Lab Cycle 2(20MCA131)

PROGRAM 1

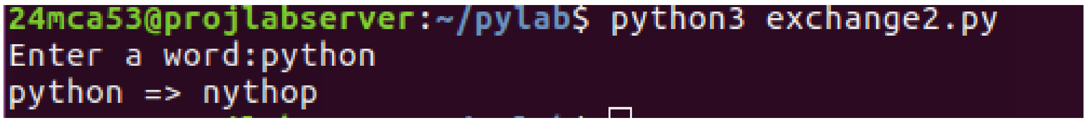
AIM:

Create a string from the given string where the first and last character are exchanged.
Eg: Python ⇒ nythoP

SOURCE CODE:

```
str=input("Enter a word:")
if len(str)>1:
 r=str[-1]+str[1:-1]+str[0]
 print(str,"=>",r)
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 exchange2.py
Enter a word:python
python => nythop
```

****

PROGRAM 2

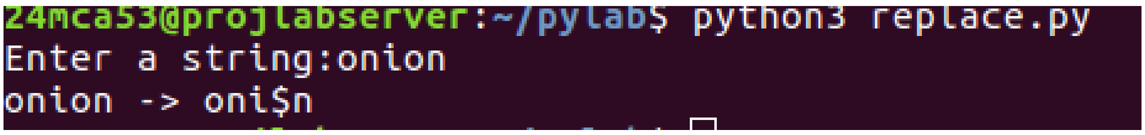
AIM:

Get a string from an input string where all occurrences of the first character are replaced with ‘$’, except the first character. [eg: onion -> oni$n]

SOURCE CODE:

```
s=input("Enter a string:")
result=s[0]+s[1:].replace(s[0],'$')
print(s,"->",result)
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 replace.py
Enter a string:onion
onion -> oni$n
```

****

PROGRAM 3

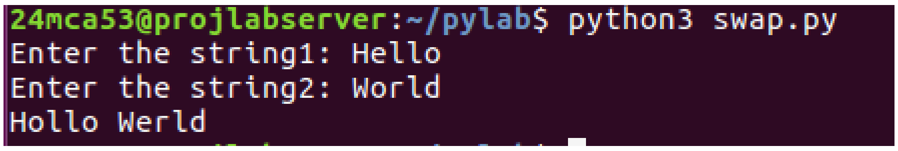
AIM:

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

SOURCE CODE:

```
str1=input("Enter the string1: ")
str2=input("Enter the string2: ")
swap_str1=str1[0]+str2[1]+str1[2:]
swap_str2=str2[0]+str1[1]+str2[2:]
result=swap_str1+" "+swap_str2
print(result)
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 swap.py
Enter the string1: Hello
Enter the string2: World
Hollo Werld
```

****

PROGRAM 4

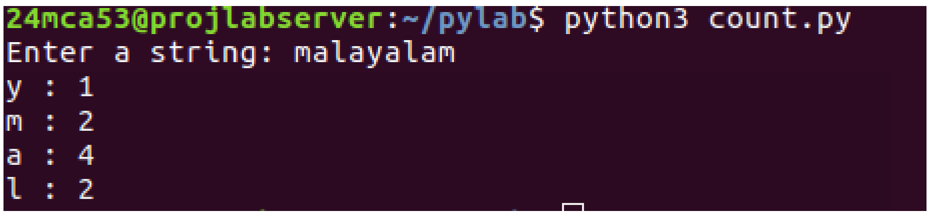
AIM:

Count the number of characters (character frequency) in a string.

SOURCE CODE:

```
s=input("Enter a string: ")
for char in set(s):
 print(char,":",s.count(char))
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 count.py
Enter a string: malayalam
y : 1
m : 2
a : 4
l : 2
```

****

PROGRAM 5

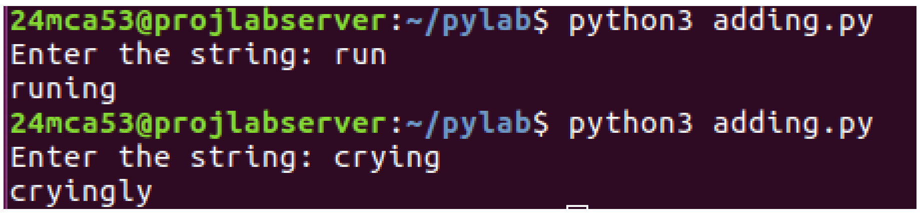
AIM:

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

SOURCE CODE:

```
s=input("Enter the string: ")
if s.endswith('ing'):
 r=s+'ly'
else:
 r=s+'ing'
print(r)
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 adding.py
Enter the string: run
runing
24mca53@projlabserver:~/pylab$ python3 adding.py
Enter the string: crying
cryingly
```

