CYCLE 2

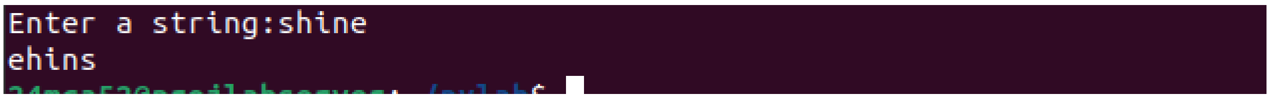
PROGRAM 1

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

SOURCE CODE:

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

OUTPUT:



```
Enter a string:shine
ehins
```

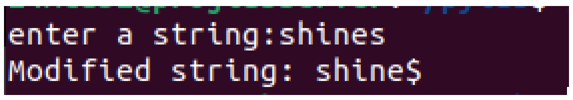
PROGRAM 2

AIM: Get a string from an input string where all occurrences of the first character are replaced with '$', except the first character.

SOURCE CODE:

```
string=input("enter a string:")
first_char=string[0]
new_string=first_char+string[1:].replace(first_char,"$")
print("Modified string:",new_string)
```

OUTPUT:



```
enter a string:shines
Modified string: shine$
```

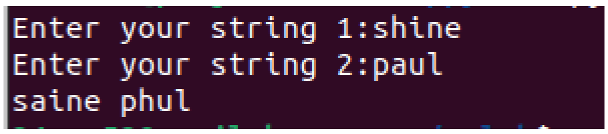
PROGRAM 3

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

SOURCE CODE:

```
string1=input("Enter your string 1:")
string2=input("Enter your string 2:")
swap_str1=string1[0]+string2[1]+string1[2:]
swap_str2=string2[0]+string1[1]+string2[2:]
string3=swap_str1+" "+swap_str2
print(string3)
```

OUTPUT:



```
Enter your string 1:shine
Enter your string 2:paul
saine phul
```

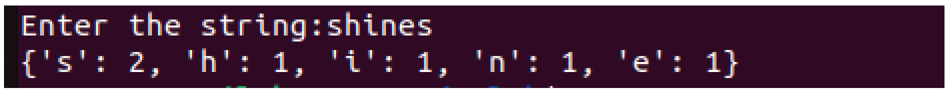
PROGRAM 4

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

SOURCE CODE:

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

OUTPUT:



```
Enter the string:shines
{'s': 2, 'h': 1, 'i': 1, 'n': 1, 'e': 1}
```

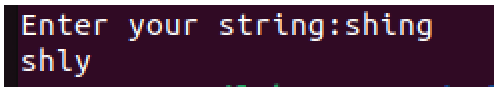
PROGRAM 5

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

SOURCE CODE:

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

OUTPUT:



```
Enter your string:shing
shly
```

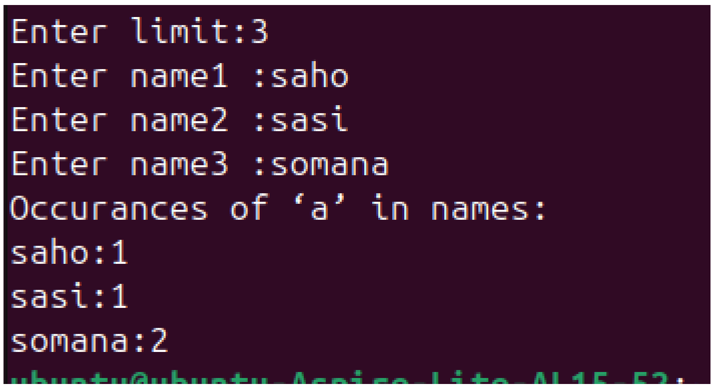
PROGRAM 6

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

SOURCE CODE:

```
names=[]
n=int(input("Enter limit:"))
for i in range(n):
	el=input(f"Enter name{i+1} :")
	names.append(el)
print("Occurances of ‘a’ in names:")
for name in names:
	print(f"{name}:{name.lower().count('a')}")
```

OUTPUT:



```
Enter limit:3
Enter name1 :saho
Enter name2 :sasi
Enter name3 :somana
Occurances of ‘a’ in names:
saho:1
sasi:1
somana:2
```

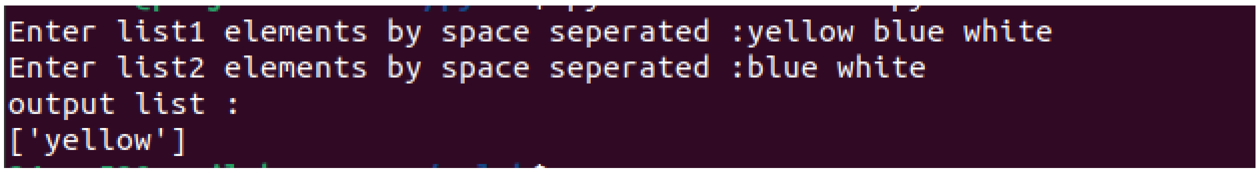
PROGRAM 7

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:

```
lst1=input("Enter list1 elements by space seperated :").split()
lst2=input("Enter list2 elements by space seperated :").split()
print("output list :")
print(list(set(lst1)-set(lst2)))
```

