LAB MANUAL

K. R. MANGALAM UNIVERSITY

B. TECH CSE (AI&ML)

SOET

(School of Engineering &Technology)

ENSP151- Clean Coding with Python Lab

Submitted by-

Name – Kunal Yadav

Roll no - 2401730157

Submitted to-

(Faculty Name)

Mr. Nikhil Sharma

Index

S No	Lab Assignment	Page No.
1	Write a Python script that demonstrates clean coding practices by organizing variables, naming conventions, and comments to improve readability.	1
2	Develop a Python script that implements and optimizes various data structures, like lists, dictionaries, and sets, to manage complex data workflows.	2
3	Construct a modular Python application that organizes code into reusable functions, demonstrating the benefits of modular programming.	3
4	Redesign an existing Python script, focusing on improving readability and performance through strategic refactoring techniques.	4
5	Develop a Python application that incorporates exception handling, allowing the program to handle unexpected inputs and errors gracefully.	5
6	Create a Python program using loops and conditional logic to simulate a real-world scenario (e.g., inventory management or banking), leveraging dictionaries and lists for effective data handling.	6-7
7	Write a Python script that reads data from a CSV file, processes it with pandas, and applies basic statistical functions (mean, median, mode) to generate insights.	8
8	Implement a Python script dedicated to advanced data preprocessing, cleaning, and transformation of real-world datasets using pandas.	9
9	Create a Python application to perform exploratory data analysis (EDA) on a dataset, identifying patterns and insights within the data.	10-11
10	Design a data visualization program using Matplotlib and Seaborn to illustrate trends and relationships within data, focusing on creating clear and meaningful plots.	12-13

Lab Assignment – 1

Write a Python script that demonstrates clean coding practices by organizing variables, naming conventions, and comments to improve readability.

Code-

```
#FIND SUM OF TWO NUMBERS
num_1=int(input("enter first number"))
num_2=int(input("enter second number"))

#logic for addition
sum=num_1+num_2

print(f"The sum of {num_1} and {num_2} is",sum)
```

Output-

```
The sum of 5 and 6 is 11
```

Lab Assignment – 2

Develop a Python script that implements and optimizes various data structures, like lists, dictionaries, and sets, to manage complex data workflows.

Code-

```
def main():
    #for list  operations
    print("--- List Operations ---")
    my_list = [int(input("Enter a number for the list: ")) for _ in range(3)]
    print("Initial List:", my_list)
    my_list.append(int(input("Enter a number to add to the list: ")))
    print("List after adding:", my_list)
    my_list.remove(int(input("Enter a number to remove from the list: ")))
    print("List after removing:", my_list)
    print("Accessing first element of the list:", my_list[0])
    #for dictionary operations
    print("--- Dictionary Operations ---")
    my_dict = {
        "name": input("Enter a name: "),
        "age": int(input("Enter an age: "))
    }
    print("Initial Dictionary:", my_dict)
    my_dict["city"] = input("Enter a city to add to the dictionary: ")
    print("Dictionary after adding city:", my_dict)
    del my_dict["age"]
    print("Dictionary after removing 'age':", my_dict)
    print("Value for 'name':", my_dict.get("name"))
    #for set operations
    print("--- Set Operations ---")
    my_set = {int(input("Enter a number for the set: ")) for _ in range(3)}
    print("Initial Set:", my_set)
    my_set.add(int(input("Enter a number to add to the set: ")))
    print("Set after adding:", my_set)
    my_set.discard(int(input("Enter a number to remove from the set: ")))
    print("Set after removing:", my_set)

main()
```

Output-

```
--- List Operations ---
Initial List: [4, 5, 6]
List after adding: [4, 5, 6, 7]
List after removing: [5, 6, 7]
Accessing first element of the list: 5
--- Dictionary Operations ---
Initial Dictionary: {'name': 'Kunal Yadav', 'age': 18}
Dictionary after adding city: {'name': 'Kunal Yadav', 'age': 18, 'city': 'Gurugram'}
Dictionary after removing 'age': {'name': 'Kunal Yadav', 'city': 'Gurugram'}
Value for 'name': Kunal Yadav
--- Set Operations ---
Initial Set: {20, 21, 22}
Set after adding: {20, 21, 22, 23}
Set after removing: {20, 22, 23}
```

Lab Assignment – 3

Construct a modular Python application that organizes code into reusable functions, demonstrating the benefits of modular programming.