****

PROGRAM 6

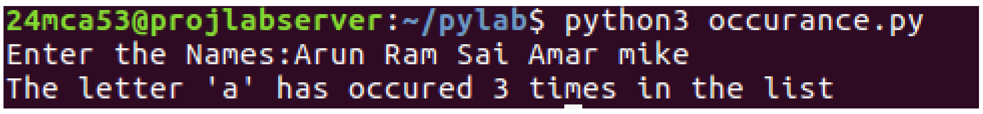
AIM:

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

SOURCE CODE:

```
names=input("Enter the Names:").split()
a_count=0
for name in names:
 a_count=a_count+name.count('a')
print(f"The letter 'a' has occured {a_count} times in the list")
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 occurance.py
Enter the Names:Arun Ram Sai Amar mike
The letter 'a' has occured 3 times in the list
```

****

PROGRAM 7

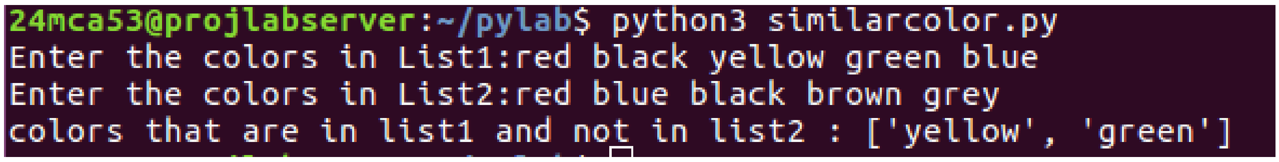
AIM:

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.

SOURCE CODE:

```
list1=input("Enter the colors in List1:").split()
list2=input("Enter the colors in List2:").split()
new=[color for color in list1 if color not in list2]
print("colors that are in list1 and not in list2 :",new)
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 similarcolor.py
Enter the colors in List1:red black yellow green blue
Enter the colors in List2:red blue black brown grey
colors that are in list1 and not in list2 : ['yellow', 'green']
```

****

PROGRAM 8

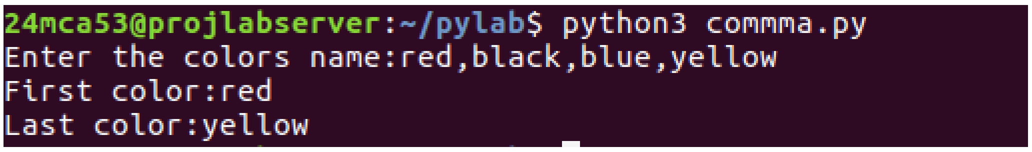
AIM:

Create a list of colors from comma-separated color names entered by the user. Display first and last colors.

SOURCE CODE:

```
colors=input("Enter the colors name:").split(',')
First_color=colors[0]
Last_color=colors[-1]
print(f"First color:{First_color}\nLast color:{Last_color}")
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 commma.py
Enter the colors name:red,black,blue,yellow
First color:red
Last color:yellow
```

****

PROGRAM 9

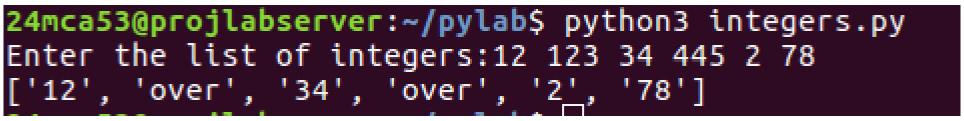
AIM:

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

SOURCE CODE:

```
integers=input("Enter the list of integers:").split()
for i in range(len(integers)):
 if int(integers[i])>100:
  integers[i]='over'
print(integers)s
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 integers.py
Enter the list of integers:12 123 34 445 2 78
['12', 'over', '34', 'over', '2', '78']
```

****

PROGRAM 10

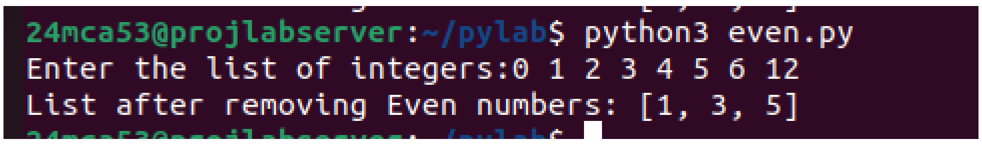
AIM:

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

SOURCE CODE:

```
integers=input("Enter the list of integers:").split()
odd=[]
for i in range(len(integers)):
 if int(integers[i])%2!=0:
  odd.append(i)
print("List after removing Even numbers:",odd)
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 even.py
Enter the list of integers:0 1 2 3 4 5 6 12
List after removing Even numbers: [1, 3, 5]
```

