**Experiment No : 1**

**Date : 25/9/2023**

**Aim:**

Write a program that prompts the user to enter his first name and last name and then displays a message "Greetings!!! First name Last name".

**Pseudocode:**

Read first name.
Read last name.
Print Greetings!!! First name Last name

**Method :**

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

**Source Code:**

```
a = input("Enter first name: ")
b = input("Enter last name: ")
print("Greetings!!!", a, b)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/Desktop/qqqq/cycle1$ python3 program1.py
Enter the first name:Jithin
Enter the last name:S John
Greetings!!! Jithin S John
```

**Result:** The program is successfully executed and the output is verified

**Experiment No : 2**

**Date : 25/9/2023**

**Aim:**

Write a program to demonstrate different number data types in python.

**Pseudocode :**

Initialize n1 as any integer value.
Initialize n2 as any float value.
Initialize n3 as any complex number.
Print n1 and its type.
Print n2 and its type.
Print n3 and its type.

**Method :**

| Function | Description | Syntax |
|----------|-------------|--------|
| type() | Returns the type of the specified object | type(object) |

**Source Code:**

```
a = 10
b = 5.5
c = 5+10j
print("Num1 =",a, type(a))
print("Num2 =",b, type(b))
print("Num3 =",c, type(c))
```

**Output:**



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

**Experiment No : 3**

**Date : 25/9/2023**

**Aim:**

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

**Pseudocode :**

Read the radius r of the circle.
Area=π*r*r
Print Area

**Source Code :**

```
r = input("Enter the Radius: ")
area = 3.14*float(r)*float(r)
print("Area of Circle =",area)
```

**Output:**



```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle1$ python3 program3.py
enter radius of circle:10
Area of Circle : 314.0
```

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

**Experiment No : 4**

**Date : 4/10/2023**

**Aim:**

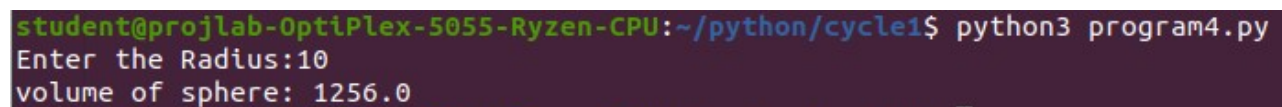Write a program to calculate the volum of a sphere by reading inputs from the user.

**Pseudocode :**

Read the radius r of the sphere.
Volume =4*π*r*r
Print Volume

**Source Code :**

```
r = input("Enter the Radius: ")
Volume = 4*3.14*float(r)*float(r)
print("Volume of sphere =",volume)
```

**Output:**



```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle1$ python3 program4.py
Enter the Radius:10
volume of sphere: 1256.0
```

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

4

**Experiment No : 5**

**Date :  4/10/2023**

**Aim:**

Write a program to calculate the salary of an employee given his basic pay (to be entered by the user). HRA = 10 percent of the basic pay, TA = 5 percent of the basic pay.
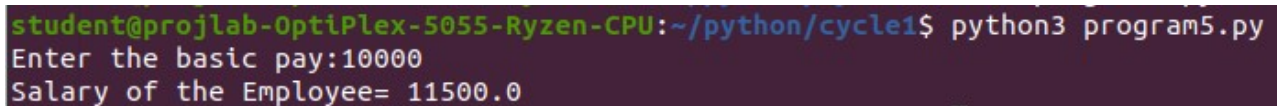
**Pseudocode :**

Input basic pay bp.
HRA=10% of bp
TA=5% of bp
Salary=bp + HRA + TA
Print Salary

**Source Code :**

```
bp = input("Enter the basic pay: ")
hra = 0.1*float(bp)
ta = 0.05*float(bp)
salary = float(bp)+hra+ta
print("\nSalary of the Employee =",salary)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle1$ python3 program5.py
Enter the basic pay:10000
Salary of the Employee= 11500.0
```

