

PROGRAM-1

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

SOURCE CODE:

```
str=input("Enter a string:")  
print(str[-1]+str[1:5]+str[0])
```

OUTPUT:

```
Enter a string: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.

SOURCE CODE:

```
str1=input("Enter the string:")
char=str1[0]
str1=str1.replace(char,'$')
print(char+str1[1:])
```

OUTPUT:

```
Enter the string:onion
oni$n
```

PROGRAM-3

AIM: Create a single string separated with space from two strings by swapping the character at position 1.

SOURCE CODE:

```
str1=input("Enter first string:")
str2=input("Enter second string:")
n=len(str1)
n2=len(str2)
str1sub=str1[1]
str2sub=str2[1]
print(str1[0]+str2sub+str1[2:n],",",str2[0]+str1sub+str2[2:n2])
```

OUTPUT:

```
Enter first string:hello
Enter second string:world
hollo  world
```

PROGRAM-4

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

SOURCE CODE:

```
n=input("Enter the string:")
s={}
for i in n:
    if i in s:
        s[i]+=1
    else:
        s[i]=1
print(s)
```

OUTPUT:

```
Enter the string:hello
{'h': 1, 'e': 1, 'l': 2, 'o': 1}
```

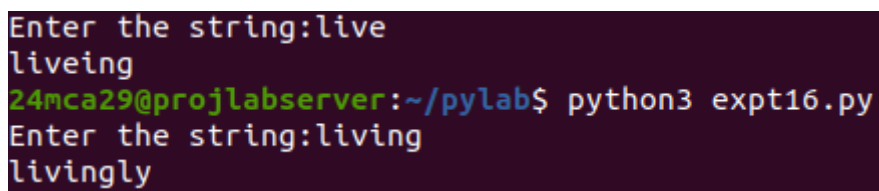
PROGRAM-5

AIM: Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'.

SOURCE CODE:

```
str=input("Enter the string:")
if str[-3:]=="ing":
    print(str+"ly")
else:
    print(str+"ing")
```

OUTPUT:

A terminal window with a dark purple background. The first prompt 'Enter the string:' is followed by the input 'live', resulting in the output 'liveing'. The second prompt 'Enter the string:' is followed by the input 'living', resulting in the output 'livingly'. The terminal prompt is '24mca29@projlabsrver:~/pylab\$ python3 expt16.py'.

```
Enter the string:live
liveing
24mca29@projlabsrver:~/pylab$ python3 expt16.py
Enter the string:living
livingly
```

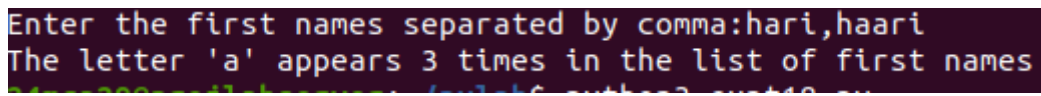
PROGRAM-6

AIM: Store a list of first names. Count the occurrences of 'a' within the list.

SOURCE CODE:

```
name=input("Enter the first names separated by comma:")  
count_a=name.lower().count('a')  
print(f"The letter 'a' appears {count_a} times in the list of first names")
```

OUTPUT:

A screenshot of a terminal window with a dark background and light-colored text. The first line shows the input prompt and the user's input: "Enter the first names separated by comma:hari,haari". The second line shows the output of the program: "The letter 'a' appears 3 times in the list of first names".

```
Enter the first names separated by comma:hari,haari  
The letter 'a' appears 3 times in the list of first names
```

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:

```
l1=input("Enter colors separated by comma:").split(',')
l2=input("Enter colors separated by comma:").split(',')
for color in l1:
    if color not in l2:
        print(color)
```

OUTPUT:

```
Enter colors seperated by comma:red,orange,blue
Enter colors seperated by comma:red
orange
blue
```

PROGRAM-8

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

SOURCE CODE:

```
color=input("Enter colors separated by comma:").split(',')  
print(color[0],",",color[-1])
```

OUTPUT:

```
Enter colors separated by comma:red,blue,orange  
red orange
```

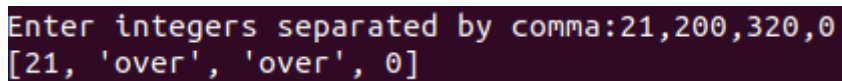

PROGRAM-9

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

SOURCE CODE:

```
inp=input("Enter integers separated by comma:").split(',')
res=[]
for num in inp:
    if int(num)>100:
        res.append('over')
    else:
        res.append(int(num))
print(res)
```

OUTPUT:

A screenshot of a terminal window showing the program's execution. The prompt 'Enter integers separated by comma:' is followed by the input '21,200,320,0'. The output is '[21, 'over', 'over', 0]'.

