LAB CYCLE -1

Experiment No : 1

Date :07/10/24

Aim :

Write a program that prompt the user to enter his first and last name and display the message "Greetings first name, last name".

Pseudocode :

1. READ first name
2. READ last name
3. PRINT "Greetings first name last name !"

Method:

Function	Description	Syntax
input()	Allow user input (Returns a string value)	input(prompt)
print()	Prints the specified message to the screen	print(object(s))

Source Code :

```
first_name=input("Enter your first name:")
last_name=input("Enter your last name:")
print(f"Greetings {first_name} {last_name}!")
```

Output :

```
Enter your first name:Shifana
Enter your last name:Shirin
Greetings Shifana Shirin!
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 2

Date : 07/10/24

Aim :

Write a program to demonstrate different numeric data type in python.

Pseudocode :

1. SET int_num=10 , float=2.5 and complex_num=2+3j
2. PRINT “Integer:” , int_num , ”Type:” , type(int_num)
4. PRINT “Float:” , float , ”Type:” , type(float)
6. PRINT “Complex Number:” , complex_num , ”Type:” , type(complex_num)

Method:

Function	Description	Syntax
type()	returns the type of an object.	type(object)

Source Code :

```
int_num=10
print(f"Integer : {int_num} , Type : {type(int_num)}")
float=2.5
print(f"Float : {float} , Type : {type(float)}")
complex_num=2+3j
print(f"Complex Number : {complex_num} , Type : {type(complex_num)}")
```

Output :

Integer : 10 . Type : <class ‘int’>

Float : 2.5 , Type : <class ‘float’>

Complex Number : (2+3j) , Type : <class ‘complex’>

Result:

The program is successfully executed and the output was verified.

Experiment No : 3

Date : 07/10/24

Aim :

Write a program to calculate the area of a circle by reading inputs from the user.

Pseudocode :

1. READ radius
2. COMPUTE area as 3.14*radius*radius
3. PRINT “The area of circle with radius” , radius , “is” , area

Source Code :

```
radius=float(input("Enter the radius:"))
area=3.14*radius*radius
print(f"The area of circle with radius {radius} is {area}")
```

Output :

```
Enter the radius:5
The area of circle with radius 5.0 is 78.5
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 4

Date : 07/10/24

Aim :

Write a program to calculate the salary of an employee. Given his basic pay, HRA=10% of basic pay, TA=5% of the basic pay.

Pseudocode :

1. READ basic_pay
2. COMPUTE hra as 0.10*basic_pay
3. COMPUTE ta as 0.05*basic_pay
4. COMPUTE total as basic_pay+hra+ta
5. PRINT basic_pay , hra, ta, total

Source Code :

```
basic_pay=float(input("Enter the basic pay:"))
hra=0.10*basic_pay
ta=0.05*basic_pay
total=basic_pay+hra+ta
print(f"Basic Pay : {basic_pay} \nHRA : {hra} \nTA : {ta} \nTotal : {total}")
```

Output :

```
Enter the basic pay:12000
Basic Pay : 12000.0
HRA : 1200.0
TA : 600.0
Total : 13800.0
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 5

Date : 07/10/24

Aim :

Write a program to perform arithmetic operations on two integer numbers.

Pseudocode :

1. READ num1 , num2
2. COMPUTE sum as num1+num2 , difference as num1-num2, product as num1*num2
3. COMPUTE quotient as num1/num2, remainder as num1%num2, exponent as num1**num2
4. PRINT sum, difference, product, quotient, remainder and exponent

Source Code :

```
num1=int(input("Enter first number:"))
num2=int(input("Enter second number:"))
sum=num1+num2
difference=num1-num2
product=num1*num2
quotient=num1/num2
remainder=num1%num2
exponent=num1**num2
print(f"Sum : {num1} + {num2} = {sum}")
print(f"Difference : {num1} - {num2} = {difference}")
print(f"Product : {num1} * {num2} = {product}")
print(f"Quotient : {num1} / {num2} = {quotient}")
print(f"Remainder : {num1} % {num2} = {remainder}")
print(f"Exponent : {num1} ** {num2} = {exponent}")
```

Output :

