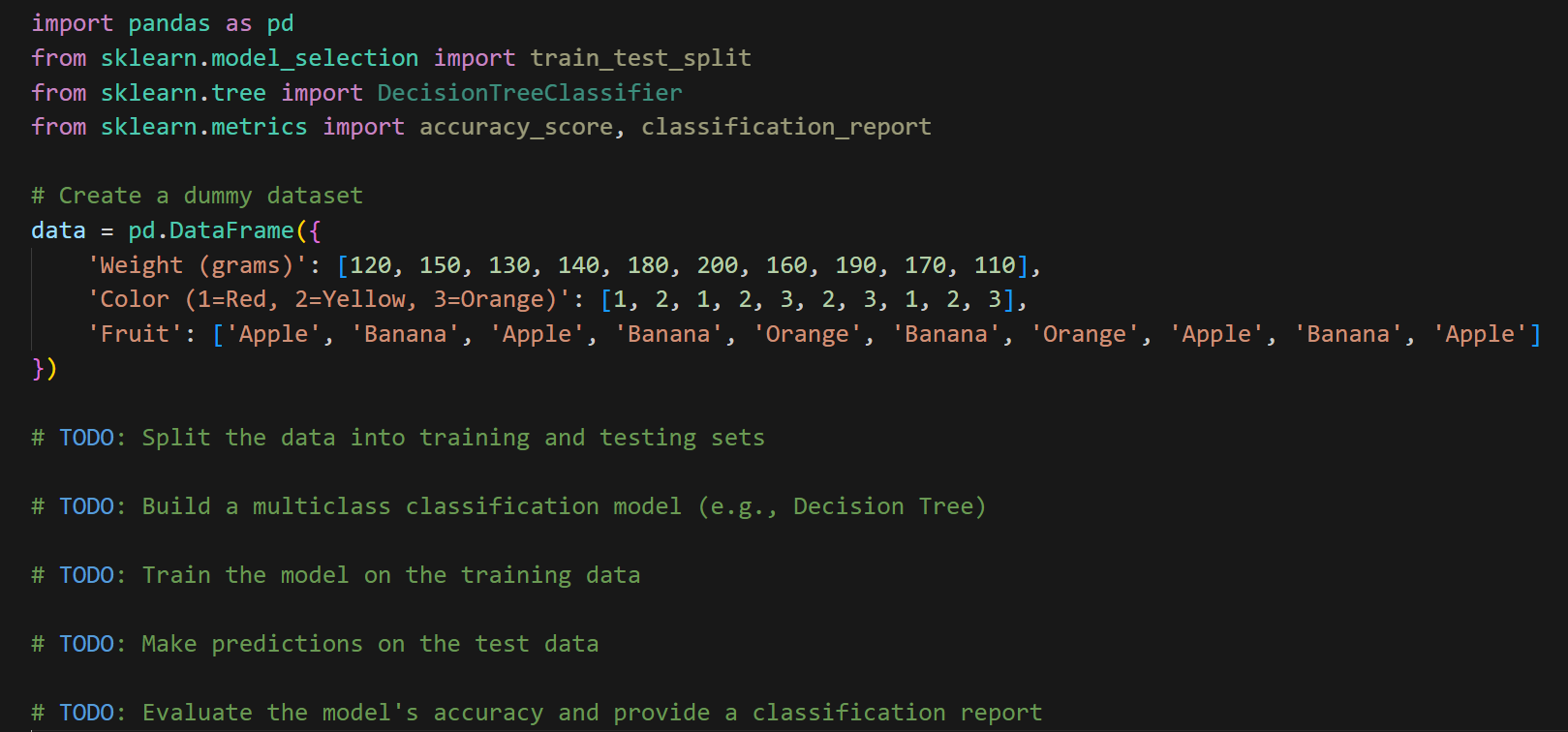
classification\_rep = classification\_report(y\_test, y\_pred)

print("\nClassification Report:\n", classification\_rep)

**Question 2: Multiclass Classification with Dummy Data**

**Problem Statement:**

Build a multiclass classification model using scikit-learn to classify fruits into three categories ("Apple," "Banana," "Orange") based on their weight and color. Here's the dummy dataset:



import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score, classification\_report

# Create a dummy dataset

data = pd.DataFrame({

'Weight (grams)': [120, 150, 130, 140, 180, 200, 160, 190, 170, 110],

'Color (1=Red, 2=Yellow, 3=Orange)': [1, 2, 1, 2, 3, 2, 3, 1, 2, 3],

'Fruit': ['Apple', 'Banana', 'Apple', 'Banana', 'Orange', 'Banana', 'Orange', 'Apple', 'Banana', 'Apple']

})

# Split the data into training and testing sets

X = data[['Weight (grams)', 'Color (1=Red, 2=Yellow, 3=Orange)']]

y = data['Fruit']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Build a multiclass classification model (e.g., Decision Tree)

model = DecisionTreeClassifier()

# Train the model on the training data

model.fit(X\_train, y\_train)

# Make predictions on the test data

y\_pred = model.predict(X\_test)

# Evaluate the model's accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy:", accuracy)

# Provide a classification report

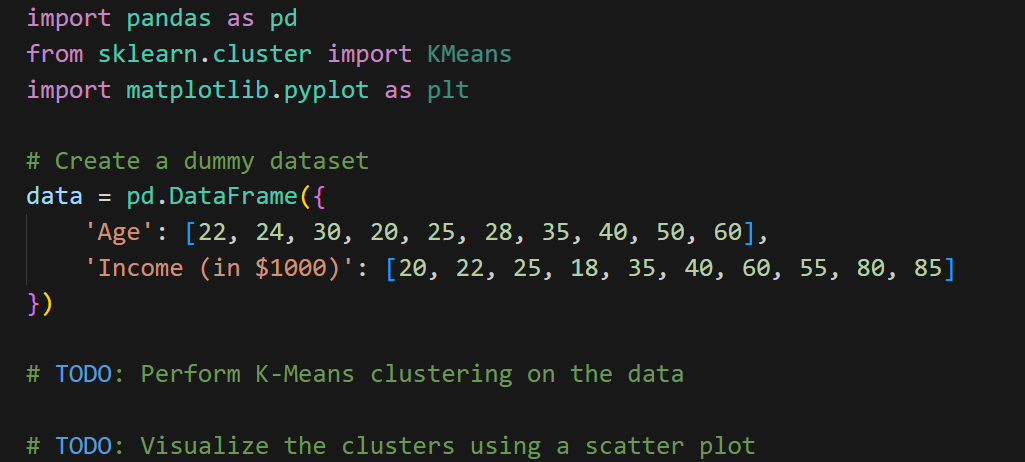
classification\_rep = classification\_report(y\_test, y\_pred)

print("\nClassification Report:\n", classification\_rep)

**Question 3: Clustering with scikit-learn**

**Problem Statement:**

You need to perform clustering on a dummy dataset of customer data to group similar customers together. Use the K-Means clustering algorithm from scikit-learn.