****

PROGRAM 11

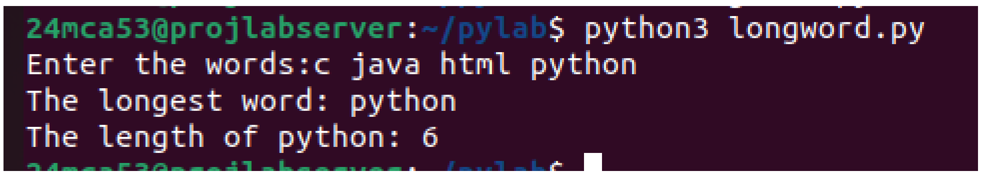
AIM:

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

SOURCE CODE:

```
words=input(“Enter the words:”).split()
longest=-1
for i in words:
 if len(i)>longest
  longest=len(i)
  word=i
print(“The longest word:”,word)
print(f”The length of{word}:{longest}”)
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 longword.py
Enter the words:c java html python
The longest word: python
The length of python: 6
```

****

PROGRAM 12

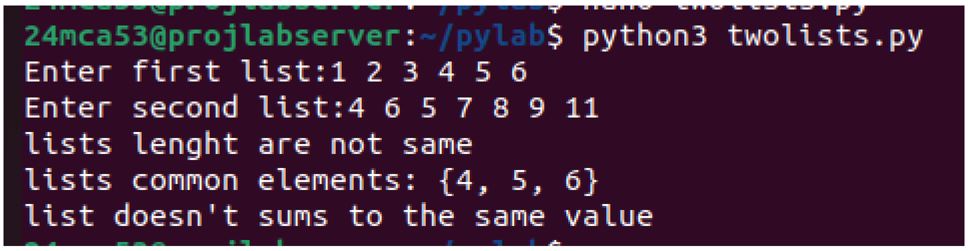
AIM:

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.

SOURCE CODE:

```
lst1=[int(num) for num in input("Enter first list:").split()]
lst2=[int(num) for num in input("Enter second list:").split()]

lenght=len(lst1)==len(lst2)
lsum=sum(lst1)==sum(lst2)
common=set(lst1)&set(lst2)

if lenght:
        print("lists lenghts are same")
else:
        print("lists lenght are not same")

print(f"lists common elements: {common}")

if lsum:
        print("list sums to the same value")
else:
        print("list doesn't sums to the same value")
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 twolists.py
Enter first list:1 2 3 4 5 6
Enter second list:4 6 5 7 8 9 11
lists lenght are not same
lists common elements: {4, 5, 6}
list doesn't sums to the same value
```

PROGRAM 13

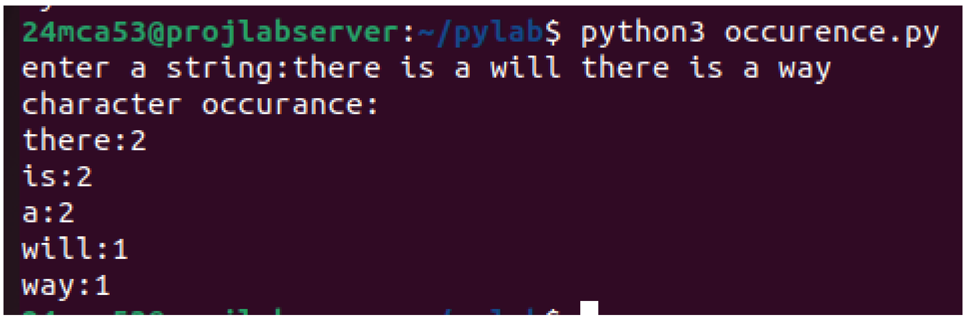
AIM:

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}

SOURCE CODE:

```
sentence=[word for word in input("enter a string:").lower().split()]
freq_dict={}
for word in sentence:
    if word in freq_dict:
        freq_dict[word]+=1
    else:
        freq_dict[word]=1
print("character occurance:")
for key,value  in freq_dict.items():
    print(f"{key}:{value}")
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 occurence.py
enter a string:there is a will there is a way
character occurance:
there:2
is:2
a:2
will:1
way:1
```

****

PROGRAM 14

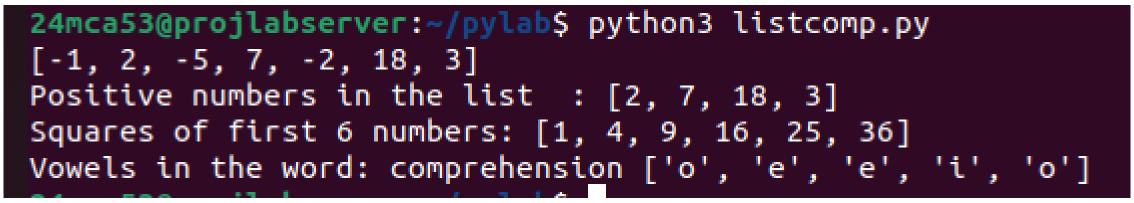
AIM:

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)

SOURCE CODE:

```
numbers = [-1, 2, -5, 7, -2, 18, 3]
print(numbers)
positive_numbers = [num for num in numbers if num > 0]
print(f"Positive numbers in the list  :", positive_numbers)
N=6
squares = [num ** 2 for num in range(1, N + 1)]
print("Squares of first 6 numbers:", squares)
word = "comprehension"
vowels = [char for char in word if char in 'aeiou']
print(f"Vowels in the word: {word}", vowels)
word = "hello"
ordinal_values = [ord(char) for char in word]
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 listcomp.py
[-1, 2, -5, 7, -2, 18, 3]
Positive numbers in the list  : [2, 7, 18, 3]
Squares of first 6 numbers: [1, 4, 9, 16, 25, 36]
Vowels in the word: comprehension ['o', 'e', 'e', 'i', 'o']
```

****

<u>**PROGRAM 15**</u>

AIM:

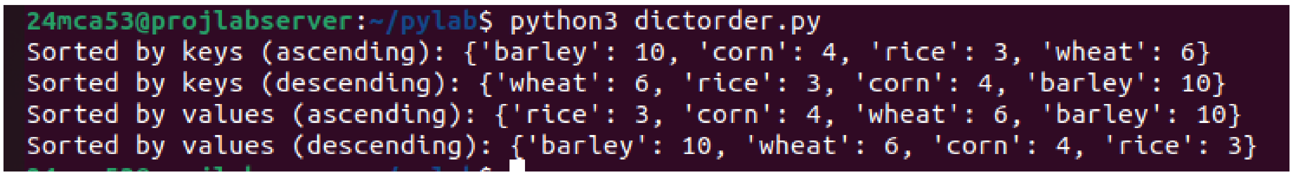
Sort dictionary in ascending and descending order.

SOURCE CODE:

```
my_dict = {'rice': 3, 'wheat': 6, 'barley': 10, 'corn': 4}
keys_asc = dict(sorted(my_dict.items()))
print("Sorted by keys (ascending):", keys_asc)
keys_desc = dict(sorted(my_dict.items(), reverse=True))
print("Sorted by keys (descending):", keys_desc)
values_asc = dict(sorted(my_dict.items(), key=lambda item: item[1]))
```

```
print("Sorted by values (ascending):", values_asc)
values_desc = dict(sorted(my_dict.items(), key=lambda item: item[1], reverse=True))
print("Sorted by values (descending):", values_desc)
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 dictorder.py
Sorted by keys (ascending): {'barley': 10, 'corn': 4, 'rice': 3, 'wheat': 6}
Sorted by keys (descending): {'wheat': 6, 'rice': 3, 'corn': 4, 'barley': 10}
Sorted by values (ascending): {'rice': 3, 'corn': 4, 'wheat': 6, 'barley': 10}
Sorted by values (descending): {'barley': 10, 'wheat': 6, 'corn': 4, 'rice': 3}
```

****

PROGRAM 16

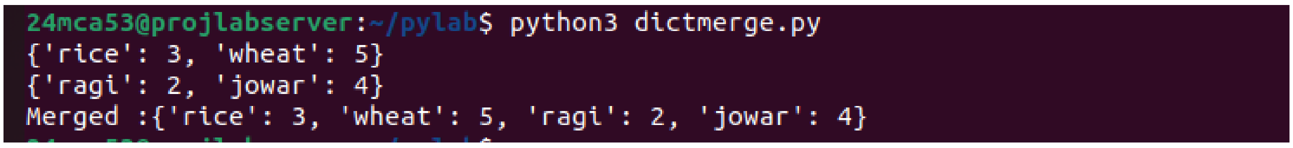
AIM:

Merge two dictionaries.

SOURCE CODE:

```
dict1 = {'rice': 3, 'wheat': 5}
dict2 = {'ragi': 2, 'jowar': 4}
print(dict1)
print(dict2)
dict1.update(dict2)
print(f"Merged :{dict1}")
```

OUTPUT:



```
24mca53@projlabserver:~/pylab$ python3 dictmerge.py
{'rice': 3, 'wheat': 5}
{'ragi': 2, 'jowar': 4}
Merged :{'rice': 3, 'wheat': 5, 'ragi': 2, 'jowar': 4}
```

****
****