OUTPUT:



```
Enter list1 elements by space seperated :yellow blue white
Enter list2 elements by space seperated :blue white
output list :
['yellow']
```

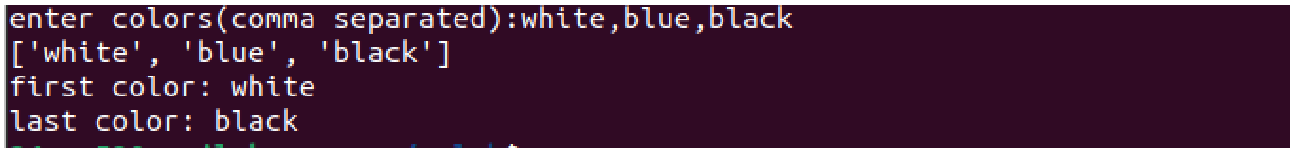
PROGRAM 8

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 colors(comma separated):").split(",")
colors=[color for color in colors]
print(colors)
print("first color:",colors[0])
print("last color:",colors[-1])
```

OUTPUT:



```
enter colors(comma separated):white,blue,black
['white', 'blue', 'black']
first color: white
last color: black
```

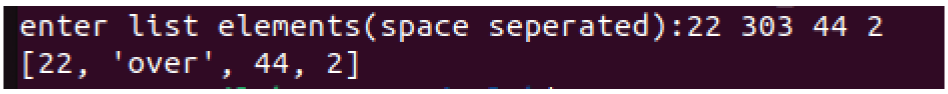
PROGRAM 9

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

SOURCE CODE:

```
lst=[int(num) for num in input("enter list elements(space seperated):").split()]

for i in range(len(lst)):
        if lst[i] > 100:
                lst[i]="over"
print(lst)
```

OUTPUT:



```
enter list elements(space seperated):22 303 44 2
[22, 'over', 44, 2]
```

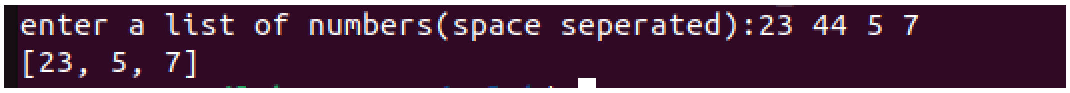
PROGRAM 10

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

SOURCE CODE:

```
lst=[int(num) for num in input("enter a list of numbers(space seperated):").split()]
odd_lst=[odd for odd in lst if odd%2!=0]

print(“New list:”,odd_lst)
```

OUTPUT:



```
enter a list of numbers(space seperated):23 44 5 7
[23, 5, 7]
```

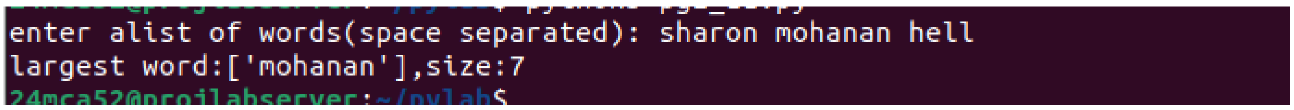
PROGRAM 11

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

SOURCE CODE:

```
lst=input("enter a list of words(space separated): ").split()
maxlenght=max(len(word) for word in lst)
lg_word=[word for word in lst if len(word)==maxlenght]

print(f"largest word:{lg_word},size:{maxlenght}")
```

OUTPUT:



```
enter alist of words(space separated): sharon mohanan hell
largest word:['mohanan'],size:7
24mca52@projlabserver:~/pylab$
```

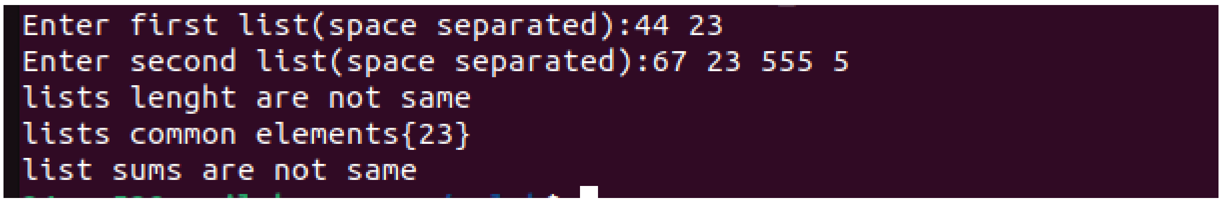
PROGRAM 12

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.

SORCE CODE:

```
lst1=[int(num) for num in input("Enter first list(space separated):").split()]
lst2=[int(num) for num in input("Enter second list(space separated):").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 are same")
else:
        print("list sums are not same")
```

OUTPUT:



```
Enter first list(space separated):44 23
Enter second list(space separated):67 23 555 5
lists lenght are not same
lists common elements{23}
list sums are not same
```

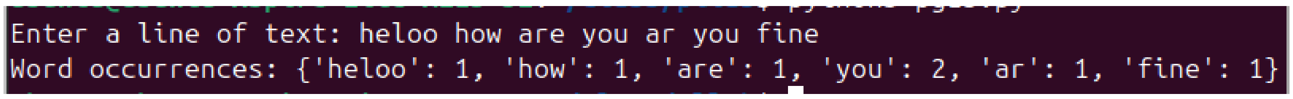
PROGRAM 13

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

SOURCE CODE:

```
text = input("Enter a line of text: ")
words = text.split()
word_count = {}
for word in words:
        word = word.lower()
        if word in word_count:
                word_count[word] += 1
        else:
                word_count[word] = 1
print("Word occurrences:", word_count)
```

OUTPUT:



```
Enter a line of text: heloo how are you ar you fine
Word occurrences: {'heloo': 1, 'how': 1, 'are': 1, 'you': 2, 'ar': 1, 'fine': 1}
```

PROGRAM 14

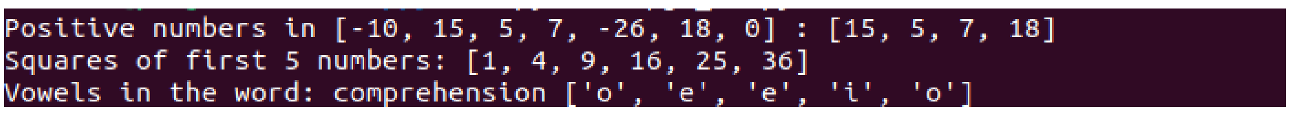
AIM: List comprehensions:

(a) Generate positive list of numbers from a given list of integers
(b) Square of N numbers

SOURCE CODE:

```
numbers = [-10, 15, 5, 7, -26, 18, 0]
positive_numbers = [num for num in numbers if num > 0]
print(f"Positive numbers in {numbers} :", 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]
print("Ordinal values of each character in the word : hello", ordinal_values)
```

OUTPUT:



```
Positive numbers in [-10, 15, 5, 7, -26, 18, 0] : [15, 5, 7, 18]
Squares of first 5 numbers: [1, 4, 9, 16, 25, 36]
Vowels in the word: comprehension ['o', 'e', 'e', 'i', 'o']
```

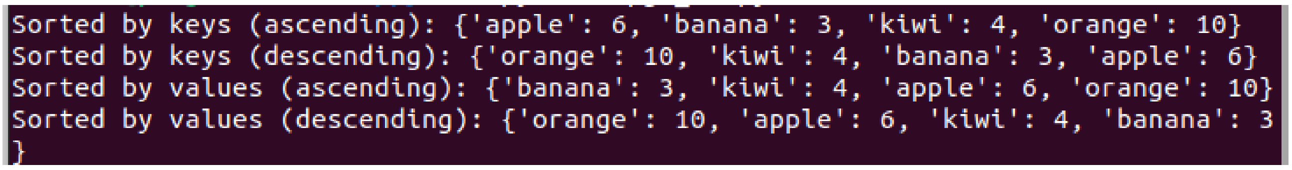
PROGRAM 15

AIM: Sort dictionary in ascending and descending order.

SOURCE CODE:

```
my_dict = {'banana': 3, 'apple': 5, 'orange': 2, 'kiwi': 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:


```
Sorted by keys (ascending): {'apple': 6, 'banana': 3, 'kiwi': 4, 'orange': 10}
Sorted by keys (descending): {'orange': 10, 'kiwi': 4, 'banana': 3, 'apple': 6}
Sorted by values (ascending): {'banana': 3, 'kiwi': 4, 'apple': 6, 'orange': 10}
Sorted by values (descending): {'orange': 10, 'apple': 6, 'kiwi': 4, 'banana': 3
}
```

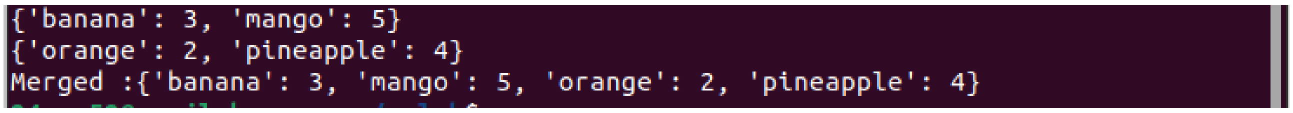
PROGRAM 16

AIM: Merge two dictionaries.

SOURCE CODE:

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

OUTPUT:


```
{'banana': 3, 'mango': 5}
{'orange': 2, 'pineapple': 4}
Merged :{'banana': 3, 'mango': 5, 'orange': 2, 'pineapple': 4}
```