Your tasks are to perform K-Means clustering on the dataset and visualize the clusters using a scatter plot.

import pandas as pd

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

# Create a dummy dataset

data = pd.DataFrame({

'Age': [22, 24, 30, 20, 25, 28, 35, 40, 50, 60],

'Income (in $1000)': [20, 22, 25, 18, 35, 40, 60, 55, 80, 85]

})

# Perform K-Means clustering on the data

kmeans = KMeans(n\_clusters=2, random\_state=42)

data['Cluster'] = kmeans.fit\_predict(data[['Age', 'Income (in $1000)']])

# Visualize the clusters using a scatter plot

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

plt.scatter(data['Age'], data['Income (in $1000)'], c=data['Cluster'], cmap='viridis')

plt.title('K-Means Clustering')

plt.xlabel('Age')

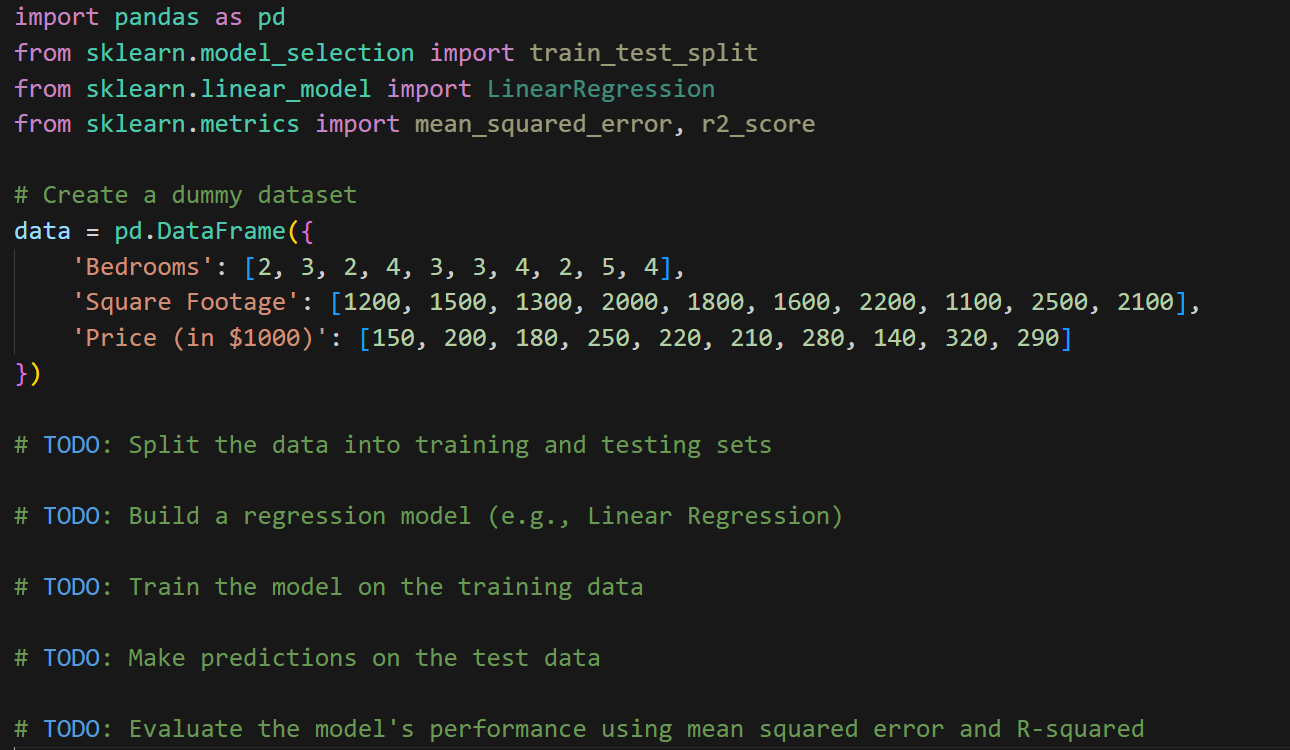
plt.ylabel('Income (in $1000)')

plt.show()

**Question 4: Regression with scikit-learn**

**Problem Statement:**

Build a regression model using scikit-learn to predict house prices based on the number of bedrooms and square footage. Here's the dummy dataset:



Your tasks are to split the data, build a regression model, train it, make predictions, and evaluate the model's performance.

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

# Create a dummy dataset

data = pd.DataFrame({

'Bedrooms': [2, 3, 2, 4, 3, 3, 4, 2, 5, 4],

'Square Footage': [1200, 1500, 1300, 2000, 1800, 1600, 2200, 1100, 2500, 2100],

'Price (in $1000)': [150, 200, 180, 250, 220, 210, 280, 140, 320, 290]

})

# Split the data into training and testing sets

X = data[['Bedrooms', 'Square Footage']]

y = data['Price (in $1000)']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Build a Linear Regression model

model = LinearRegression()

# Train the model on the training data

model.fit(X\_train, y\_train)

# Make predictions on the test data

y\_pred = model.predict(X\_test)

# Evaluate the model's performance using mean squared error (MSE)

mse = mean\_squared\_error(y\_test, y\_pred)

print("Mean Squared Error:", mse)

# Evaluate the model's performance using R-squared

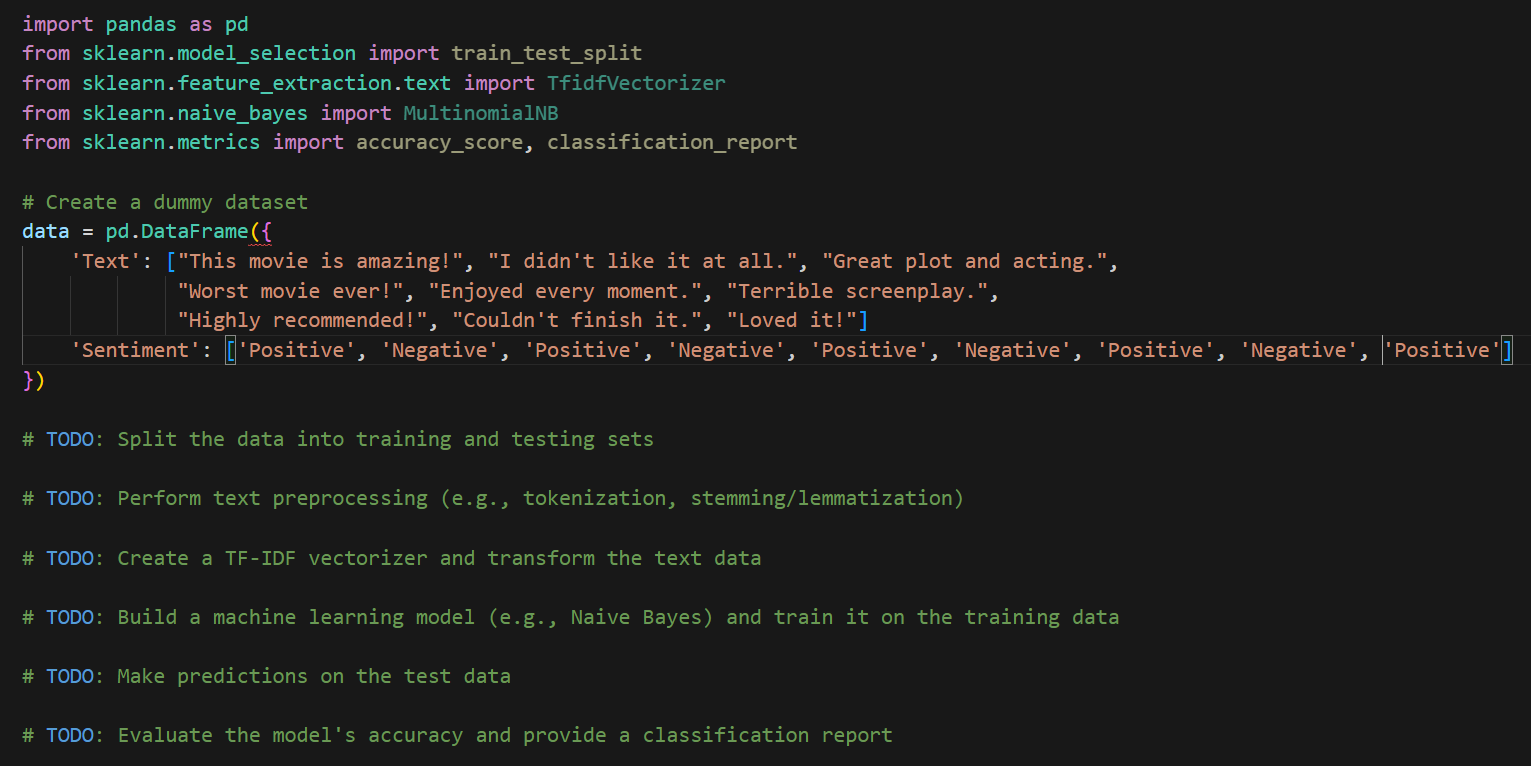
r2 = r2\_score(y\_test, y\_pred)

print("R-squared:", r2)