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

**Experiment No : 6**

**Date : 4/10/2023**

**Aim:**

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

**Pseudocode :**

Input numbers n1, n2.
Print n1+n2
Print n1-n2
Print n1*n2
Print n1/n2

**Source Code :**

```
a = int(input("Enter number1: "))
b = int(input("Enter number2: "))
print("Sum = ",a+b)
print("Differences=", a-b)
print("Product=",a*b)
print("Divison=",a/b)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle1$ python3 program6.py
Enter number1:20
Enter number2:10
Sum= 30
Differences= 10
Product= 200
Division= 2.0
```

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

**Experiment No : 7**
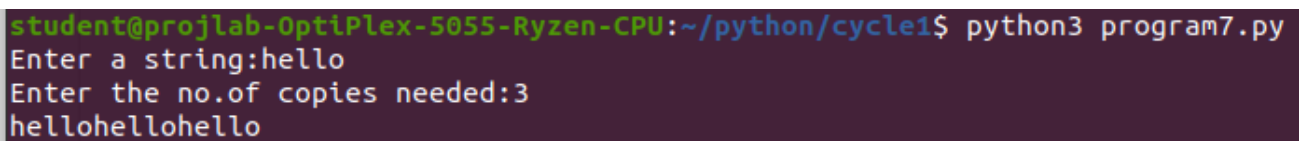
**Date :  4/10/2023**

**Aim:**

Write a Python program to get a string which is n (non-negative integer) copies of a given string.

**Pseudocode :**

Input string s.
Input no of times of repetition r.
Print s*r.

**Source Code :**

```
s = input("Enter a string : ")
r = int(input("Enter the no.of copies : "))
if(n<0):
   Print("Enter a non -ve integer")
else:
print(s*r)
```

**Output:**



```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle1$ python3 program7.py
Enter a string:hello
Enter the no.of copies needed:3
hellohellohello
```

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

**Experiment No : 8**

**Date :  4/10/2023**

**Aim:**

Write a Python program to find biggest of three numbers entered.

**Pseudocode :**

Input three numbers n1,n2,n3.
Check n1<n2 and n3< n2
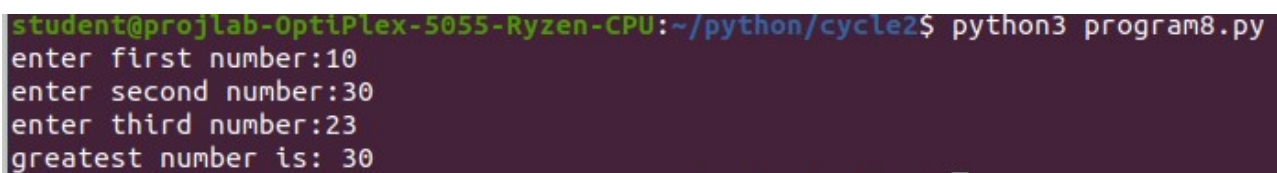Print n2 is greater.
Check n1<n3
Print n3 is greater.
Else
Print n1 is greater.

**Source Code :**

```
n1=int(input("Enter the 1st number"))
n2=int(input("Enter the 2nd number"))
n3=int(input("Enter the 3rd number"))
if n1<n2 and n3<n2:
  print("THE LARGEST :",n2)
elif n1<n3:
  print("THE LARGEST :",n3)
else:
   print("THE LARGEST :",n1)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program8.py
enter first number:10
enter second number:30
enter third number:23
greatest number is: 30
```

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

8

**Experiment No : 9**

**Date :  4/10/2023**

**Aim:**

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

**Pseudocode :**

Input n.
sum=int(n)+int(n*2)+int(n*3)
Print sum

**Source Code :**

n = input("Enter a no. : ")
sum=int(n)+int(n*2)+int(n*3)
Print(n,"+",n*2,"+",n*3)
print("Sum = ",sum)

