# LIST OF EXPERIMENTS

1. Get an integer input from a user. If the number is odd, then find the factorial of a number and find the number of digits in the factorial of the number. If the number is even, then check the given number is palindrome or not.

number = int(input("Enter an integer: "))

**if** number % 2 == 1:

fact = 1 **for** i **in** range(2, number + 1): fact \*= i

digit\_count = 0 temp = fact **while** temp != 0: temp //= 10 digit\_count += 1

print("Factorial:", fact) print("Number of digits in factorial:", digit\_count)

**else**: num\_str = str(number) **if** num\_str == num\_str[::-1]: print("The number is a palindrome.") **else**: print("The number is not a palindrome.")

Enter an integer: 72

The number is not a palindrome.

1. Given two strings, PRINT (YES or NO) whether the second string can be obtained from the first by deletion of none, one or more characters.

*# Get user input*

first\_string = input("Enter the first string: ") second\_string = input("Enter the second string: ")

*# Convert the strings to lowercase for case-insensitive comparison* first\_string = first\_string.lower() second\_string = second\_string.lower()

index1 = 0 index2 = 0

**while** index1 < len(first\_string) **and** index2 < len(second\_string): **if** first\_string[index1] == second\_string[index2]: index1 += 1 index2 += 1

**else**: index1 += 1

*# Check if all characters in second\_string have been processed* **if** index2 == len(second\_string): print("YES") **else**: print("NO")

1. List and its operations

1. Programs for positive and negative indexing.
2. Program to check if the given list is in Ascending order or not.

*# a) Programs for positive and negative indexing.* my\_list = [1, 2, 3, 4, 5]

*# Positive indexing*

index = int(input("Enter the positive index: ")) **if** 0 <= index < len(my\_list): print("Positive indexing - Element at index", index, ":", my\_list[index]) **else**: print("Positive indexing - Index out of range")

*# Negative indexing*

index = int(input("Enter the negative index: ")) **if** -len(my\_list) <= index < 0: print("Negative indexing - Element at index", index, ":", my\_list[index]) **else**: print("Negative indexing - Index out of range")

Enter the positive index: 4

Positive indexing - Element at index 4 : 5

Enter the negative index: -3

Negative indexing - Element at index -3 : 3

*# b) Program to check if the given list is in Ascending order or not.*

**def** is\_ascending(lst): **for** i **in** range(1, len(lst)): **if** lst[i] < lst[i-1]:

**return** False

## return True

my\_list = [1, 2, 3, 4, 5] print(is\_ascending(my\_list))

True

Tuples and its operations

1. Python program to convert a tuple to a string.
2. Python program to reverse a tuple.

*# a) Python program to convert a tuple to a string.*

**def** tuple\_conv(tup): str = '' **for** item **in** tup: str = str + item

## return str

tup\_1 = ('p','y','t','h','o','n') str = tuple\_conv(tup\_1) print(str)

python

*# Python program to reverse a tuple.*

tup\_1 = (1, 2, 3, 4, 5) rev\_tup = tup\_1[::-1]

print("Original Tuple:", tup\_1) print("Reversed Tuple:", rev\_tup)

Original Tuple: (1, 2, 3, 4, 5)

Reversed Tuple: (5, 4, 3, 2, 1)

Sets and its operations:

Python program to check if a set is a subset of another set

*#Python program to check if a set is a subset of another set*

set\_1 = {1, 2, 3, 4, 5} set\_2 = {2, 4, 5}

**if** set\_2.issubset(set\_1): print("Set 2 is a subset of Set 1") **else**: print("Set 2 is not a subset of Set 1")

Set 2 is a subset of Set 1

Dictionaries and its operations:

Python program to iterate over dictionaries using for loops.

*# Python program to iterate over dictionaries using for loops.* dict\_1 = {'Name': 'Sanjith', 'Age': 18, 'Area': 'Tiruppur'}

**for** key **in** dict\_1:

value = dict\_1[key] print(key, ':', value)

Name : Sanjith

Age : 18

Area : Tiruppur

# Computations using NumPy functions

1. NumPy program to convert a list of numeric value into a one-dimensional NumPy array.
2. NumPy program to convert a list and tuple into arrays.