```
Enter first number:20
Enter second number:2
```

Sum : 20 + 2 = 22

Difference : 20 – 2 =18

Product : 20 * 2 =40

Quotient : 20 / 2 = 10.0

Remainder : 20 % 2 =0

Exponent : 20 ** 2 =400

Result:

The program is successfully executed and the output was verified.

Experiment No : 6

Date : 07/10/24

Aim :

Write a program to get a string when in copies of given string.

Pseudocode :

1. READ string , n
2. COMPUTE c as string*n
3. PRINT c

Source Code :

```
string=input("Enter a string:")
n=int(input("Enter the number of copies:"))
c=string*n
print(f"The string after copying is:{c}")
```

Output :

```
Enter a string:python
Enter the number of copies:3
The string after copying is: pythonpython
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 7

Date : 07/10/24

Aim :

Write a program to accept an integer n and compute n+nn+nnn

[Hint : If n=6, the compute 5+55+555]

Pseudocode :

1. READ n
2. COMPUTE res as n + " " + n*2 + " " + n*3
3. PRINT "The pattern is" , res
4. COMPUTE sum as int(n) + int(n*2) + int(n*3)
5. PRINT sum

Source Code :

```
n=input("Enter a number :")
res=n+" "+(n*2)+" "+(n*3)
print("The pattern is",res)
sum=int(n)+int(n*2)+int(n*3)
print("The sum of the numbers in this pattern is",sum)
```

Output :

```
Enter a number :5
The pattern is 5 55 555
The sum of the numbers in the pattern is 615
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 8

Date : 07/10/24

Aim :

Write a program to find the biggest of three numbers.

Pseudocode :

1. READ num1 , num2 , num3
2. CHECK IF num1>num2 AND num1>num3 THEN
   PRINT num1 , “is biggest”
   ELIF num2>num1 AND num2>num3 THEN
   PRINT num2 , “is biggest”
   ELSE
   PRINT (num3 , “is biggest”)

Source Code :

```
num1=int(input("Enter first number:"))
num2=int(input("Enter second number:"))
num3=int(input("Enter third number:"))
if (num1>num2 and num1>num3):
    print(num1,"is biggest")
elif (num2>num1 and num2>num3):
    print(num2,"is biggest")
else:
    print(num3,"is biggest")
```

Output :

```
Enter first number:5
Enter second number:4
Enter third number:3
5 is biggest
```

Enter first number:5
Enter second number:3
Enter third number:4
5 is biggest

Enter first number:4
Enter second number:3
Enter third number:5
5 is biggest

Enter first number:3
Enter second number:4
Enter third number:5
5 is biggest

Enter first number:4
Enter second number:5
Enter third number:3
5 is biggest

Enter first number:3
Enter second number:5
Enter third number:4
5 is biggest

Result:

The program is successfully executed and the output was verified.

Experiment No : 9

Date : 07/10/24

Aim :

Write a program to check whether the year is leap year or not.

Pseudocode :

1. READ year
2. CHECK IF year%4 not equal to 0 THEN
   PRINT not leap year
   ELIF year%100 not equal to 0 and year%400 equal to 0 THEN
   PRINT leap year
   ELSE
   PRINT not leap year

Source Code :

```
year=int(input("Enter a year:"))
if year % 4 == 0 and (year % 100 != 0 or year % 400 == 0):
    print(f"{year} is a leap year.")
else:
    print(f"{year} is not a leap year.")
```

Output :

Enter a year:1900
1900 is not a leap year

Enter a year:2000
2000 is a leap year

Result:

The program is successfully executed and the output was verified.

Experiment No : 10

Date : 07/10/24

Aim :

Write a program to determine the rate of entry ticket in a trade fair based on age as follows:

AGE	RATE
<10	7
>=10 and <60	10
>=60	5

Pseudocode :

1. READ age
2. CHECK IF age less than 10 THEN
   PRINT “ticket fair:7”
   ELIF 10 less than or equal to age less than 60 THEN
   PRINT “ticket fair:10”
   ELIF age greater than or equal to 60 THEN
   PRINT “ticket fair:5”
   ELSE
   PRINT "Enter a valid age"

Source Code :