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program9.py
Enter an integer:10
10 + 1010 + 101010
Sum= 102030
```

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

**Experiment No : 10**

**Date : 12/10/2023**

**Aim:**

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

**Pseudocode :**

Input string str.
Length of str, n=len(str)
Print rearranged string sliced using index.
str[n-1]+str[1:n-1]+str[0]

**Method:**

| Function | Description | Syntax |
|----------|-------------|--------|
| Slicing | Returns a range of characters using slice syntax | Str[a:b]<br>{a->start index<br>b->end index} |
| len() | Returns length of a string | len(x) |

**Source Code :**

```
str=input("Enter a string:")
n=len(str)
print("Modified string is : ",str[n-1]+str[1:n-1]+str[0])
```

**Output:**



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

10

**Experiment No : 11**

**Date :  12/10/2023**

**Aim:**

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

**Pseudocode :**

nitialize list x=[]
Input number of elements in the list, n.
For i=0 to n-1:
 Input integer a
 If a<=100:
 Append a to x
 Else:
 Append "over" to x
 End If
Next i
Print x

 **Method:**

| Function | Description | Syntax |
|---|---|---|
| for | A for loop is used for iterating over a sequence | for i in sequence: statement(s) |
| range | Returns a sequence of numbers | range(n) {default:0} |
| append | Used to add an item to the end of the list | list_name.append("str") |

**Source Code :**

```
x=[]
n=int(input("Enter no.of elements in list : "))
for i in range(0,n):
 a = int(input("Enter element : "))
 if a <= 100:
 x.append(a)
 else:
 x.append("over")
print("Resulting List : ", x)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program11.py
Enter no.of elements in list:5
Enter element:1
Enter element:200
Enter element:2
Enter element:300
Enter element:3
LIST: [1, 'over', 2, 'over', 3]
```

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

**Experiment No : 12**

**Date :  12/10/2023**

**Aim:**

Write a Python program to Store a list of first names. Count the occurrences of 'a' within the list.

**Pseudocode :**

Initialize list x=[], temp=0
Input number of names in list, n.
For i=0 to n:
 Input string(name) a
 Spilt a and store its Ist name to a itself.
 Append a to x.
 Count "a" in a and add it to temp.
Next i
Print x
Print temp

 **Method:**

| Function | Description | Syntax |
|---|---|---|
| count | Return the no of times a specified value appears in the string. | string.count(value) |

**Source Code :**

```
x=[]
temp=0
n=int(input("Enter no.of names in list : "))
for i in range(0, n):
 a = input("Enter the name : ")
 x.append(a)
 temp += x[i].count("a")
print("The list is: ", x)
print("No.of a in the list: ", temp))
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program12.py
ENTER THE NO.OF NAMES:3
ENTER THE NAME:arjun
ENTER THE NAME:jithin
ENTER THE NAME:allen
no.of 'a&A' in the list: 2
```

**Result:** The program is successfully executed and the output is verified

14

**Experiment No : 13**

**Date : 12/10/2023**

**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 :**

Initialize list L1[], L2[]
Input number of elements in list1, n1
For i=0 to n1:
 Input number,a
 Append a to L1
Next i
Input number of elements in list2, n2
For i=0 to n1:
 Input number,a
 Append a to L2
Next i
Print L1, L2
If length of L1=Length of L2:
Print L1=L2
Else:
Print L1!=L2
End If
If Sum of L1=Sum of L2:
Print sum(L1)=sum(L2)
Else:
Print sum(L1)!=sum(L2)
End If
Print element that occur in both list
For element in L1:
 If element in L2:
Print(element)
 Else:
Print none
 End If
Next element

**Source Code :**

List1=[]
List2=[]
common=[]

15

