

Jagannath Gupta Institute of Engineering and
Technology

LAB MANUAL

Python Programming Lab-6CS4-23

Program 1:-Write a program to demonstrate basic data type in python.

```
# Python program to
# demonstrate numeric value

a = 5
print("Type of a: ", type(a))

b = 5.0
print("\nType of b: ", type(b))

c = 2 + 4j
print("\nType of c: ", type(c))
```

Output:

Type of a: <class 'int'>

Type of b: <class 'float'>

Type of c: <class 'complex'>

Program2:- a program to compute distance between two points taking input from the user Write a program add.py that takes 2 numbers as command line arguments and prints its sum.

Solution:-

Distance can be calculated using the two points (x_1, y_1) and (x_2, y_2) , the distance d between these points is given by the formula:

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

for e.g : let $x_1, y_1=10,9$ and $x_2, y_2=4,1$ then $(x_2-x_1)^2=(10-4)^2=6^2=36$ and $(y_2-y_1)^2=(9-1)^2=8^2=64$ now $64+36=100$ and 100 is square root of 10 sp distance between (10,9) and (4,1) is 10

Code : -

```
x1=int(input("enter x1 : "))
x2=int(input("enter x2 : "))
y1=int(input("enter y1 : "))
y2=int(input("enter y2 : "))
result= (((x2 - x1)**2) + ((y2-y1)**2) )**0.5
print("distance between",(x1,x2),"and",(y1,y2),"is : ",result)
```

Program-3 Write a Program for checking whether the given number is an even number or not.
Using a for loop.

Solution:

Python program to check if the input number is odd or even.

A number is even if division by 2 gives a remainder of 0.

If the remainder is 1, it is an odd number.

```
num = int(input("Enter a number: "))
```

```
if (num % 2) == 0:
```

```
print("{0} is Even".format(num))

else:

    print("{0} is Odd".format(num))
```

Program 4:-Write a Program to demonstrate list and tuple in python. Write a program using a for loop that loops over a sequence. Write a program using a while loop that asks the user for a number, and prints a countdown from that number to zero.

Solution:-

```
# Python program to create a list of tuples
# from given list having number and
# its cube in each tuple

# creating a list
list1 = [1, 2, 5, 6]

# using list comprehension to iterate each
# values in list and create a tuple as specified
res = [(val, pow(val, 3)) for val in list1]

# print the result
print(res)
```

Output:

```
[(1, 1), (2, 8), (5, 125), (6, 216)]
```

Write a program using a while loop that asks the user for a number, and prints a countdown from that number to zero.

```
n=int(input("Enter n value:"))

for i in range(n,-1,-1):

    print(i)
```

Program 5:-Find the sum of all the primes below two million. By considering the terms in the Fibonacci sequence whose values do not exceed four million, WAP to find the sum of the even-valued terms.

```
primes_below_number = 2000000 # number to find summation of all primes below number
```

```

numbers = (range(1, primes_below_number + 1, 2)) # creates a list excluding even numbers

pos = 0 # index position

sum_of_primes = 0 # total sum

number = numbers[pos]

while number < primes_below_number and pos < len(numbers) - 1:

    pos += 1

    number = numbers[pos] # moves to next prime in list numbers

    sum_of_primes += number # adds prime to total sum

    num = number

    while num < primes_below_number:

        num += number

        if num in numbers[:]:

            numbers.remove(num) # removes multiples of prime found

print sum_of_primes + 2

```

Program 6:-Write a program to count the numbers of characters in the string and store them in a dictionary data structure Write a program to use split and join methods in the string and trace a birthday of a person with a dictionary data structure

```

Code:-
str=input("enter string : ")
f = {}
for i in str:
    if i in f:
        f[i] += 1
    else:
        f[i] = 1
print(f)

```

Write a program to use split and join methods in the string and trace a birthday of a person with a dictionary data structure

```
split  
Str.split()  
join  
Str1.join(str2)
```

Algorithm

Step 1: Input a string.
Step 2: here we use split method for splitting and for joining use join function.
Step 3: display output.