Code-

```
# module for finding sum
def sum():
    num_1=int(input("enter first number"))
    num_2=int(input("enter second number"))

    sum=num_1+num_2

    print(f"The sum of {num_1} and {num_2} is",sum)

sum()
```

Output-

```
The sum of 5 and 6 is 11
```

Lab Assignment – 4

Redesign an existing Python script, focusing on improving readability and performance through strategic refactoring techniques.

Code-

```
# Function to get a number from the user
def get_number(number):
    return float(input(number))

# Function to calculate the sum of two numbers
def calculate_sum(num1, num2):
    return num1 + num2

# Function to display the result
def display_result(num1, num2, result):
    print(f"The sum of {num1} and {num2} is: {result}")

# Main function to organize the workflow
def main():
    # Input: Get two numbers from the user
    num1 = get_number("Enter the first number: ")
    num2 = get_number("Enter the second number: ")

    # Perform addition
    sum_result = calculate_sum(num1, num2)

    # Output: Display the result
    display_result(num1, num2, sum_result)

main()
```

Output-

```
The sum of 5.0 and 6.0 is: 11.0
```

Lab Assignment – 5

Develop a Python application that incorporates exception handling, allowing the program to handle unexpected inputs and errors gracefully.

Code-

```
try:
    num1 = int(input("Enter the first number:"))
    num2 = int(input("Enter the second number to devide from:"))
    division=num1/num2
    print(f"after dividing {num1} by {num2} the answer is {division}")
except:
    print("error:division by zero")
```

Output-

```
error:division by zero
```

Lab Assignment – 6

Create a Python program using loops and conditional logic to simulate a real-world scenario (e.g., inventory management or banking), leveraging dictionaries and lists for effective data handling.

Code-

```
Fruits_Shop = {
    "Apple": 500,
    "Banana": 1500,
    "Mango": 1000,
    "Guava": 300,
    "Watermelon": 200,
    "Orange": 1200,
    "Pomegranate": 1000,}
def display_menu():
    print("Welcome to Kunal Fruit Store")
    print("1. Check stock")
    print("2. Add stock")
    print("3. Remove stock")
    print("4. Exit")

fruit = input("Enter the fruit name to add stock: ")
while True:
  display_menu()
  choice = input("Choose an option: ")

  if choice == "1":
        print(f"Current Stock:",{Fruits_Shop[fruit]})

  elif choice == "2":
      if fruit in Fruits_Shop:
       quantity = int(input(f"Enter the quantity to add to {fruit}: "))
       Fruits_Shop[fruit] += quantity
       print(f"{quantity} units added to {fruit}. New stock: {Fruits_Shop[fruit]}")
  elif choice =="3":
      if fruit in Fruits_Shop:
       quantity = int(input(f"Enter the quantity to remove to {fruit}: "))
       Fruits_Shop[fruit] -= quantity
       print(f"{quantity} units removed to {fruit}. Left stock: {Fruits_Shop[fruit]}")
      else:
         print("Insufficient Stock")
```

```
elif choice == "4":
    print("Stock Data Updated")
    break
else:
   print("Invaild Input")
```

Output-

```
Welcome to Kunal Fruit Store
1. Check stock
2. Add stock
3. Remove stock
4. Exit
Current Stock: {500}
Welcome to Kunal Fruit Store
1. Check stock
2. Add stock
3. Remove stock
4. Exit
Stock Data Updated
```

Lab Assignment – 7

Write a Python script that reads data from a CSV file, processes it with pandas, and applies basic statistical functions (mean, median, mode) to generate insights.