```python
n1=int(input("ENTER THE NO.OF VALUES IN LIST1"))
n2=int(input("ENTER THE NO.OF VALUES IN LIST2"))
print("ENTER THE INTEGER VALUES IN 1ST LIST")
i=0
for i in range(n1):
 num=int(input(""))
 List1.append(num)
print("ENTER THE INTEGER VALUES IN 2ND LIST")
i=0
for i in range(n2):
 num=int(input(""))
 List2.append(num)
if len(List1)==len(List2):
  print("BOTH LIST HAVE SAME LENGTH.")
else:
  print("BOTH LIST HAVE DIFFERENT LENGTH.")
if sum(List1)==sum(List2):
 print("SUM OF BOTH LIST ARE SAME")
else:
 print("SUM OF BOTH LIST ARE DIFFERENT")
for value1 in List1:
  for value2 in List2:
    if value1==value2:
      common.append(value1)
if common:
 print("COMMON VALUES ARE:",common)
else:
 print("NO COMMON VAULES")
```

**Output:**



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

**Experiment No : 14**

**Date :  12/10/2023**

**Aim:**

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

**Pseudocode :**

Input a string str
Initialize dictionary dict
Spilt str and store it as list str itself
For x in str:
 If x in dict:
Add value by 1 of that key
Else:
Assign 1 to key( i.e., x )
 End If
Next x
Print dict

**Method**

| Function | Description | Syntax |
|---|---|---|
| spilt | Used to spilt a string into list {default separator:whitespace} | string.spilt() |
| Dictionary | Used to store data value in key:value pairs. Items can be referred using keyname. | dict={key:value} |

**Source Code :**

```
input_line=input("ENTER THE LINE OF TEXT")
words=input_line.split()
word_count={}
for word in words:
  word=word.strip('.,!?').lower()
  word_count[word]=word_count.get(word,0)+1
print("WORD OCCURRENCS:",word_count)
```

**Output:**



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

17

**Experiment No : 15**

**Date : 12/10/2023**

**Aim:**

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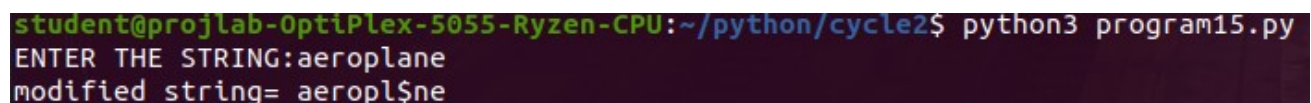
**Pseudocode :**

Input string str
Initialize a as 0th index of str
Replace all occurrence of a as '$' from 1st index of str
Print new string by concatenating a and replaced new string

**Source Code :**

```
string1=input("ENTER THE STRING")
first=string1[0]
modified_string=first
for char in string1[1:]:
  if char==first:
    modified_string+="$"
  else:
    modified_string+=char
print("modified string=",modified_string)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program15.py
ENTER THE STRING:aeroplane
modified string= aeropl$ne
```

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

**Experiment No : 16**

**Date :  12/10/2023**

**Aim:**

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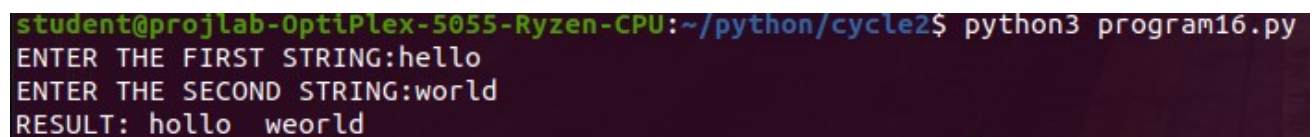
**Pseudocode :**

Input string s1,s2
Initialize new_str by slicing and concatenating s1 and s2
Print new_str

**Source Code :**