Example code

```
#split of string  
  
str1=input("Enter first String with space :: ")  
print(str1.split())      #splits at space  
  
  
str2=input("Enter second String with (,) :: ")  
print(str2.split(','))    #splits at ','  
  
  
str3=input("Enter third String with (:) :: ")  
print(str3.split(':'))    #splits at ':'  
  
  
str4=input("Enter fourth String with (;) :: ")  
print(str4.split(';'))    #splits at ';'   
  
  
str5=input("Enter fifth String without space :: ")  
print([str5[i:i+2] for i in range(0,len(str5),2)])    #splits at  
position 2
```

Output

```
Enter first String with space :: python program
['python', 'program']
Enter second String with (,) :: python, program
['python', 'program']
Enter third String with (:) :: python: program
['python', 'program']
Enter fourth String with (;) :: python; program
['python', 'program']
Enter fifth String without space :: python program
['py', 'th', 'on', 'pr', 'og', 'ra', 'm']
```

Example Code

```
#string joining

str1=input("Enter first String  :: ")
str2=input("Enter second String  :: ")

str=str2.join(str1)      #each character of str1 is concatenated to
the #front of str2

print("AFTER JOINING OF TWO STRING ::>",str)
```

Output

```
Enter first String  :: AAA
Enter second String  :: BBB
AFTER JOINING OF TWO STRING ::>ABBBABBBA
```

Program7:-

Write a program to count frequency of characters in a given file. Can you use character frequency to tell whether the given file is a Python program file, C program file or a text file? Write a program to count frequency of characters in a given file. Can you use character frequency to tell whether the given file is a Python program file, C program file or a text file?

```
str=input("enter string : ")
f = {}
for i in str:
    if i in f:
        f[i] += 1
    else:
        f[i] = 1
print(f)
```

? Write a program to count frequency of characters in a given file.

Here is source code of the Python Program to count the occurrences of a letter in a text file. The program output is also shown below.

```
fname = input("Enter file name: ")
l=input("Enter letter to be searched:")
k = 0

with open(fname, 'r') as f:
    for line in f:
        words = line.split()
        for i in words:
            for letter in i:
                if(letter==l):
                    k=k+1
print("Occurrences of the letter:")
print(k)
```

Program Explanation

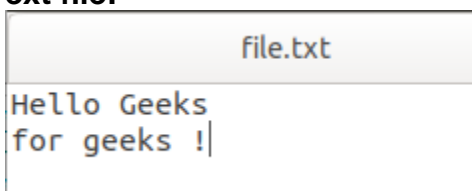
1. User must enter a file name and the letter to be searched.
2. The file is opened using the open() function in the read mode.
3. A for loop is used to read through each line in the file.
4. Each line is split into a list of words using split().
5. A for loop is used to traverse through the words list and another for loop is used to traverse through the letters in the word.
6. If the letter provided by the user and the letter encountered over iteration are equal, the letter count is incremented.
7. The final count of occurrences of the letter is printed.

Program 8 :-Write a program to print each line of a file in reverse order. Write a program to compute the number of characters, words and lines in a file.

Given a text file. The task is to reverse as well as stores the content from an input file to an output file.

This reversing can be performed in two types.

- **Full reversing:** In this type of reversing all the content get reversed.
- **Word to word reversing:** In this kind of reversing the last word comes first and the first word goes to the last position.
- **ext file:**



```
file.txt
Hello Geeks
for geeks !
```

-
- filter_none
- brightness_4


```

# Open the file in write mode
f1 = open("output1.txt", "w")

# Open the input file and get
# the content into a variable data
with open("file.txt", "r") as myfile:
    data = myfile.read()

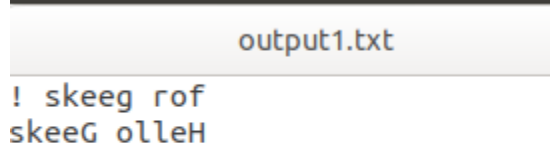
# For Full Reversing we will store the
# value of data into new variable data_1
# in a reverse order using [start: end: step],
# where step when passed -1 will reverse
# the string
data_1 = data[::-1]

# Now we will write the fully reverse
# data in the output1 file using
# following command
f1.write(data_1)

f1.close()

```

- **Output:**



output1.txt

```

! skeeg rof
skeeG olleH

```

-

Write a program to compute the number of characters, words and lines in a file.

```

import sys

fname = sys.argv[1]
lines = 0
words = 0
letters = 0

for line in open(fname):
    lines += 1
    letters += len(line)

    pos = 'out'
    for letter in line:
        if letter != ' ' and pos == 'out':
            words += 1
            pos = 'in'
        elif letter == ' ':
            pos = 'out'

print("Lines:", lines)
print("Words:", words)

```

```
print("Letters:", letters)
```

Program 9:-Write function to compute gcd, lcm of two numbers.