**Question 5: Natural Language Processing (NLP) with Text Classification**

**Problem Statement:**

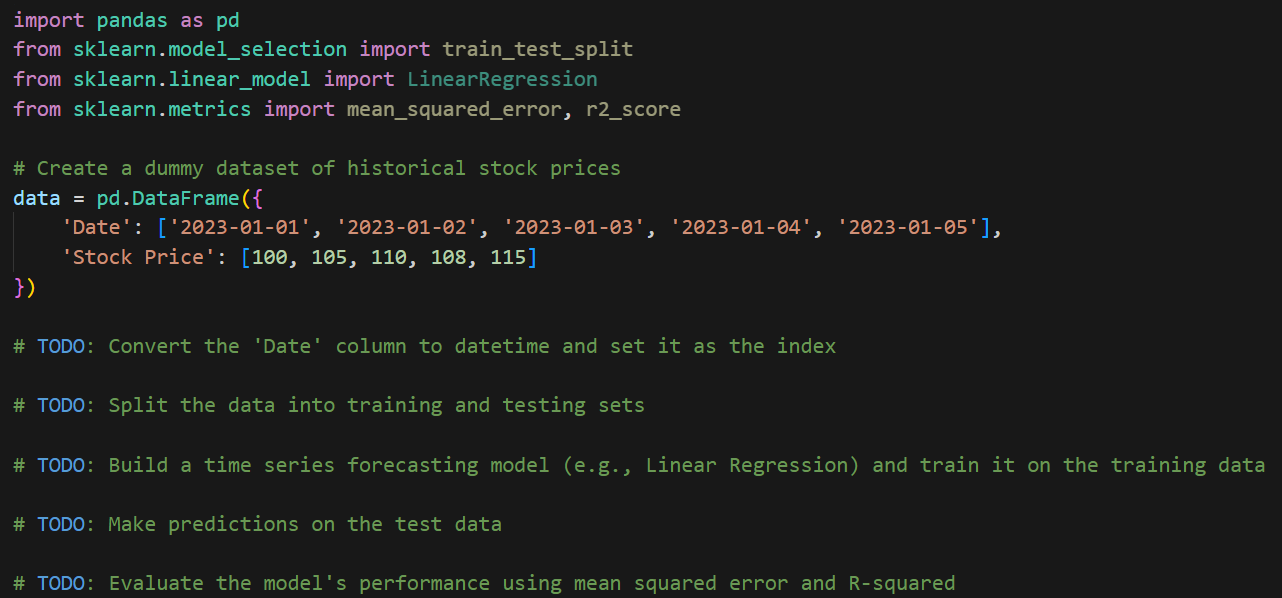
Perform text classification using scikit-learn to classify movie reviews into two categories ("Positive" and "Negative") based on their text content.



**Question 6: Time Series Forecasting**

**Problem Statement:**

Perform time series forecasting using scikit-learn to predict stock prices based on historical data.

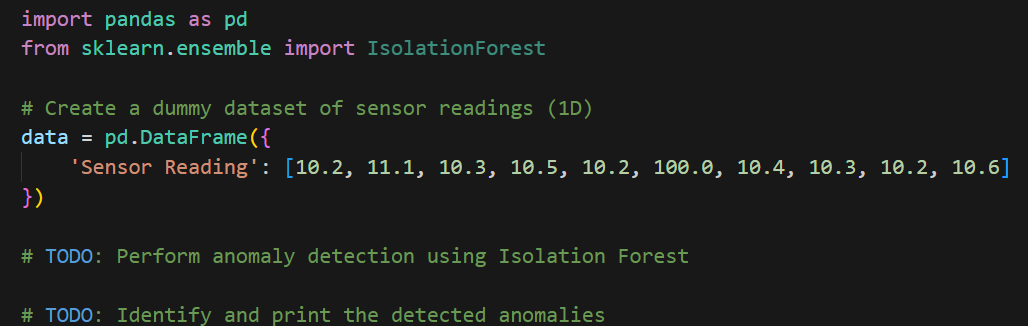


Your tasks are to convert the date column, split the data, build a time series forecasting model, train it, make predictions, and evaluate the model's performance.

**Question 7: Anomaly Detection with scikit-learn**

**Problem Statement:**

Detect anomalies in a dataset of sensor readings using scikit-learn.

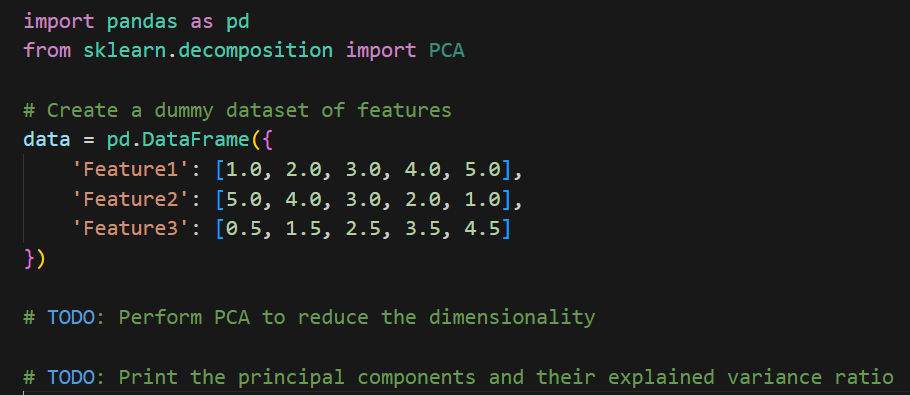


Your task is to perform anomaly detection using the Isolation Forest algorithm and identify the detected anomalies in the dataset.

**Question 8: Dimensionality Reduction with Principal Component Analysis (PCA)**

**Problem Statement:**

Perform dimensionality reduction using PCA on a dataset of features.

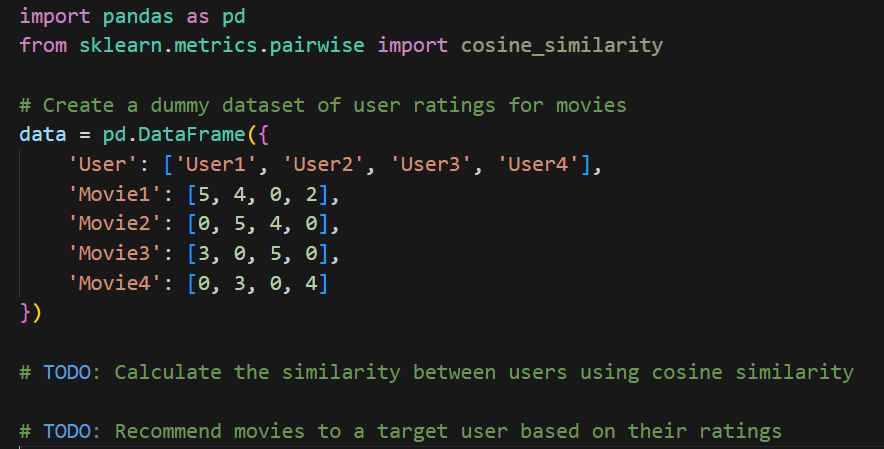


Your task is to perform PCA to reduce the dimensionality of the dataset and print the principal components along with their explained variance ratios.