```
str1=input("ENTER THE FIRST STRING")
str2=input("ENTER THE SECOND STRING")
str3=str1[0]+str2[1]+str1[2:]+"  "+str2[0]+str1[1]+str2[1:]
print("RESULT:",str3)
```

**Output:**



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

**Experiment No : 17**

**Date :  12/10/2023**

**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.

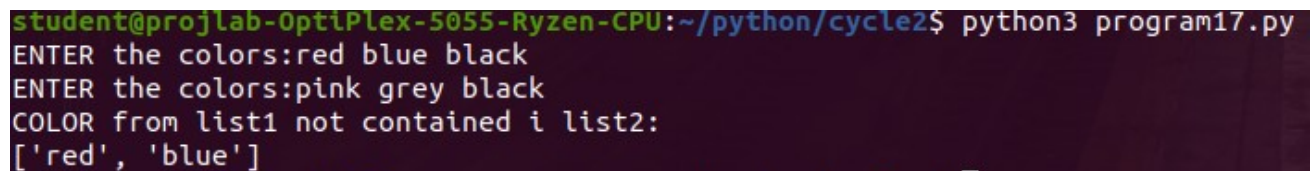**Pseudocode :**

Input string s1, s2
Convert s1 and s2 as list using split
Find unique= set(s1) - set(s2)
Print unique

**Source Code :**

```
color_list1=input("ENTER the colors").split()
color_list2=input("ENTER the colors").split()
unique_color=[]
for color in color_list1:
  if color not in color_list2:
    unique_color.append(color)
print("COLOR from list1 not contained i list2:")
print(unique_color)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program17.py
ENTER the colors:red blue black
ENTER the colors:pink grey black
COLOR from list1 not contained i list2:
['red', 'blue']
```

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

20

**Experiment No : 18**

**Date :  12/10/2023**

**Aim:**

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

**Pseudocode :**

Input string s
s is split with ',' and stored in s as list
Print first element s[0]
Print last element s[-1]

**Source Code :**

```
color=input("Enter comma seperated color")
color_list=color.split(',')
print("First color:",color_list[0].strip())
print("Last color:",color_list[-1].strip())
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program18.py
Enter comma seperated color:red,blue,pink,violet,green
First color: red
Last color: green
```

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

**Experiment No : 19**

**Date :  12/10/2023**

**Aim:**

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

**Pseudocode :**

Input list of integers into s
Spilt s
Print list
For x in s:
 If (x%2!=0):
 l=x
 End If
Next x
Print list l

**Method**

| Function | Description | Syntax |
|---|---|---|
| List comprehension | It offers a shorter syntax when you want to create a new list based on the values of an existing list | newlist=[expression for item in iterable if condition==True] |

**Source Code :**

```
numbers=[]
n=int(input("Enter the no.of values"))
i=0
for i in range(n):
  val=int(input("Enter the number"))
  numbers.append(val)
odd_numbers=[]
for num in numbers:
  if num%2!=0:
    odd_numbers.append(num)
print("LIST OF NUMBERS AFTER REMOVING EVEN NUMBERS:",odd_numbers)\
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program19.py
Enter the no.of values:5
Enter the number:2
Enter the number:1
Enter the number:5
Enter the number:7
Enter the number:8
LIST OF NUMBERS AFTER REMOVING EVEN NUMBERS: [1, 5, 7]
```

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

**Experiment No : 20**

**Date :  26/10/2023**

**Aim:**

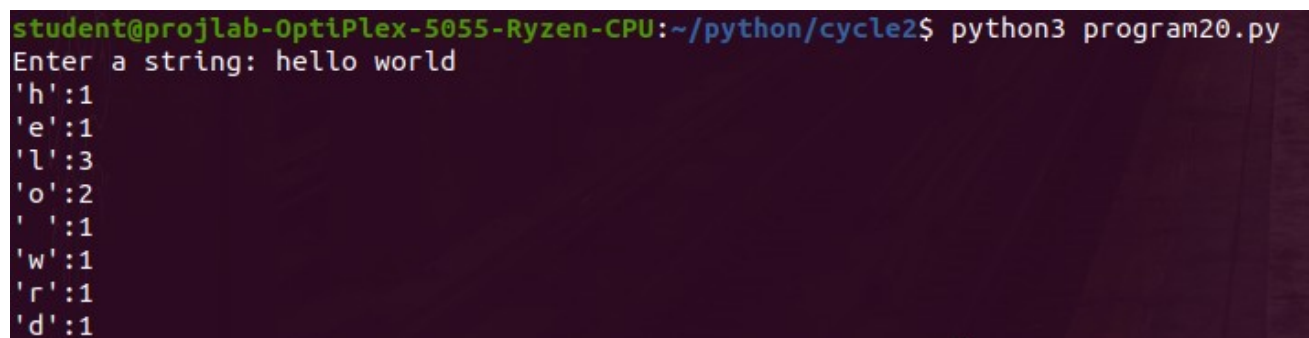Count the number of characters (character frequency) in a string.

**Pseudocode :**

Input string str
Initialize dictionary dict
keys=List of keys of dict
For x in str:
 If(x in keys):
dict(x)=dict(x)+1
 Else:
dict(x)=1
 End If
Next x
Print(dict)

**Source Code :**