```
age=int(input("Enter your age:"))
if age<10:
    print("Your rate for entry ticket in this fair is 7 ")
elif 10<=age<60:
    print("Your rate for entry ticket in this fair is 10 ")
elif age>=60:
    print("Your rate for entry ticket in this fair is 5 ")
else:
    print("Enter a valid age")
```

Output :

Enter your age:34

Your rate for entry ticket in this fair is 10

Enter your age:2

Your rate for entry ticket in this fair is 7

Enter your age:66

Your rate for entry ticket in this fair is 5

Enter your age:2ab

Enter a valid age

Result:

The program is successfully executed and the output was verified.

Experiment No : 11

Date : 07/10/24

Aim :

Write a program to solve a quadratic equation.

Pseudocode :

1. READ a , b , c and COMPUTE d as b**2-4*a*c
2. CHECK IF d equal to 0 THEN COMPUTE root as b/(2*a)

   ELIF d greater than 0 THEN

   COMPUTE root1 as b+(math.sqrt(d))/(2*a) and root2 as b-(math.sqrt(d))/(2*a)

   ELIF d less than 0 THEN

   COMPUTE real_part as b/(2*a) and img_part as math.sqrt(-d)/(2*a)

Method :

Function	Description	Syntax
sqrt()	calculates the square root of a given number	math.sqrt(x)

Source Code :

```
import math
a=float(input("Enter coefficient of x^2:"))
b=float(input("Enter coefficient of x:"))
c=float(input("Enter the constant:"))
print(f"Quadratic Equation: {a}x^2+{b}x+{c}=0")
d=b**2-4*a*c
if d==0:
    root=b/(2*a)
    print(f"The roots are real and equal\nThe root is {root:.2f}")
elif d>0:
    root1=b+(math.sqrt(d))/(2*a)
    root2=b-(math.sqrt(d))/(2*a)
```

```
        print(f"The roots are real and different\nThe roots are {root1:.2f} and {root2:.2f}")
elif d<0:
        real_part=b/(2*a)
        img_part=math.sqrt(-d)/(2*a)
        print(f"The roots are complex\nThe roots are {real_part:.2f}+{img_part:.2f} and
        {real_part:.2f}{img_part:.2f}")
else:
        print("The equation has no real roots!\n")
```

Output :

```
Enter coefficient of x^2:1
Enter coefficient of x:2
Enter the constant:1
Quadratic Equation: 1.0x^2+2.0x+1.0=0
The roots are real and equal
The root is 1.00

Enter coefficient of x^2:1
Enter coefficient of x:4
Enter the constant:3
Quadratic Equation: 1.0x^2+4.0x+3=0
The roots are real and different
The roots are 5.00 and 3.00

Enter coefficient of x^2:1
Enter coefficient of x:2
Enter the constant:3
Quadratic Equation: 1.0x^2+2.0x+3=0
The roots are complex
The roots are 1.00+1.41i and 1.00-1.41i
```

Result:

The program is successfully executed and the output was verified.

LAB CYCLE -2

Experiment No : 1

Date :14/10/24

Aim :

Create a string from the given string where the first and last character are exchanged.

[Eg: python => nythonp]

Pseudocode :

1. READ str
2. COMPUTE new_str as str[-1]+str[1:-1]+str[0]
3. PRINT new_str

Source Code :

```
str=input("Enter a string:")
new_str=str[-1]+str[1:-1]+str[0]
print("New String : ",new_str)
```

Output :

```
Enter a string:python
New String : nythop
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 2

Date : 14/10/24

Aim :

Get a string from an input string where all occurences of the first character are replaced with '$' , expect the first character.

[Eg: onion => oni$n]

Pseudocode :

1. READ str
2. COMPUTE firstchar as str[0] and new_str as firstchar+str[1:].replace(firstchar,'$')
3. PRINT new_str

Method :

Function	Description	Syntax
replace()	create a new string by replacing occurrences of a specified substring with another substring.	string.replace(old, new, count)

Source Code :

```
str=input("Enter a string which has reoccurence of first character:")
firstchar=str[0]
newstr=firstchar+str[1:].replace(firstchar,'$')
print("New String : ",newstr)
```

Output :

```
Enter a string which has reoccurrence of first character:onion
New String : oni$n
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 3

Date : 14/10/24

Aim :

Create a single string separated with space from two strings by swapping the character at position 1. Eg : str1 = “Hello” str2 =”World” , then create a string str3 = “Hollo Werld” [Hint: use slicing and concatenation]

Pseudocode :

1. READ str1 , str2
2. COMPUTE newstr as str1[0]+str2[1]+str1[2:]+" " +str2[0]+str1[1]+str2[2:]
3. PRINT new_str

Source Code :

```
str1=input("Enter the first string:")
str2=input("Enter the second string:")
newstr=str1[0]+str2[1]+str1[2:]+" " +str2[0]+str1[1]+str2[2:]
print("New String : ",newstr)
```

Output :

```
Enter the first string:hello
Enter the second string:world
New String : hollo werld
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 4

Date : 14/10/24

Aim :

Count the number of characters in a string.