**Question 9: Recommender System with Dummy User Ratings**

**Problem Statement:**

Build a basic recommender system that recommends movies to a user based on their ratings.

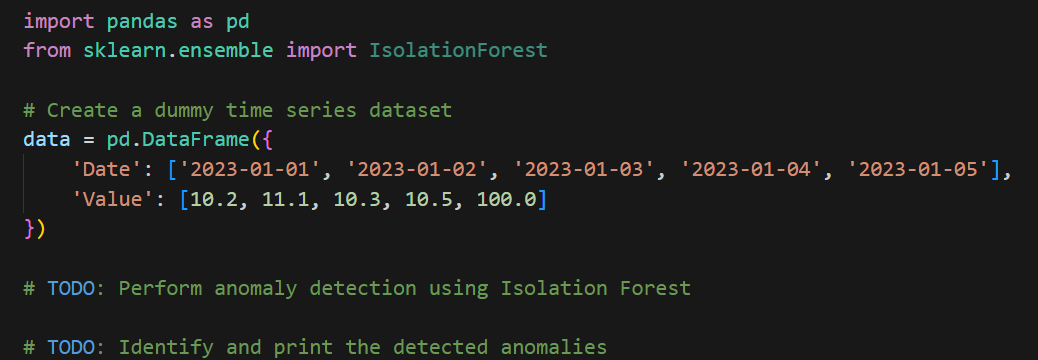


Your tasks are to calculate user similarity using cosine similarity and recommend movies to a target user based on their ratings.

**Question 10: Anomaly Detection with Dummy Time Series Data**

**Problem Statement:**

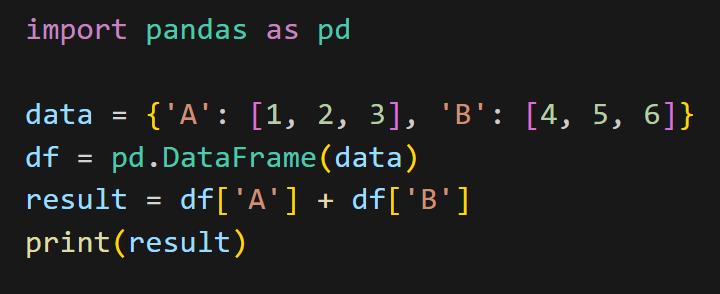
Detect anomalies in a time series dataset using scikit-learn.



Your task is to perform anomaly detection using the Isolation Forest algorithm on the time series data and identify the detected anomalies.

**Numpy and Pandas**

**Question 1:**

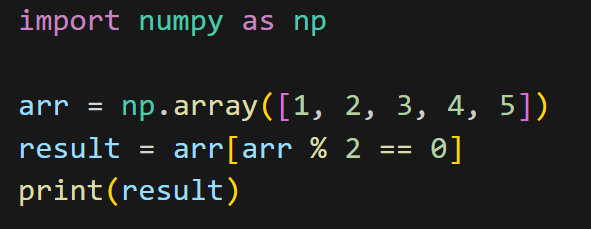


1. [5, 7, 9]
2. [1, 2, 3]

C. 10

D. [4, 5, 6]

**Question 2:**



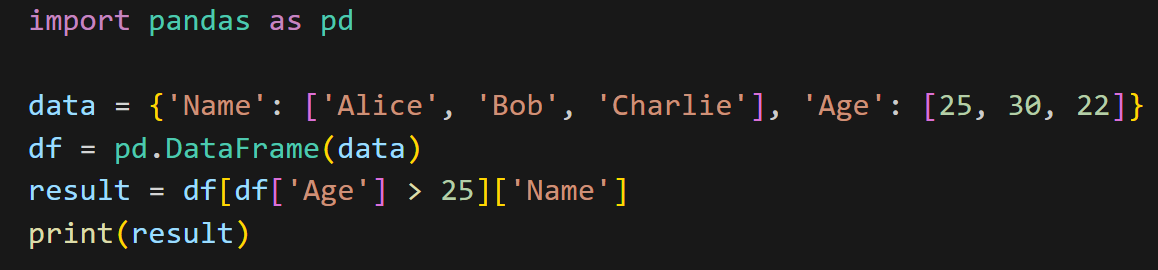
A. [1, 2, 3, 4, 5]

B. [2, 4]

C. [1, 3, 5]

D. []

**Question 3:**



A. ['Bob', 'Charlie']

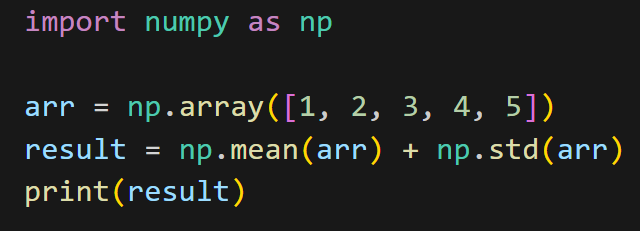
B. ['Alice', 'Bob', 'Charlie']

C. ['Alice', 'Bob']

D. ['Charlie']

According to question result should be Bob, but in answer there is no bob, the nearest answer is C, if it is greater than 25 then it is only bob, if it is greaterthan equal to 25 answer is alice and bob.

**Question 4:**



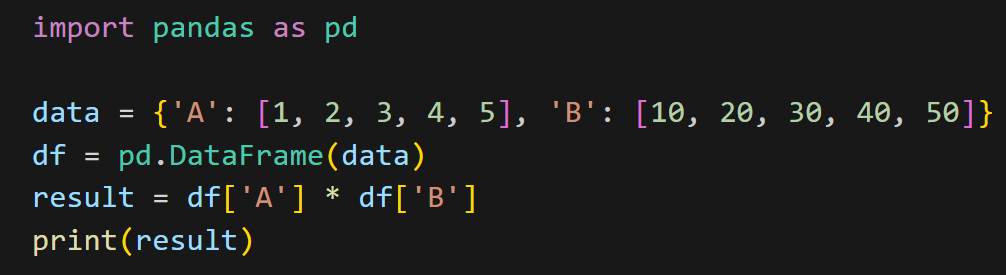
A. 2.0

B. 3.0

C. 1.0

D. 2.5

**Question 5:**



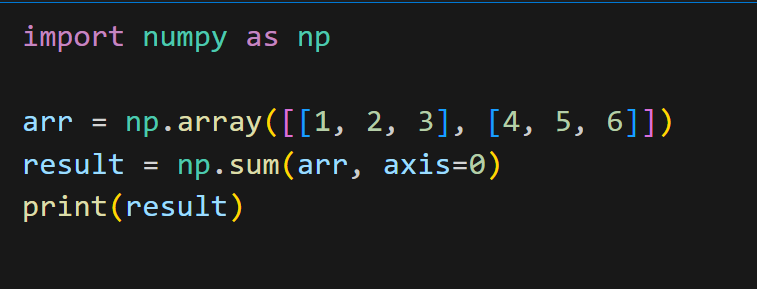
A. [10, 40, 90, 160, 250]