Code-

```
import pandas as pd
# food wastage data
data = pd.read_csv("file:///C:/Users/Kunal/OneDrive/Documents/university%20work/clean%20code%20with%20python/project%20work/project/food_wastage_data.csv")
mean = data["Wastage Food Amount"].mean()
median = data["Wastage Food Amount"].median()
mode =data["Wastage Food Amount"].mode()[0]
print(f"Mean: {mean}, Median: {median}, Mode: {mode}")
```

Output-

```
Mean: 28.536475869809202, Median: 26.5, Mode: 20
```

	Type of Food	Number of Guests	Event Type	Quantity of Food	Storage Conditions	Purchase History	Seasonality	Preparation Method	Geographical Location	Pricing	Wastage Food Amount
0	Meat	310	Corporate	450	Refrigerated	Regular	All Seasons	Buffet	Urban	Low	25
1	Meat	400	Birthday	500	Room Temperature	Regular	Winter	Buffet	Suburban	High	40
2	Vegetables	302	Birthday	371	Refrigerated	Regular	Summer	Buffet	Suburban	Low	27
3	Meat	491	Birthday	497	Refrigerated	Regular	All Seasons	Finger Food	Rural	High	32
4	Meat	300	Corporate	400	Refrigerated	Regular	Winter	Finger Food	Urban	Moderate	25
...	...	...	...	...	...	...	...	...	...	...	...
1777	Baked Goods	310	Corporate	350	Room Temperature	Regular	Summer	Finger Food	Urban	High	35
1778	Baked Goods	284	Social Gathering	443	Room Temperature	Regular	Winter	Buffet	Rural	Low	32
1779	Fruits	220	Wedding	300	Room Temperature	Regular	All Seasons	Finger Food	Urban	Moderate	15
1780	Fruits	250	Wedding	350	Room Temperature	Regular	All Seasons	Finger Food	Rural	Moderate	20
1781	Baked Goods	400	Wedding	500	Room Temperature	Regular	Winter	Sit-down Dinner	Rural	High	45

1782 rows × 11 columns

Lab Assignment – 8

Implement a Python script dedicated to advanced data preprocessing, cleaning, and transformation of real-world datasets using pandas.

Code-

```
import pandas as pd

# Step 1: Create a dataset
data = {
    "Order ID": [101, 102, 103, 104, 105],
    "Customer Name": ["Kunal", "Devesh", None, "Vikash", "Kevin"],
    "Product": ["Laptop", "Laptop", "Tablet", "Mobile", "Mobile"],
    "Quantity": [2, 2, 3, 1, None],
    "Price Per Unit": [2000, 1300, 900, 800, 600],
}

# Load the data into a pandas DataFrame
df = pd.DataFrame(data)
df["Total Price"] = df["Quantity"] * df["Price Per Unit"]

df.fillna({"Customer Name":"Unknown","Quantity":"Unknown","Total Price":"Unknown"}, inplace=True)
df
```

Output-

	Order ID	Customer Name	Product	Quantity	Price Per Unit	Total Price
0	101	Kunal	Laptop	2.0	2000	4000.0
1	102	Devesh	Laptop	2.0	1300	2600.0
2	103	Unknown	Tablet	3.0	900	2700.0
3	104	Vikash	Mobile	1.0	800	800.0
4	105	Kevin	Mobile	Unknown	600	Unknown