Pseudocode :

1. READ str
2. COMPUTE a as len(str)
3. PRINT a

Method :

Function	Description	Syntax
len()	get the length of an object, such as a string, list, tuple, or dictionary	len(object)

Source Code :

```
str=input("Enter a string:")
a=len(str)
print(f"The number of characters in the string {str} is {a}")
```

Output :

```
Enter a string:python
The number of charcacters in the string python is 6
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 5

Date : 14/10/24

Aim :

Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’

Pseudocode :

1. READ str
2. IF str endswith ing THEN COMPUTE newstr as str+"ly"
   ELSE COMPUTE newstr as str+"ing"
3. PRINT new_str

Method :

Function	Description	Syntax
endswith()	check if a string ends with a specified suffix	string.endswith(suffix)

Source Code :

```
str=input("Enter a string:")
if str.endswith(‘ing’):
        newstr=str+"ly"
else:
        newstr=str+"ing"
print("New String : ",newstr)
```

Output :

```
Enter a string:fast
New String : fasting

Enter a string:fasting
New String : fastingly
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 6
Date : 14/10/24

Aim :
Store a list of first names. Count the occurrences of ‘a’ within the list.

Pseudocode :
1. READ 3 names and STORE them in a list and SET count to 0
2. FOR each i in the list DO
    COUNT the occurrences of 'a' in the i and ADD to count
3. PRINT count

Method :

Function	Description	Syntax
append()	add a single item to the end of a list	list.append(item)
range()	generates a sequence of numbers, often used for iteration in loops	range(start, stop, step)

Source Code :
```
for i in range(3):
    str=input(“Enter the first name:”)
    l.append(str)
count=0
for i in l:
    for j in i:
        if j=="a":
            count+=1
print("The occurence of 'a' within the list of these first names is ",count)
```

Output :
Enter the first name of first person:shifana

Enter the first name of second person:jahana

Enter the first name of fifth person:reeja

The occurrence of ‘a’ within the list of these first names is 6

Result:
The program is successfully executed and the output was verified.

Experiment No : 7

Date : 14/10/24

Aim :

Write a python program to read two lists color-list1 and color-list2. Print out all colors from color-list1 not contained in color-list2.

Pseudocode :

1. READ a,b
2. FOR i = 1 to a
    INPUT colour and ADD colour to colour_list1
3. FOR i = 1 to b
    INPUT colour and ADD colour to colour_list2
4. FOR each i in colour_list1
    IF i not in colour_list2
        ADD i to result_list
5. PRINT result_list

Source Code :

```
a=int(input("Enter the number of colours to be inserted in first list of  colours:"))
colour_list1=[]
for i in range(a):
    colour_list1.append(input("Enter the  name of a colour:"))
print("The first list of colours is:",colour_list1)
b=int(input("Enter the number of colours to be inserted in second list of  colours:"))
colour_list2=[]
for i in range(b):
    colour_list2.append(input("Enter the  name of a colour:"))
print("The second list of colours is:",colour_list2)
result_list=[]
for i in colour_list1:
    if i not in colour_list2:
```

```
            result_list.append(i)
print("The colors from first list not contained in second list are:",result_list)
```

Output :

Enter the number of colours to be inserted in first list of colours:4

Enter the name of a colour:red

Enter the name of a colour:blue

Enter the name of a colour:green

Enter the name of a colour:black

The first list of colours is: [‘red’ , ‘blue’ , ‘green’ , ‘black’]

Enter the number of colours to be inserted in second list of colours:4

Enter the name of a colour:blue

Enter the name of a colour:black

Enter the name of a colour:yellow

Enter the name of a colour:grey

The second list of colours is: [‘blue’ , ‘black’ , ‘yellow’ , ‘grey’]

The colors from first list not contained in second list are: [‘red’ , ‘green’]

Result:

The program is successfully executed and the output was verified.