B. [1, 2, 3, 4, 5]

C. [10, 20, 30, 40, 50]

D. 150

**Question 6:**



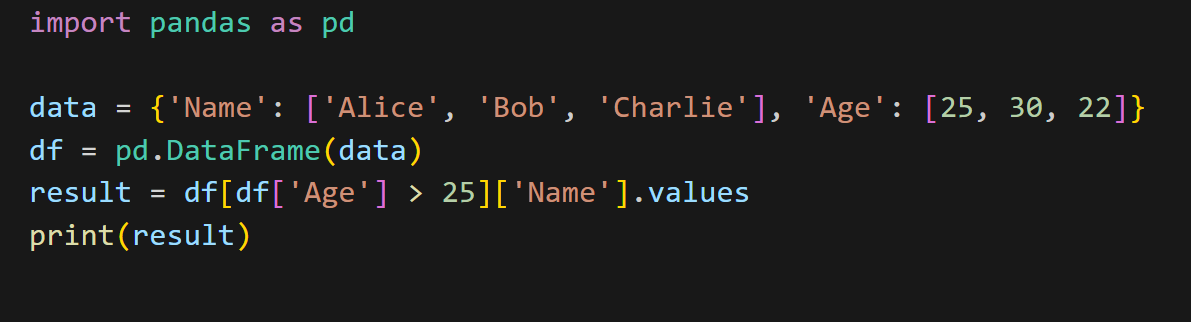
A. [5, 7, 9]

B. [10, 11, 12]

C. [1, 2, 3, 4, 5, 6]

D. [4, 5, 6]

**Question 7:**



A. ['Bob', 'Charlie']

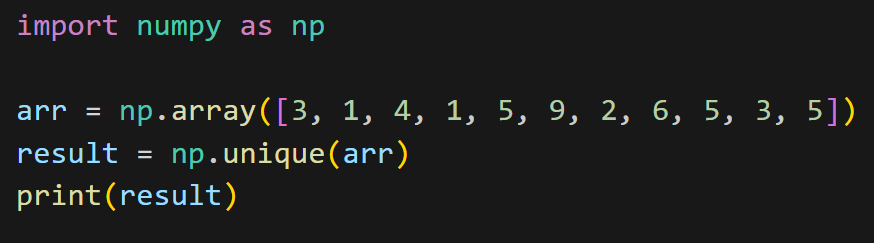
B. ['Alice', 'Bob', 'Charlie']

C. ['Alice', 'Bob']

D. ['Charlie']

Question is repeated original question is Q3

**Question 8:**



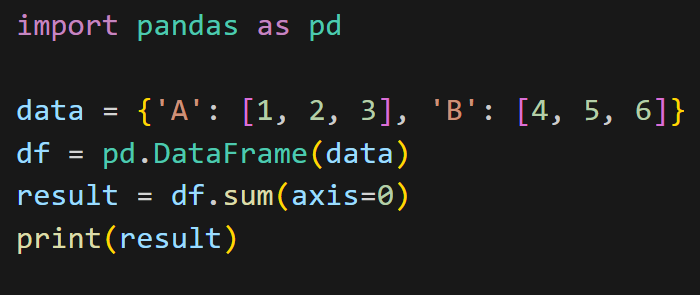
A. [1, 2, 3, 4, 5, 6, 9]

B. [3, 1, 4, 5, 9, 2, 6]

C. [1, 1, 2, 3, 3, 4, 5, 5, 6, 9]

D. [1, 4, 9, 2, 6]

**Question 9:**



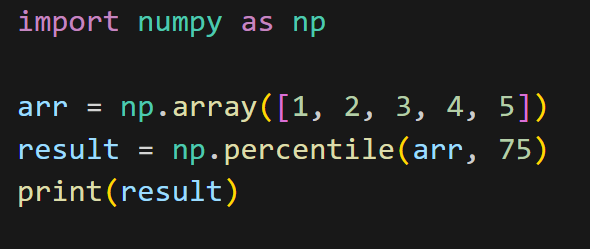
A. A 6 B 15 dtype: int64

B. [6, 15]

C. A 1 2 3 B 4 5 6 dtype: int64

D. [10, 15]

**Question 10:**



A. 1.0

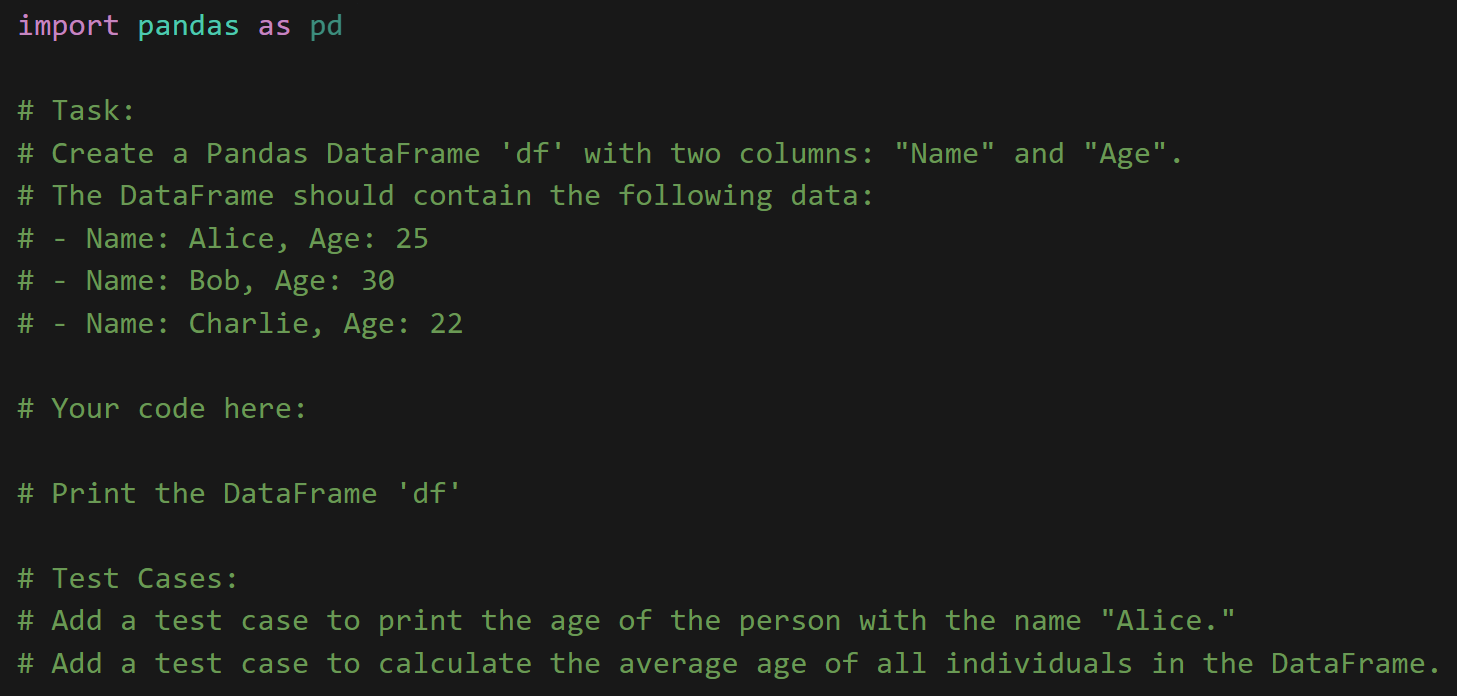
B. 3.0

C. 4.0

D. 3.75

**PROGRAMMING QUESTIONS**

1



import pandas as pd

# Task:

# Create a Pandas DataFrame 'f' with two columns: "Name" and "Age'

# The DataFrame should contain the following data:

# - Name: Alice, Age: 25

# - Name: Bob, Age: 30

# - Name: Charlie, Age: 22

# Your code here:

data = {'Name': ['Alice', 'Bob', 'Charlie'],

'Age': [25, 30, 22]}

df = pd.DataFrame(data)