```
input_string = input("Enter a string: ")
char_frequency = {}
for char in input_string:
    if char in char_frequency:
        char_frequency[char] += 1
    else:
        char_frequency[char] = 1
for char, frequency in char_frequency.items():
    print(f"'{char}':{frequency}")
```

**Output:**



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

**Experiment No : 21**

**Date :  26/10/2023**

**Aim:**

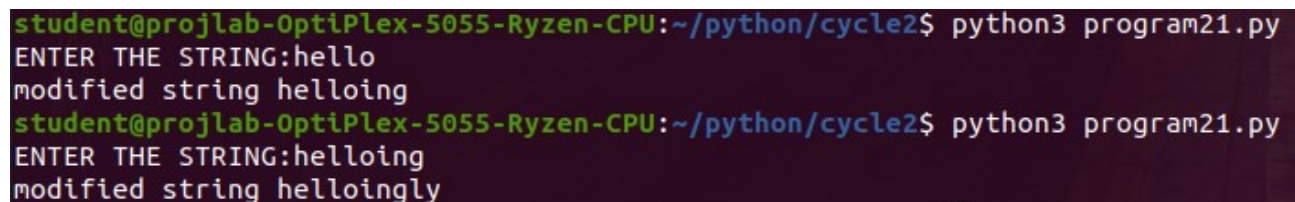Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'.

**Pseudocode :**

Input string str
If str[-3]='ing':
 Print str+'ly'
Else:
 Print str+'ing'
End If

**Source Code :**

```
str=input("ENTER THE STRING:")
if str.endswith("ing"):
  str=str+"ly"
else:
  str=str+"ing"
print("modified string",str)
```

**Output:**



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

**Experiment No : 22**

**Date :  26/10/2023**

**Aim:**

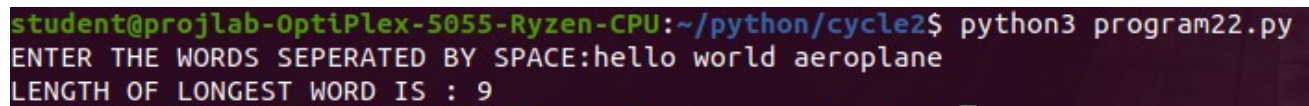Accept a list of words and return the length of the longest word.

**Pseudocode :**

List[]
Read the words into a list splited by space.
For word in list
  If length (word)>longest_word
     longest_word = length(word)
Print longest_word

**Source Code :**

