

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:01

GREATEST OF THREE NUMBERS

AIM:

To write a python program to read three numbers and print the greatest of three numbers.

ALGORITHM:

- 1.Start the program.
- 2.Read the first number.
- 3.Read the second number.
- 4.Read the third number.
- 5.Compare first and second numbers.
- 6.Compare the largest with the third number.
- 7.Check if first number is the greatest.
- 8.If true, assign it to big.
- 9.Else check if second number is the greatest.
- 10.Assign the greatest value to big.
- 11.Print the greatest number.
- 12.Stop the program.

PROGRAM:

```
n1 = int(input("Enter the first number: "))
n2 = int(input("Enter the second number: "))
n3 = int(input("Enter the third number: "))
if n1 >= n2 and n1 >= n3:
    big = n1
elif n2 >= n1 and n2 >= n3:
    big = n2
else:
    big = n3
print("The greatest of the three numbers is:", big)
```

RESULT:

Thus, the Python program to find the greatest of three numbers was executed successfully and the output was obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:02

USING RANGE () FUNCTION

AIM:

To write a python program to find the sum of N number using range () function in for loop.

ALGORITHM:

- 1.Start the program.
- 2.Read the value of N.
- 3.Initialize sum as 0.
- 4.Start a for loop from 1 to N.
- 5.Add current number to sum.
- 6.Repeat loop until N.
- 7.Exit the loop.
- 8.Store final sum.
- 9.Display the sum.
- 10.Check output correctness.
- 11.End loop execution.
- 12.Stop the program.

PROGRAM:

```
N = int(input("Enter a positive integer N: "))  
total = 0  
for i in range(1, N + 1):  
    total += i  
print("The sum of the first", N, "numbers is:", total)
```

RESULT:

Thus, the above Python program is executed and the output is obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:03

USING STRING

AIM:

To Write a python program to demonstrate the string slicing, concatenation, replication and len() method.

ALGORITHM:

- 1.Start the program.
- 2.Read a string.
- 3.Read start index.
- 4.Read end index.
- 5.Perform string slicing.
- 6.Display sliced string.
- 7.Read first string.
- 8.Read second string.
- 9.Concatenate both strings.
- 10.Replicate a string.
- 11.Find string length.
- 12.Stop the program.

PROGRAM:

```
s = input("Enter a string: ")
start = int(input("Enter the start index: "))
end = int(input("Enter the end index: "))
print("Sliced string:", s[start:end])
first = input("Enter first string: ")
second = input("Enter second string: ")
print("Concatenated string:", first + " " + second)
rep = input("Enter string to replicate: ")
count = int(input("Enter count: "))
print("Replicated string:", rep * count)
print("Length of the string:", len(s))
```

RESULT:

Thus, the above Python program is executed and the output is obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:04

USING TUPLE AND LIST

AIM:

To write a python program to create a tuple and convert into a list and print the list in sorted order.

ALGORITHM:

- 1.Start the program.
- 2.Read input elements.
- 3.Convert input to tuple.
- 4.Display tuple.
- 5.Convert tuple to list.
- 6.Display list.
- 7.Sort the list.
- 8.Store sorted list.
- 9.Print sorted list.
- 10.Verify sorting.
- 11.End process.
- 12.Stop the program.

PROGRAM:

```
s = input("Enter elements separated by commas: ")
t = tuple(map(int, s.split(",")))
print("Tuple:", t)
li = list(t)
li.sort()
print("Sorted list:", li)
```

RESULT:

Thus, the above Python program is executed and the output is obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:05

USING DICTIONARY

AIM:

To write a python program to create a dictionary and check whether a key or value exist in the dictionary.

ALGORITHM:

- 1.Start the program.
- 2.Create empty dictionary
- 3.Read number of pairs
- 4.Read key
- 5.Read value
- 6.Store key-value pair
- 7.Repeat for all inputs
- 8.Display dictionary
- 9.Read key to check
- 10.Check key existence
- 11.Check value existence
- 12.Stop the program.