# Print the DataFrame 'df'

print(df)

# Test Cases:

# Add a test case to print the age of the person with the name "Alice."

alice\_age = df[df['Name'] == 'Alice']['Age'].values[0]

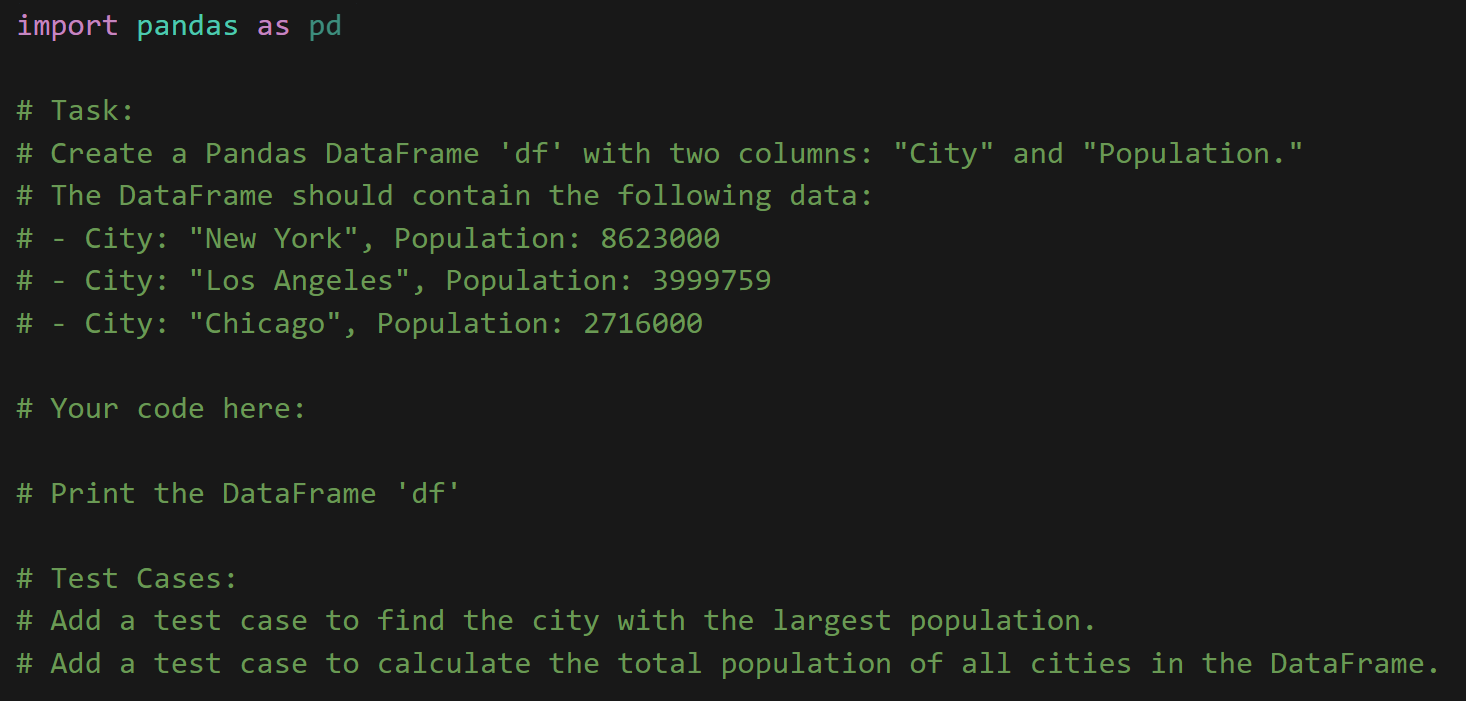
print("Alice's Age:", alice\_age)

# Add a test case to calculate the average age of all individuals in the DataFrame.

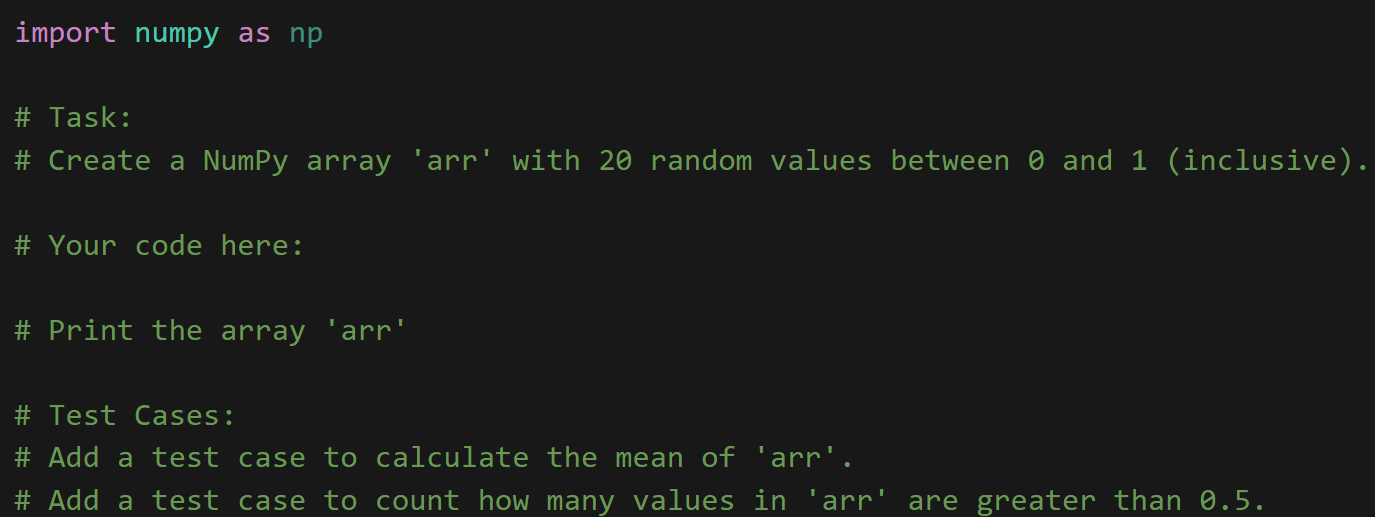
average\_age = df['Age'].mean()

print("Average Age:", average\_age)