```
list=input("ENTER THE WORDS SEPERATED BY SPACE").split()
long_len=0
for word in list:
  if len(word)>long_len:
    long_len=len(word)
print("LENGTH OF LONGEST WORD IS :",long_len)
```

**Output:**



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

**Experiment No : 23**

**Date :  26/10/2023**

**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 :**
Input str
For (x in split(str)):
 l=int(x)
Next x
Print l

For (x in l):
 If (x>=0):
pos=x
End If
Next x
Print positive numbers, pos

For (x in l):
sq=x2
Next x
Print squares of x, sq

For ch in str:
m=ch
Next x
Print m

For (ch in m):
If (ch=a,e,i,o,u,A,E,I,O,U):
vow=ch
End if
Next ch
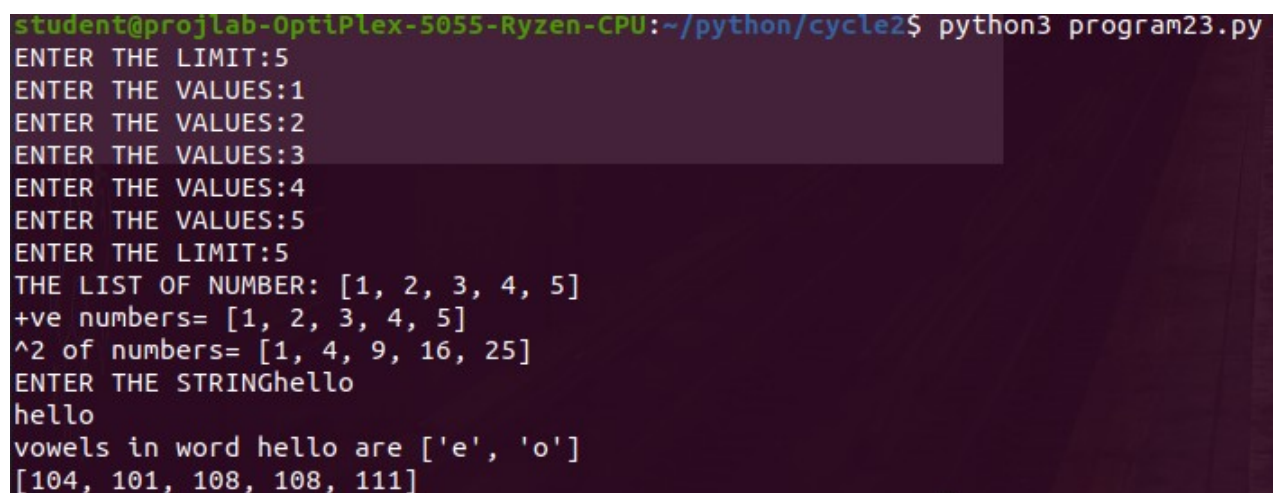Print vowels, vow

For x in m:
 ordi=ord(x)
Print ordinal value, ordi

**Method:**

| Function | Description | Syntax |
|---|---|---|
| ord() | Returns the number representing the unicode code of a specified character | ord(character) |

**Source Code :**

```
list_num=[]
n=int(input("ENTER THE LIMIT:"))
i=0
for i in range(n):
  num=int(input("ENTER THE VALUES:"))
  list_num.append(num)
positive=[num1 for num1 in list_num if num1>0]
N=int(input("ENTER THE LIMIT:"))
squered=[num2 **2 for num2 in range(1,N+1)]
print("THE LIST OF NUMBER:",list_num)
print("+ve numbers=",positive)
print("^2 of numbers=",squered)
word=input("ENTER THE STRING")
print(word)
vowels=[char for char in word if char.lower() in'aeiou']
print("vowels in word",word,"are",vowels)
orginal_val=[ord(char)for char in word]
print(orginal_val)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program23.py
ENTER THE LIMIT:5
ENTER THE VALUES:1
ENTER THE VALUES:2
ENTER THE VALUES:3
ENTER THE VALUES:4
ENTER THE VALUES:5
ENTER THE LIMIT:5
THE LIST OF NUMBER: [1, 2, 3, 4, 5]
+ve numbers= [1, 2, 3, 4, 5]
^2 of numbers= [1, 4, 9, 16, 25]
ENTER THE STRINGhello
hello
vowels in word hello are ['e', 'o']
[104, 101, 108, 108, 111]
```

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

**Experiment No : 24**

**Date :  26/10/2023**

**Aim:**

Sort dictionary in ascending and descending order

**Pseudocode :**

Initialize dictionary dict
Input number of elements num
For i in range(num):
 Input key k
Input value v
Update dict
Next i
Print dict
Sort dict by key in ascending order using sort function
Sort dict by key in descending order by reverse=true
Sort dict by value in ascending and descending order
Print sorted dictionaries

**Method:**

| Function | Description | Syntax |
|----------|-------------|--------|
| items | Returns a view object (contains key-value pairs of dictionary as tuple in list) | dictionary.items() |
| sorted | Returns a sorted list of specified iterable object. [Iterable(Req) => Seq to sort; key(optional)=>fn to execute to decide order] | sorted(iterable, key=key, reverse=reverse) |

**Source Code :**