PROGRAM:

```
d = {}  
n = int(input("Enter number of key-value pairs: "))  
for i in range(n):  
    key = input("Enter key: ")  
    value = input("Enter value: ")  
    d[key] = value  
keycheck = input("Enter key to check: ")  
print("Key exists" if keycheck in d else "Key does not exist")  
valuecheck = input("Enter value to check: ")  
print("Value exists" if valuecheck in d.values() else "Value does not exist")
```

RESULT:

Thus, the above Python program is executed and the output is obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:06

NUMPY – ONE DIMENSIONAL ARRAY & RESHAPE()

AIM:

To write a python program to create one dimensional array and convert into a 2D-dimensional array using reshape(), print the first two columns alone using slicing.

ALGORITHM:

- 1.Start the program.
- 2.Import NumPy
- 3.Read number of elements
- 4.Read elements
- 5.Create 1D array
- 6.Read row size
- 7.Calculate column size
- 8.Reshape array
- 9.Display 2D array
- 10.Print first two columns
- 11.Verify output
- 12.Stop the program.

PROGRAM:

```
import numpy as np
n = int(input("Enter number of elements: "))
a = [int(input()) for i in range(n)]
a = np.array(a)
row = int(input("Enter rows: "))
a2 = a.reshape(row, n//row)
print(a2)
print(a2[:, :2])
```

RESULT:

Thus, the above Python program is executed and the output is obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:07

NUMPY – TWO DIMENSIONAL ARRAY & WHERE()

AIM:

To write a python program to create two-dimensional array and search for an element using where () function.

ALGORITHM:

- 1.Start the program.
- 2.Import NumPy
- 3.Read rows and columns
- 4.Read array elements
- 5.Create 2D array
- 6.Display array
- 7.Read search element
- 8.Apply where()
- 9.Check result
- 10.Display positions
- 11.Handle not found case
- 12.Stop the program.

PROGRAM:

```
import numpy as np
a = np.array([[1,2],[3,4]])
search = int(input("Enter element: "))
pos = np.where(a == search)
print(pos)
```

RESULT:

Thus, the above Python program is executed and the output is obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:08

NUMPY – AGGREGATE FUNCTIONS

AIM:

To write a python program to create a 2D-dimensional array and demonstrate aggregation functions sum (), min () and max () in the row and column wise.

ALGORITHM:

- 1.Start the program.
- 2.Import NumPy
- 3.Read rows and columns
- 4.Read elements
- 5.Create array
- 6.Display array
- 7.Find row sum
- 8.Find column sum
- 9.Find row min
- 10.Find column max
- 11.Display results
- 12.Stop the program.

PROGRAM:

```
import numpy as np
a = np.array([[1,2],[3,4]])
print(np.sum(a, axis=1))
print(np.sum(a, axis=0))
print(np.min(a, axis=1))
print(np.max(a, axis=0))
```

RESULT:

Thus, the above Python program is executed and the output is obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:09

FILE HANDLING

AIM:

To write a python program to read a text file and write the content in another file.

ALGORITHM:

- 1.Start the program.
- 2.Open input file.
- 3.Read contents.
- 4.Close input file.
- 5.Open output file.
- 6.Write contents.
- 7.Save file.
- 8.Close output file.
- 9.Display message.
- 10.Verify file copy.
- 11.End execution.
- 12.Stop the program.

PROGRAM:

```
infile = open("input.txt","r")  
content = infile.read()  
infile.close()  
outfile = open("output.txt","w")  
outfile.write(content)  
outfile.close()
```

RESULT:

Thus, the above Python program is executed and the output is obtained.

PYTHON PROGRAMMING

LAB EXERCISES

EX.NO:10

USING PANDAS

AIM:

To write a python program to read a csv file using pandas and print the content.

ALGORITHM:

- 1.Start the program.
- 2.Import pandas.
- 3.Specify CSV file name.
- 4.Read CSV file.
- 5.Store data in DataFrame.
- 6.Display DataFrame.
- 7.Check data format.
- 8.Verify columns.
- 9.Verify rows.
- 10.Print output.
- 11.End execution.
- 12.Stop the program.

PROGRAM:

```
import pandas as pd
csv_file='data.csv'
df = pd.read_csv("data.csv")
print(df)
```

RESULT:

Thus, the above Python program is executed and the output is obtained.