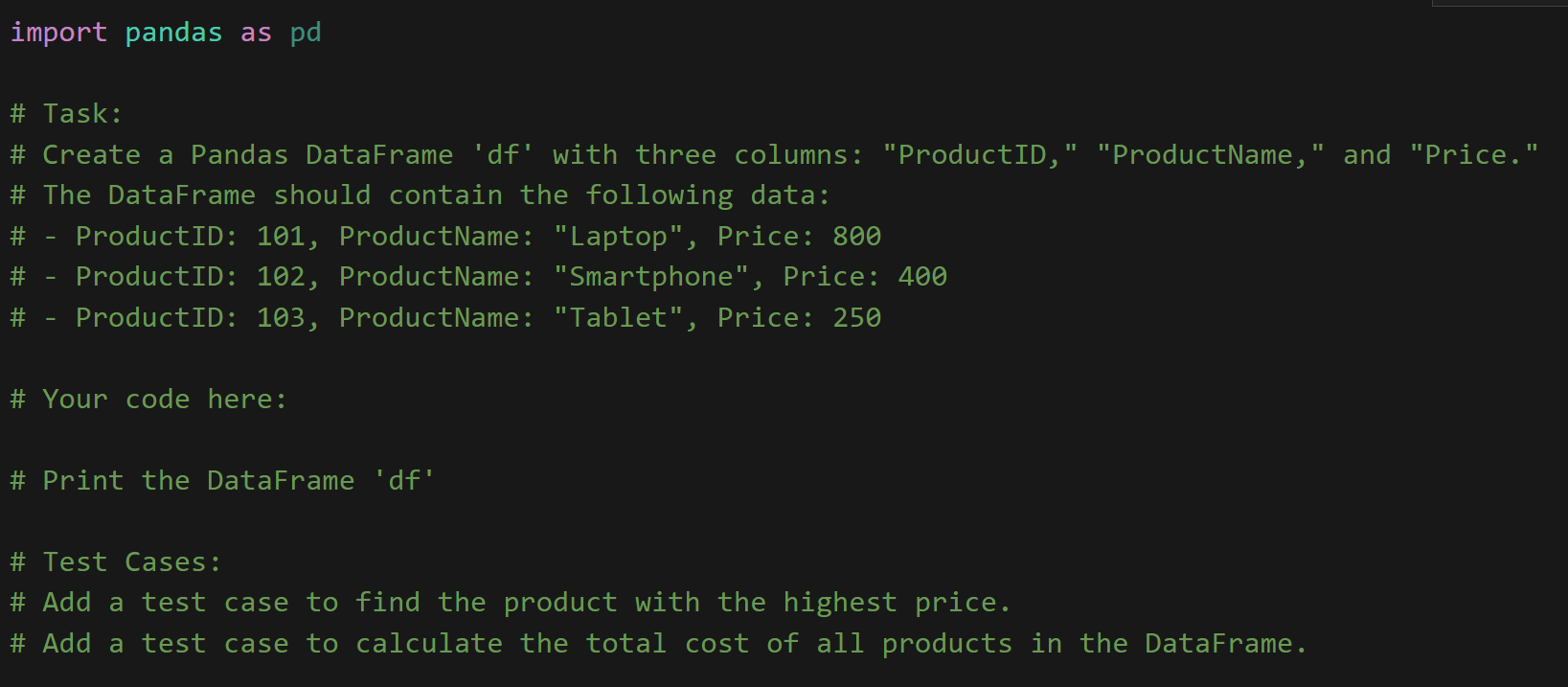
2



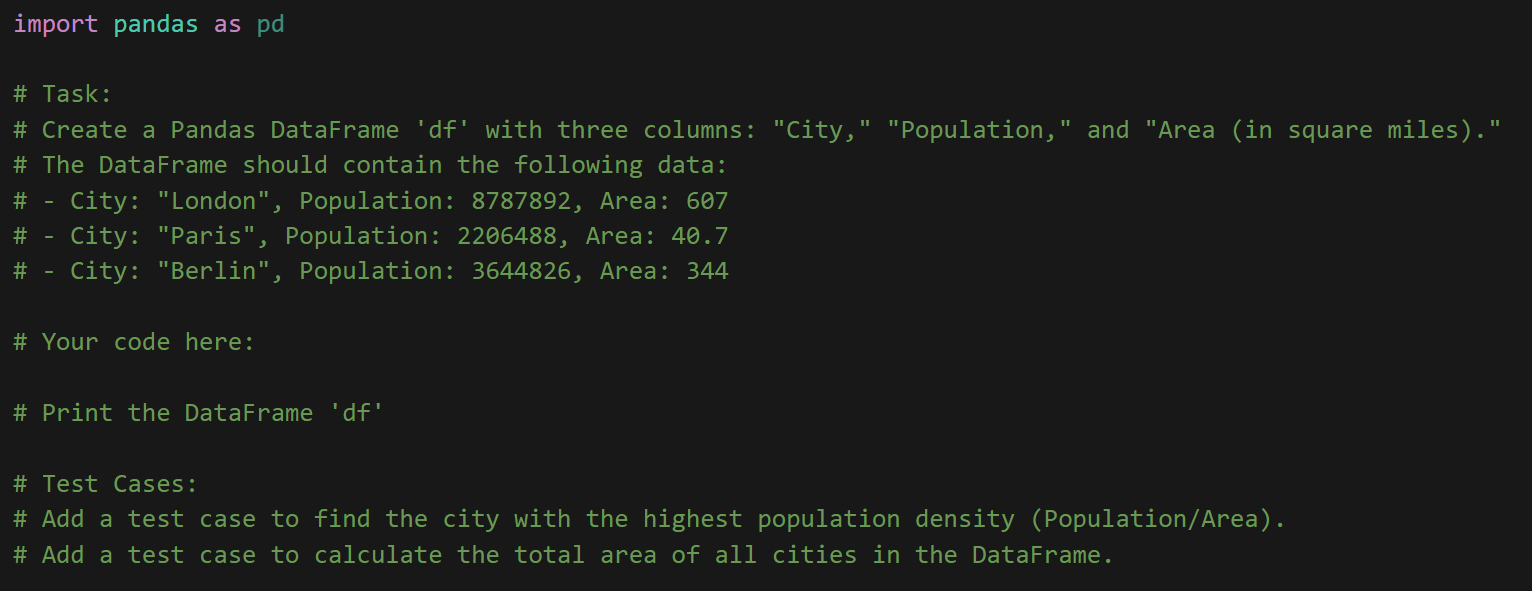
3



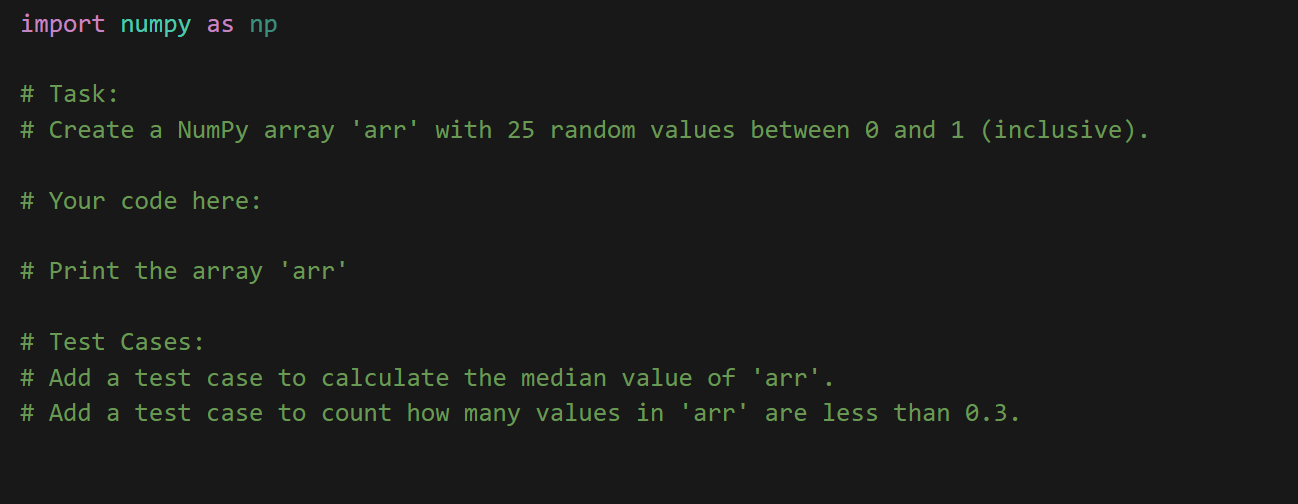
4