Experiment No : 8
Date : 14/10/24

Aim :
Create a list of colors from comma seperated color names entered by the user. Display first and last colors.

Pseudocode :
1. READ colors , SPLIT colors by commas and REMOVE spaces from each item

3. IF color_list is not empty

    PRINT "First Colour:", color_list[0] and PRINT "Last Colour:", color_list[-1]

  ELSE PRINT "No colors entered"

Method :

Function	Description	Syntax
split()	splits a string into a list of substrings	string.split(separator, maxsplit)
strip()	remove any leading and trailing whitespaces	string.strip([chars])

Source Code :
```
colors=input("Enter the name of colours seperated by commas:")
colors=colors.split(',')
color_list=[]
for i in colors:
    color_list.append(i.strip())
print("The list of colours is:",color_list)
if color_list:
    print("First Colour is",color_list[0],"and Last Colour is",color_list[-1])
else:
    print("No colors entered")
```

Output :
Enter the name of colours separated by commas:red , blue , yellow , green

The list of colours is: [‘red’ , ‘blue’ , ‘yellow’ , ‘green’]

First Colour: red

Last Colour: green

Result:
The program is successfully executed and the output was verified.

Experiment No : 9

Date : 14/10/24

Aim :

Write a program to prompt the user for a list of integers. For all values greater than 100 ,store ‘over’ instead.

Pseudocode :

1. READ numbers and SPLIT numbers into 'a'
2. FOR each i in a ADD int(i) to l
3. FOR each i in l
    IF i > 100 ADD "Over" to 'newlist'
    ELSE ADD i to 'newlist'
4. PRINT newlist

Source Code :

```
numbers=input("Enter a list of integers seperated by spaces:")
a=numbers.split()
l=[]
for i in a:
    l.append(int(i))
print("List:",l)
newlist=[]
for i in l:
    if i>100:
        newlist.append('Over')
    else:
        newlist.append(i)
print("New List:",newlist)
```

Output :

Enter a list of integers separated by spaces:23 45 67 234 567 23 333 56 345

List: [23, 45, 67, 234, 567, 23, 333, 56, 345]

New List: [23, 45, 67, ‘Over’, ‘Over’, 23, ‘Over’, 56, ‘Over’]

Result:

The program is successfully executed and the output was verified.

Experiment No : 10

Date : 14/10/24

Aim :

From a list of integers, create a list after removing even numbers.

Pseudocode :

1. READ numbers and SPLIT numbers into 'a'
2. FOR each i in a ADD int(i) to l
3. FOR each i in l
    IF i %2!=0 ADD i to 'newlist'
4. PRINT newlist

Source Code :

```
numbers=input("Enter a list of integers seperated by spaces:")
a=numbers.split()
l=[]
for i in a:
    l.append(int(i))
print("List:",l)
newlist=[]
for i in l:
    if i%2!=0:
        newlist.append(i)
print("New List:",newlist)
```

Output :

```
Enter a list of integers separated by spaces:1 2 3 4 5 6 7 8
List: [1, 2, 3, 4, 5, 6, 7, 8]
New List: [1, 3, 5, 7]
```

Result:

The program is successfully executed and the output was verified.

Experiment No : 11

Date : 14/10/24

Aim :

Accept a list of words and return the length of the longest word.

Pseudocode :

1. INPUT words and SPLIT words into list l and strip each word
2. PRINT l and SET max_len to -1
3. FOR each i in l , IF length of i > max_len
    SET max_len to length of i
4. PRINT max_len

Source Code :

```
words=input("Enter a list of words seperated by spaces:")
a=words.split()
l=[]
for i in a:
    l.append(i.strip())
print("The list of words is:",l)
max_len=-1
long=[]
for i in l:
    if len(i)>max_len:
        max_len=len(i)
print("Length of longest word: ",max_len)
```

Output :

Enter a list of words separated by spaces:red green lightblue yellow blue

The list of words is: [‘red’, ‘green’, ‘lightblue’, ‘yellow’, ‘blue’]

Length of longest word: 9

Result:

The program is successfully executed and the output was verified.

Experiment No : 12

Date : 14/10/24

Aim :

Write a program to prompt the user to enter two lists of integers and check

(a) Whether lists are of the same length.

(b) Whether the list sums to the same value.

(c) Whether any value occurs in both Lists.