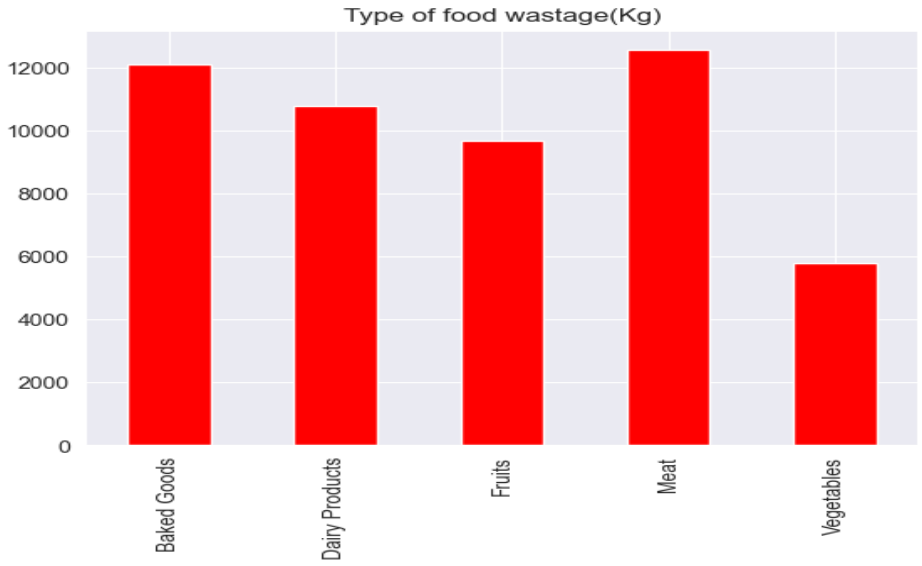
Lab Assignment – 9

Create a Python application to perform exploratory data analysis (EDA) on a dataset, identifying patterns and insights within the data.

Code-

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('C:/Users/Kunal/OneDrive/Documents/university work/clean code with python/project work/project/food_wastage_data.csv')

print(df)

bar=df.groupby('Type of Food')['Wastage Food Amount'].sum()
bar.plot.bar('Type of Food','Wastage Food Amount',color='red')
plt.title("Type of food wastage(Kg)")
```

Output-




```
     Type of Food  Number of Guests        Event Type  Quantity of Food  \
0            Meat               310         Corporate               450
1            Meat               400          Birthday               500
2      Vegetables               302          Birthday               371
3            Meat               491          Birthday               497
4            Meat               300         Corporate               400
...           ...               ...               ...               ...
1777  Baked Goods               310         Corporate               350
1778  Baked Goods               284  Social Gathering               443
1779       Fruits               220           Wedding               300
1780       Fruits               250           Wedding               350
1781  Baked Goods               400           Wedding               500

     Storage Conditions Purchase History  Seasonality Preparation Method  \
0          Refrigerated           Regular  All Seasons              Buffet
1      Room Temperature           Regular       Winter              Buffet
2          Refrigerated           Regular       Summer              Buffet
3          Refrigerated           Regular  All Seasons         Finger Food
4          Refrigerated           Regular       Winter         Finger Food
...                 ...               ...          ...                 ...
1777   Room Temperature           Regular       Summer         Finger Food
1778   Room Temperature           Regular       Winter              Buffet
1779   Room Temperature           Regular  All Seasons         Finger Food
1780   Room Temperature           Regular  All Seasons         Finger Food
1781   Room Temperature           Regular       Winter    Sit-down Dinner
...
1780                Rural  Moderate                  20
1781                Rural      High                  45