Python program to find LCM of two numbers

```
# Python 3 program to find  
# LCM of 2 numbers without  
# using GCD  
import sys
```

```
# Function to return  
# LCM of two numbers  
def findLCM(a, b):
```

```
    lar = max(a, b)  
    small = min(a, b)  
    i = lar  
    while(1) :  
        if (i % small == 0):  
            return i  
        i += lar
```

```
# Driver Code
```

```
a = 5  
b = 7  
print("LCM of " , a , " and "  
      , b , " is " ,  
      findLCM(a, b), sep = "")
```

LCM of 15 and 20 is 60

Recursive function to return gcd of a and b

```
def gcd(a,b):
```

```
    # Everything divides 0  
    if (a == 0):  
        return b  
    if (b == 0):  
        return a
```

```
    # base case  
    if (a == b):  
        return a
```

```
    # a is greater  
    if (a > b):  
        return gcd(a-b, b)  
    return gcd(a, b-a)
```

```
# Driver program to test above function
```

```

a = 98
b = 56
if(gcd(a, b)):
    print('GCD of', a, 'and', b, 'is', gcd(a, b))
else:
    print('not found')

```

Output:

```
GCD of 98 and 56 is 14
```

Program 10 :-Write a program to implement Merge sort. Write a program to implement Selection sort, Insertion sort

Python program for implementation of MergeSort

```

# Merges two subarrays of arr[].
# First subarray is arr[l..m]
# Second subarray is arr[m+1..r]
def merge(arr, l, m, r):
    n1 = m - l + 1
    n2 = r - m

    # create temp arrays
    L = [0] * (n1)
    R = [0] * (n2)

    # Copy data to temp arrays L[] and R[]
    for i in range(0 , n1):
        L[i] = arr[l + i]

    for j in range(0 , n2):
        R[j] = arr[m + 1 + j]

    # Merge the temp arrays back into arr[l..r]
    i = 0      # Initial index of first subarray
    j = 0      # Initial index of second subarray
    k = l      # Initial index of merged subarray

    while i < n1 and j < n2 :
        if L[i] <= R[j]:
            arr[k] = L[i]
            i += 1
        else:
            arr[k] = R[j]
            j += 1
        k += 1

```

```

# Copy the remaining elements of L[], if there
# are any
while i < n1:
    arr[k] = L[i]
    i += 1
    k += 1

# Copy the remaining elements of R[], if there
# are any
while j < n2:
    arr[k] = R[j]
    j += 1
    k += 1

# l is for left index and r is right index of the
# sub-array of arr to be sorted
def mergeSort(arr,l,r):
    if l < r:

        # Same as (l+r)//2, but avoids overflow for
        # large l and h
        m = (l+(r-1))//2

        # Sort first and second halves
        mergeSort(arr, l, m)
        mergeSort(arr, m+1, r)
        merge(arr, l, m, r)

# Driver code to test above
arr = [12, 11, 13, 5, 6, 7]
n = len(arr)
print ("Given array is")
for i in range(n):
    print ("%d" %arr[i]),

mergeSort(arr,0,n-1)
print ("\n\nSorted array is")
for i in range(n):
    print ("%d" %arr[i]),

```

Output:

Given array is

12 11 13 5 6 7

Sorted array is

5 6 7 11 12 13

```

# Python program for implementation of Selection
# Sort
import sys
A = [64, 25, 12, 22, 11]

# Traverse through all array elements
for i in range(len(A)):

    # Find the minimum element in remaining
    # unsorted array
    min_idx = i
    for j in range(i+1, len(A)):
        if A[min_idx] > A[j]:
            min_idx = j

    # Swap the found minimum element with
    # the first element
    A[i], A[min_idx] = A[min_idx], A[i]

# Driver code to test above
print ("Sorted array")
for i in range(len(A)):
    print("%d" %A[i]),

# Python program for implementation of Insertion Sort

# Function to do insertion sort
def insertionSort(arr):

    # Traverse through 1 to len(arr)
    for i in range(1, len(arr)):

        key = arr[i]

        # Move elements of arr[0..i-1], that are
        # greater than key, to one position ahead
        # of their current position
        j = i-1
        while j >= 0 and key < arr[j] :
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key

# Driver code to test above
arr = [12, 11, 13, 5, 6]
insertionSort(arr)
for i in range(len(arr)):
    print ("% d" % arr[i])

```

Output:

5 6 11 12 13