Pseudocode :

1. INPUT number and number2, SPLIT and convert number1 to list 'l1' and number2 to list 'l2'
2. CHECK IF len(l1) == len(l2) THEN PRINT "Both lists have same length"
   ELSE "Both lists have of same length"
3. CHECK IF sum(l1) == sum(l2) THEN PRINT "Sums are equal"
   ELSE "Sums are not equal"
4. CREATE 'bothlist' as common elements between l1 and l2
5. CHECK IF 'bothlist' is not empty THEN PRINT bothlist
   ELSE PRINT "No common values"

Source Code :

```
number1=input("Enter integers seperated by spaces which is to be inserted to first list:").split()
l1=[]
for i in number1:
    l1.append(int(i))
number2=input("Enter integers seperated by spaces which is to be inserted to second list:").split()
l2=[]
for i in number2:
    l2.append(int(i))
print(f"First List: {l1}\nSecond List: {l2}\n(a)")
print("Length of first list: ",len(l1))
print("Length of second list: ",len(l2))
```

```
if len(l1)==len(l2):
        print("Both lists are of same length")
else:
        print("Both lists are not of same length")
sum1=0
sum2=0
for i in l1:
        sum1+=i
for i in l2:
        sum2+=i
print("(b)\nSum of elements of first list: ",sum1)
print("Sum of elements of second list: ",sum2)
if sum1==sum2:
        print("The sum of elements of both the lists are equal")
else:
        print("The sum of elements of both the lists are not equal")
bothlist=[]
for i in l1:
         if i in l2:
                bothlist.append(i)
print("(c)")
if bothlist:
        print("The value occurs in both the lists are:")
        for i in bothlist:
                print (i)
else:
        print("There is no value occurs in both the lists")
```

Output :

```
Enter integers separated by spaces which is to be inserted to first list:1 2 3
Enter integers separated by spaces which is to be inserted to second list:3 2 4
First List: [1, 2, 3]
Second List: [3, 2, 4]
```

(a)

Length of first list: 3

Length of second list: 3

Both lists are of same length

(b)

Sum of elements of first list: 6

Sum of elements of second list: 9

The sum of elements of both the lists are not equal

(c)

The value occurs in both the lists are:

2

3

Result:

The program is successfully executed and the output was verified.

Experiment No : 13

Date : 21/10/24

Aim :

Write a Python program to count the occurrences of each word in a line of text.

[Hint: use split() function and dictionary]

[Sample input : the quick brown fox jumps over the lazy dog

Output : {'the': 2, 'jumps': 1, 'brown': 1, 'lazy': 1, 'fox': 1, 'over': 1, 'quick': 1, 'dog.': 1}]

Pseudocode :

1. INPUT line and SPLIT line into list 'l'
2. FOR each i in 'l'
    CHECK IF i in 'd' THEN INCREMENT d[i]
    ELSE SET d[i] = 1
3. FOR each i in 'd'
    PRINT d[i]

Method :

Function	Description	Syntax
lower()	convert all uppercase characters in a string to lowercase	string.lower()
dict()	create a dictionary	dict()

Source Code :

```
line=input("Enter a line of text:").split()
l=[]
for i in line:
    i.lower()
    l.append(i)
print("The list of words in this line of text is:",l)
d=dict()
for i in l:
```

```
        if i in d:
                d[i]=d[i]+1
        else:
                d[i]=1
for i in d:
        print(f"The number of occurences of {i} : {d[i]}")
```

Output :

Enter a line of text:the quick brown fox jumps over the lazy dog

The list of words in this line of text is: [‘the’, ‘quick’, ‘brown’, ‘fox’, ‘jumps’, ‘over’, ‘the’, ‘lazy’, ‘dog’]

The number of occurrences of the:2

The number of occurrences of quick: 1

The number of occurrences of brown: 1

The number of occurrences of fox: 1

The number of occurrences of jumps: 1

The number of occurrences of over: 1

The number of occurrences of lazy: 1

The number of occurrences of dog: 1

Result:

The program is successfully executed and the output was verified.