[1782 rows x 11 columns]
```

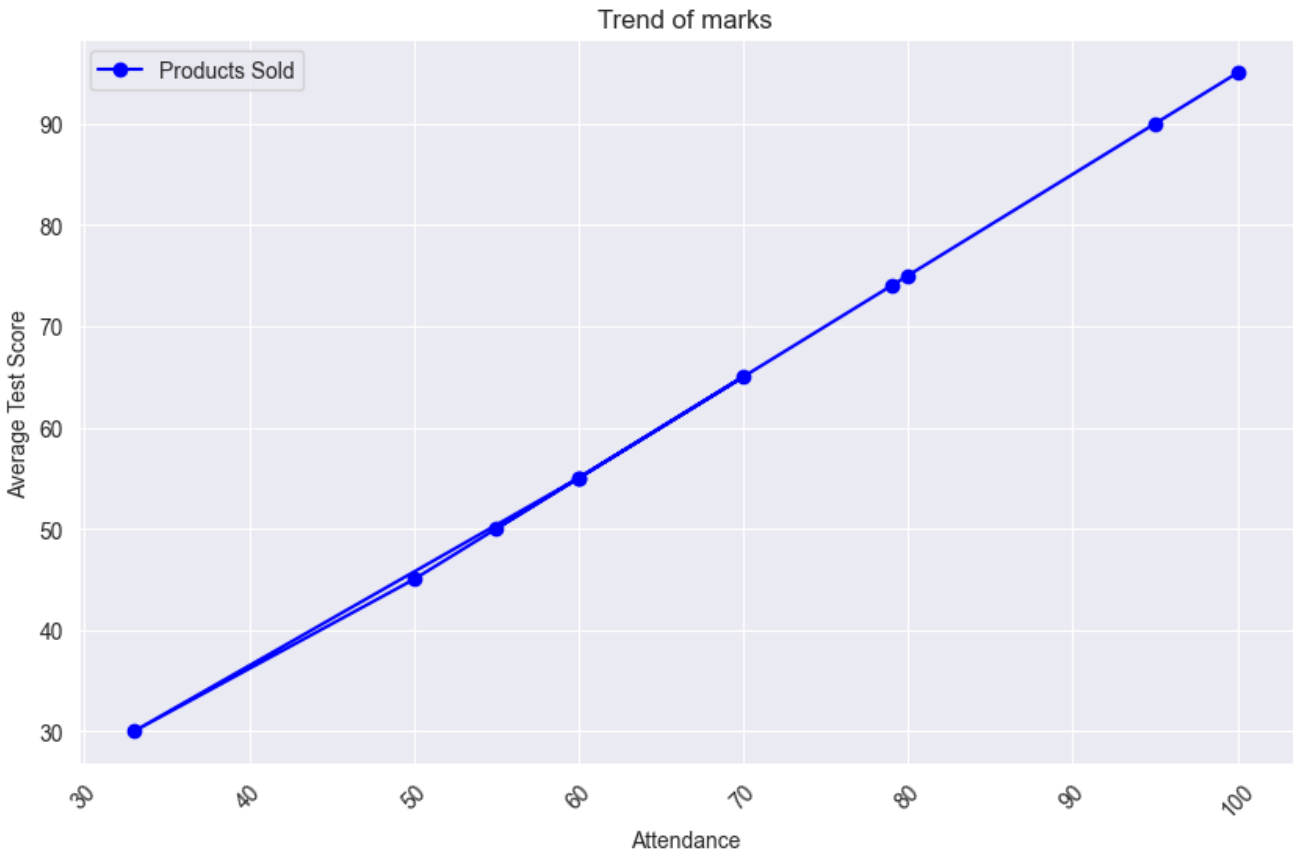
Lab Assignment – 10

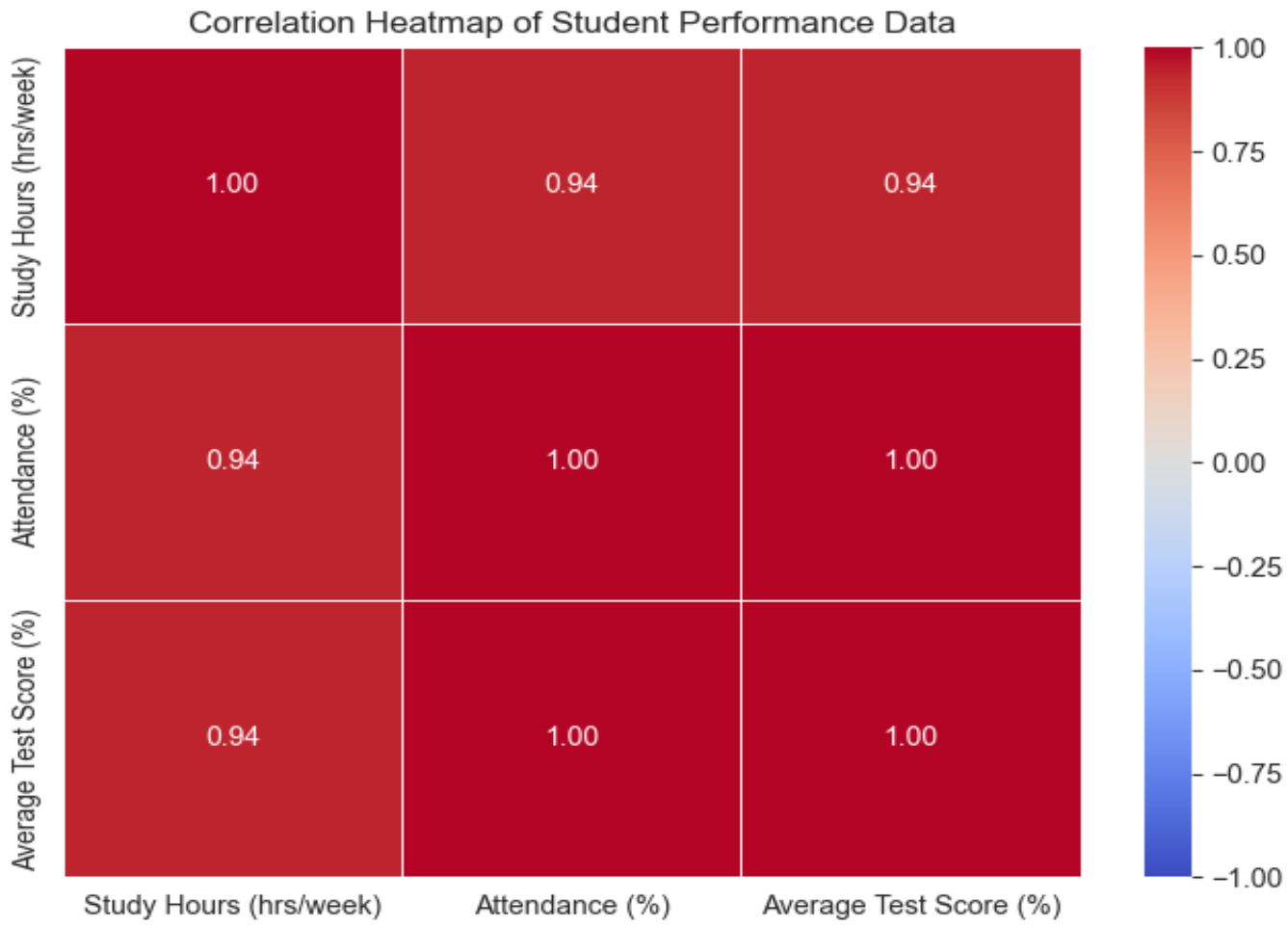
Design a data visualization program using Matplotlib and Seaborn to illustrate trends and relationships within data, focusing on creating clear and meaningful plots.

Code-

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd


data = {
    'Student ID': [101, 102, 103, 104, 105, 106, 107, 108, 109, 110],
    'Study Hours (hrs/week)': [40, 30, 25, 20, 13, 12, 2, 4, 9, 5],
    'Attendance (%)': [100, 95, 80, 79, 60, 60, 33, 50, 70, 55],
    'Average Test Score (%)': [95, 90, 75, 74, 55, 55, 30, 45, 65, 50],
    'Extracurricular Activities': ['Y', 'N', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'N']
}


df = pd.DataFrame(data)


Attendance=df['Attendance (%)']
Average_Test_Score=df['Average Test Score (%)']
plt.figure(figsize=(10, 6))
plt.plot(Attendance,Average_Test_Score, label='Products Sold', marker='o', color='blue')

plt.title('Trend of marks')
plt.xlabel('Attendance')
plt.ylabel('Average Test Score')
plt.xticks(rotation=45)
plt.legend()


corr_matrix =df[["Study Hours (hrs/week)", "Attendance (%)", "Average Test Score (%)"]].corr()


plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5, vmin=-1, vmax=1)
plt.title('Correlation Heatmap of Student Performance Data')
plt.show()
```

Output-


Trend of marks
Products Sold
Average Test Score
Attendance
30
40
50
60
70
80
90
100


Correlation Heatmap of Student Performance Data
Study Hours (hrs/week)
Attendance (%)
Average Test Score (%)
1.00
0.94
0.94
0.94
1.00
1.00
0.94
1.00
1.00
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00