*# NumPy program to convert a list of numeric value into a onedimensional NumPy array.* **import** numpy **as** np

list\_1 = [1, 2, 3, 4, 5] np\_array = np.array(list\_1)

*# Print the original list and the NumPy array* print("Original List:", list\_1) print("\nNumPy Array:", np\_array)

Original List: [1, 2, 3, 4, 5]

NumPy Array: [1 2 3 4 5]

*# NumPy program to convert a list and tuple into arrays.*

**import** numpy **as** np

list\_2 = [1, 2, 3, 4, 5] np\_list = np.array(list\_2)

tup\_2 = (6, 7, 8, 9, 10) np\_tuple = np.array(tup\_2)

*# Print the original list, list array, tuple, and tuple array* print("Original List:", list\_2) print("\nList Array:", np\_list) print("\nOriginal Tuple:", tup\_2) print("\nTuple Array:", np\_tuple)

Original List: [1, 2, 3, 4, 5]

List Array: [1 2 3 4 5]

Original Tuple: (6, 7, 8, 9, 10)

Tuple Array: [ 6 7 8 9 10]

# Data manipulations using Pandas

1. Program to convert a NumPy array and series to data frames.
2. Program to add, subtract, multiple and divide two Pandas Series.
3. Program to retrieve and manipulate data using data frames.

*# Program to convert a NumPy array and series to data frames.* **import** pandas **as** pd

np\_array = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) df\_from\_array = pd.DataFrame(np\_array, columns=['a', 'b', 'c'])

series\_data = pd.Series([10, 20, 30, 40, 50]) df\_from\_series = pd.DataFrame(series\_data, columns=['Values'])

print("NumPy array:\n", np\_array)

print("\nDataFrame from NumPy array:\n", df\_from\_array) print("\nSeries:\n", series\_data)

print("\nDataFrame from Series:", df\_from\_series)

NumPy array:

[[1 2 3]

[4 5 6]

[7 8 9]]

DataFrame from NumPy array:

a b c 0 1 2 3

1. 4 5 6
2. 7 8 9

Series:

1. 10
2. 20
3. 30
4. 40
5. 50

dtype: int64

DataFrame from Series: Values

1. 10
2. 20
3. 30
4. 40
5. 50

*# Program to add, subtract, multiple and divide two Pandas Series.*

series1 = pd.Series([1, 2, 3, 4, 5]) series2 = pd.Series([6, 7, 8, 9, 10])

addition = series1 + series2 subtraction = series1 - series2 multiplication = series1 \* series2 division = series1 / series2

print("\nAddition:\n", addition) print("\nSubtraction:\n", subtraction) print("\nMultiplication:\n", multiplication) print("\nDivision:\n", division)

Addition:

1. 7
2. 9
3. 11
4. 13 4 15 dtype: int64

Subtraction:

1. -5
2. -5
3. -5
4. -5 4 -5 dtype: int64

Multiplication:

1. 6
2. 14
3. 24
4. 36 4 50 dtype: int64

Division:

1. 0.166667
2. 0.285714
3. 0.375000
4. 0.444444 4 0.500000 dtype: float64

*# Program to retrieve and manipulate data using data frames* **import** pandas **as** pd

data = {

'Name': ['John', 'Emily', 'Michael', 'Jessica'],

'Age': [25, 28, 32, 30],

'City': ['New York', 'Los Angeles', 'Chicago', 'Houston']

}

df = pd.DataFrame(data)

print("Original DataFrame:") print(df)

*# Accessing specific columns* names = df['Name'] ages = df['Age'] cities = df['City']

print("\nRetrieved Columns:") print("Names:", names) print("Ages:", ages) print("Cities:", cities)

*# Accessing specific rows* first\_row = df.loc[0] second\_row = df.loc[1] last\_row = df.loc[df.shape[0] - 1]

print("\nRetrieved Rows:") print("First Row:\n", first\_row) print("Second Row:\n", second\_row) print("Last Row:\n", last\_row)

*# Manipulating data in the DataFrame # Adding a new column* df['Salary'] = [50000, 60000, 70000, 55000]

*# Modifying values in a column* df['Age'] = df['Age'] + 1

*# Deleting a column* df.drop('City', axis=1, inplace=True)

print("\nModified DataFrame:") print(df)

Original DataFrame:

Name Age City

0 John 25 New York 1 Emily 28 Los Angeles

1. Michael 32 Chicago
2. Jessica 30 Houston

Retrieved Columns:

Names: 0 John

1 Emily 2 Michael

3 Jessica

Name: Name, dtype: object

Ages: 0 25

1. 28
2. 32
3. 30

Name: Age, dtype: int64

Cities: 0 New York

1. Los Angeles
2. Chicago
3. Houston

Name: City, dtype: object

Retrieved Rows:

First Row:

Name John

Age 25

City New York

Name: 0, dtype: object

Second Row:

Name Emily

Age 28 City Los Angeles Name: 1, dtype: object

Last Row:

Name Jessica

Age 30 City Houston

Name: 3, dtype: object

Modified DataFrame:

Name Age Salary

1. John 26 50000
2. Emily 29 60000
3. Michael 33 70000
4. Jessica 31 55000