Experiment No : 14

Date : 21/10/24

Aim :

List comprehensions:

(a) Generate positive list of numbers from a given list of integers

(b) Square of N numbers(c) Form a list of vowels selected from a given word

(d) Form a list ordinal value of each element of a word (Hint: use ord() to get ordinal values)

Pseudocode :

1. READ list of integers, FILTER positive numbers to 'l' and PRINT l
2. READ limit 'n', CREATE 'square' as squares of numbers from 1 to n and PRINT square
3. READ word, EXTRACT vowels into 'vowel' and PRINT vowel
4. READ word, CONVERT each character to ordinal value into 'ordinal' and PRINT ordinal

Method :

Function	Description	Syntax
ord()	returns the Unicode code point (integer representation) of a given character	ord(charcter)

Source Code :

```
print("(a)")
num=input("Enter a list of integers seperated by spaces:").split()
l=[]
for i in num:
        i=int(i)
        if i>0:
                l.append(i)
print("List of numbers containing only positive numbers:",l)
print("(b)")
square=[]
n=int(input("Enter a limit to find the square of numbers:"))
```

```
for i in range(1,n+1):
        square.append(i**2)
print(f"List containing the square of first {n} numbers : {square}")
print("(c)")
word=input("Enter a word:")
vowel=[]
for i in word:
        if i in ['a','e','i','o','u']:
                vowel.append(i)
print(f"List of vowels selected from the word {word} is {vowel}")
print("(d)")
w=input("Enter a word:")
ordinal=[]
for i in w:
        ordinal.append(ord(i))
print(f"List of ordinal value of each element of the word {w} is {ordinal}")
```

Output :

(a)

Enter a list of integers separated by spaces:1 -2 3 -4 5 -6

List of numbers containing only positive numbers: [1, 3, 5]

(b)

Enter a limit to find the square of numbers:5

List containing the square o first 5 numbers: [1, 4, 9, 16, 25]

(c)

Enter a word:education

List of vowels selected from the word education is [‘e’, ‘u’, ‘a’, ‘i’, ‘o’]

(d)

Enter a word:shifana

List of ordinal value of each element of the word shifana is [115, 104, 105, 102, 97, 110, 97]

Result:

The program is successfully executed and the output was verified.

Experiment No : 15

Date : 21/10/24

Aim :

Sort dictionary in ascending and descending order.

Pseudocode :

1. CREATE dictionary 'd' with key-value pairs
2. SORT 'd' in ascending order and store in 'asc'
3. SORT 'd' in descending order and store in 'desc'
4. PRINT d, asc and desc

Method :

Function	Description	Syntax
sorted()	return a new sorted list from the items of any iterable	sorted(iterable,key=None, reverse=False)
items()	used with dictionaries to return a view object that displays a list of key-value pairs as tuples	dictionary.items()

Source Code :

```
d={'orange':2,'banana':3,'apple':5}
asc=dict(sorted(d.items()))
desc=dict(sorted(d.items(),reverse=True))
print("Dictionary :",d)
print("Ascending Order :",asc)
print("Descending Order :",desc)
```

Output :

Dictionary : {‘orange’ : 2, ‘banana’ : 3, ‘apple’ : 5}

Ascending Order : {‘apple’ : 5, ‘banana’ : 3, ‘orange’ : 2}

Descending Order : {‘orange’ : 2, ‘banana’ : 3, ‘apple’ : 5}

Result:

The program is successfully executed and the output was verified.

Experiment No : 16

Date : 21/10/24

Aim :

Merge two dictionaries.

Pseudocode :

1. CREATE dictionary 'd1' with key-value pairs
2. CREATE dictionary 'd2' with key-value pairs
3. PRINT "First Dictionary:", d1
4. PRINT "Second Dictionary:", d2
5. MERGE 'd2' into 'd1' using update()
6. PRINT "Merged Dictionary:", d1

Method :

Function	Description	Syntax
update()	update a dictionary with the key-value pairs from another dictionary or an iterable of key-value pairs.	dictionary1.update(dictionary2)

Source Code :

```
d1={'a':1,'b':2}
d2={'b':3,'c':4}
print("First Dictionary : ",d1)
print("Second Dictionary : ",d2)
d1.update(d2)
print("Merged Dictionary : ",d1)
```

Output :

First Dictionary : {‘a’ : 1, ‘b’ : 2}

Second Dictionary : {‘b’ : 3, ‘c’ : 4}

Merged Dictionary : {‘a’ : 1, ‘b’ : 3, ‘c’ : 4}

Result:

The program is successfully executed and the output was verified.