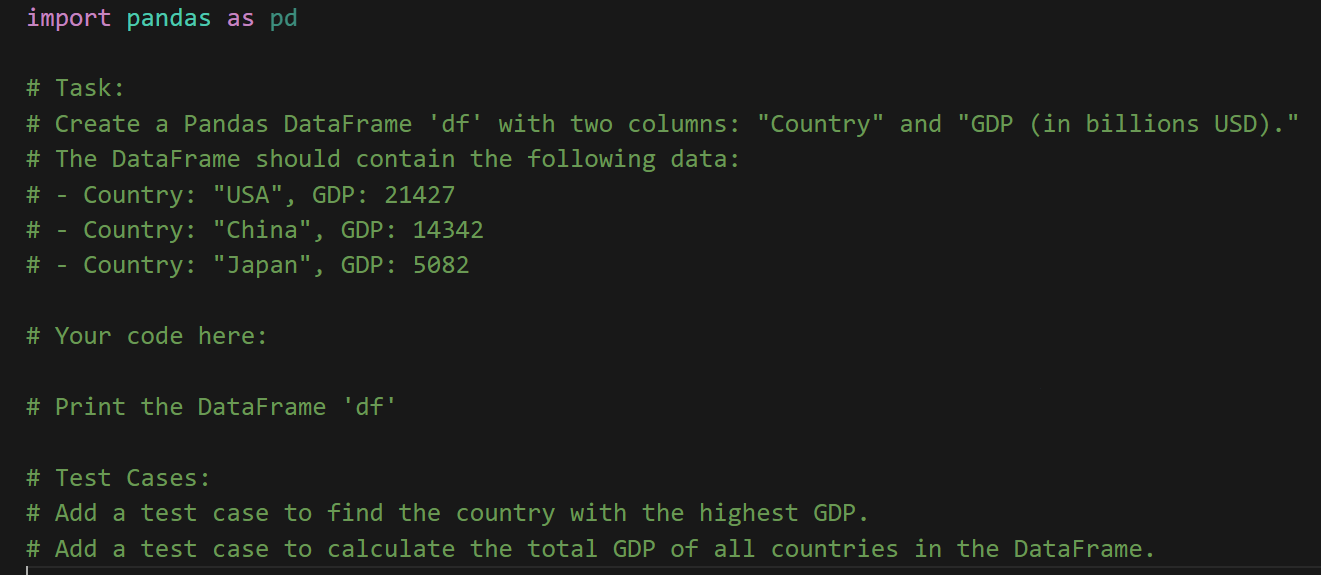
5



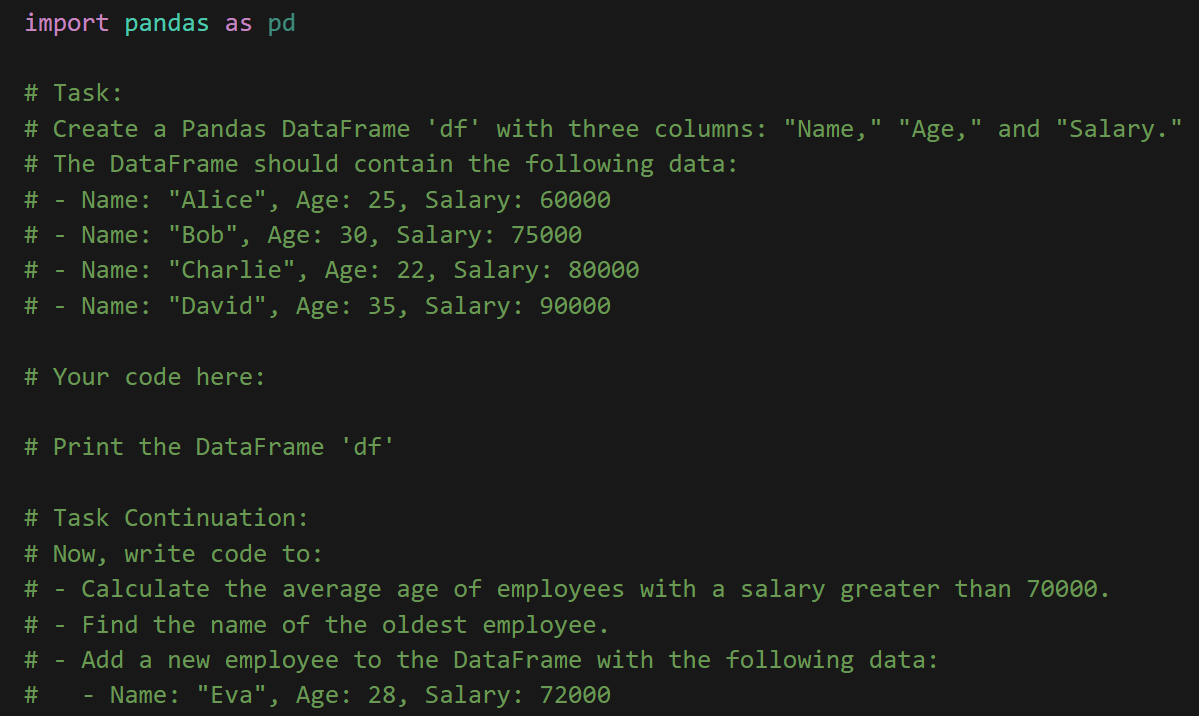
6



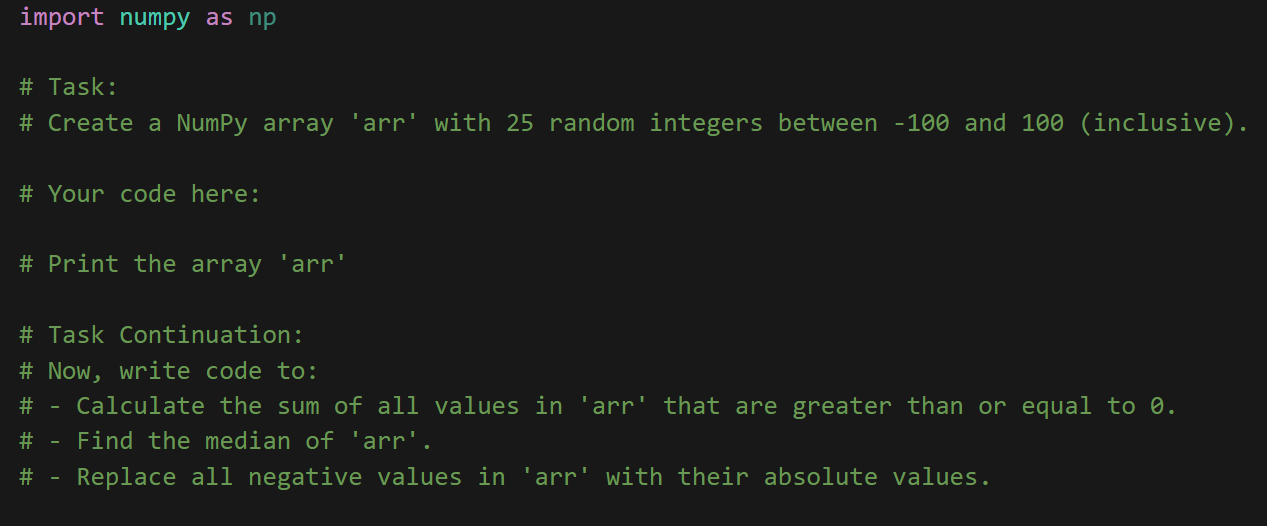
7



8



9



10