```
Enter integers separated by comma:21,200,320,0
[21, 'over', 'over', 0]
```

PROGRAM-10

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

SOURCE CODE:

```
inp=input("Enter integers separated by comma:").split(',')
res=[]
for num in inp:
    if int(num)%2!=0:
        res.append(num)
    elif int(num)==0:
        res.append(num)
print("New list is:",res)
```

OUTPUT:

```
Enter integers separated by comma:21,12,14,0
New list is: ['21', '0']
```

PROGRAM-11

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

SOURCE CODE:

```
wd=input("Enter words separated by space:").split(' ')
max=0
for w in wd:
    if len(w)>max:
        max=len(w)
print("Length of longest word is ",max)
```

OUTPUT:

```
Enter words separated by space:hello ooooooo
Length of longest word is 7
```

PROGRAM-12

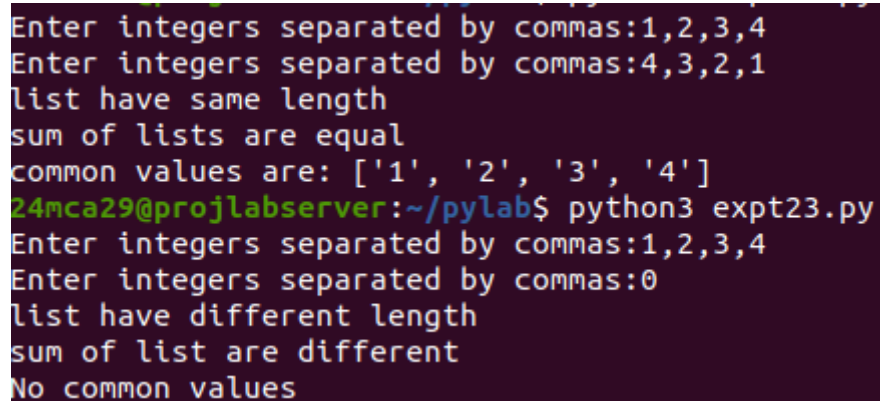
AIM: Write a program to prompt the user to enter two lists of integers and check

- (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:

```
inp=input("Enter the line of text:".lower())
words=inp.split()
word_s={}
for inp in words:
    if inp in word_s:
        word_s[inp]+=1
    else:
        word_s[inp]=1
print(word_s)
```

OUTPUT:



```
Enter integers separated by commas:1,2,3,4
Enter integers separated by commas:4,3,2,1
list have same length
sum of lists are equal
common values are: ['1', '2', '3', '4']
24mca29@projlabsrver:~/pylab$ python3 expt23.py
Enter integers separated by commas:1,2,3,4
Enter integers separated by commas:0
list have different length
sum of list are different
No common values
```

PROGRAM-13

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

SOURCE CODE:

```
inp=input("enter the line of text:".lower())
words=inp.split()
word_s={}
for inp in words:
    if inp in word_s:
        word_s[inp]+=1
    else:
        word_s[inp]=1
print(word_s )
```

OUTPUT:

```
enter the line of text:hello hi hi
{'hello': 1, 'hi': 2}
```

PROGRAM-14

AIM: . List comprehensions:

- (a) Generate a 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:

```
ls1=[-5,8,10,8,-15,18]
pls1=[num for num in ls1 if num>0]
print("+ve numbers",pls1)
print([i*i for i in range(1,6)])
wd="hello"
vow={word for word in wd if word in['a','e','i','o','u']}
print(vow)
ordval=[ord(ch) for ch in wd]
print("Ordinal values:",ordval)
```

OUTPUT:

```
+ve numbers [8, 10, 8, 18]
[1, 4, 9, 16, 25]
{'e', 'o'}
Ordinal values: [104, 101, 108, 108, 111]
```

PROGRAM-15

AIM: Sort dictionary in ascending and descending order.

SOURCE CODE:

```
my_dict={'banana':3,'apple':5,'orange':2,'kiwi':4}
askey=sorted(my_dict.keys())
dskey=sorted(my_dict.keys(),reverse=True)
print("Ascending sorting of keys:",askey)
print("Descending sorting of keys:",dskey)
asv=sorted(my_dict.values())
dsv=sorted(my_dict.values(),reverse=True)
print("Ascending sorting of values:",asv)
print("Descending sorting of values:",dsv)
```

OUTPUT:

```
Ascending sorting of keys: ['apple', 'banana', 'kiwi', 'orange']
Descending sorting of keys: ['orange', 'kiwi', 'banana', 'apple']
Ascending sorting of values: [2, 3, 4, 5]
Descending sorting of values: [5, 4, 3, 2]
```

PROGRAM-16

AIM: . Merge two dictionaries.

SOURCE CODE:

```
dict1={'banana':3,'apple':5}  
dict2={'orange':2,'kiwi':4}  
print(dict1)  
print(dict2)  
dict1.update(dict2)  
print("Merged:",dict1)
```

OUTPUT:

```
{'banana': 3, 'apple': 5}  
{'orange': 2, 'kiwi': 4}  
Merged: {'banana': 3, 'apple': 5, 'orange': 2, 'kiwi': 4}
```