```
dictionary={}
N=int(input("Enter the no.of values"))
i=0
for i in range(N):
  new_key=input("Enter the key shouid be a alphabet")
  new_val=int(input("Enter the value number"))
  dictionary[new_key]=new_val
ascend_dict=dict(sorted(dictionary.items()))
print("Dictionary in ascending order by keys:",ascend_dict)
descend_dict=dict(sorted(dictionary.items(),reverse=True))
print("Dictionary in descening order by keys:",descend_dict)
```

29

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program24.py
Enter the no.of values:5
Enter the key .it should be a alphabeta
Enter the value number:1
Enter the key .it should be a alphabetb
Enter the value number:2
Enter the key .it should be a alphabeth
Enter the value number:5
Enter the key .it should be a alphabetr
Enter the value number:7
Enter the key .it should be a alphabetu
Enter the value number:34
Dictionary in ascending order by keys: {'a': 1, 'b': 2, 'h': 5, 'r': 7, 'u': 34}
Dictionary in descening order by keys: {'u': 34, 'r': 7, 'h': 5, 'b': 2, 'a': 1}
```

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

**Experiment No : 25**

**Date :  26/10/2023**

**Aim:**

Merge two dictionaries

**Pseudocode :**

Initialize dictionaries dict1, dict2
Print "Dictionary 1"
Input number of elements num
For i in range (num):
 Input key k
 Input value v corresponding to k
Next i
Print "Dictionary 2"
Merged_dict=dict1.copy()
Merged_dict.update(dict2)
Print("Merged dictionary:",merged_dict)

**Source Code :**

```
dict1={}
dict2={}
n1=int(input("ENTER THE LIMIT OF FIRST DICTIONARY:"))
i=0
for i in range(n1):
 new_key=input("ENTER THE ALPHABET KEY")
 new_value=input("ENTER THE VALUE INTEGER")
 dict1[new_key]=new_value
n2=int(input("ENTER THE LIMIT OF SECOND DICTIONARY:"))
i=0
for i in range(n2):
 new_key=input("ENTER THE ALPHABET KEY")
 new_value=input("ENTER THE VALUE INTEGER")
 dict2[new_key]=new_value

merged_dict=dict1.copy()
merged_dict.update(dict2)
print("dictionary1:",dict1)
print("dictionary2:",dict2)
print("Merged dictionary:",merged_dict)
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle2$ python3 program25.py
ENTER THE LIMIT OF FIRST DICTIONARY:5
ENTER THE ALPHABET KEYa
ENTER THE VALUE INTEGER12
ENTER THE ALPHABET KEYb
ENTER THE VALUE INTEGER345
ENTER THE ALPHABET KEYi
ENTER THE VALUE INTEGER876
ENTER THE ALPHABET KEYr
ENTER THE VALUE INTEGER121
ENTER THE ALPHABET KEYg
ENTER THE VALUE INTEGER675
ENTER THE LIMIT OF SECOND DICTIONARY:3
ENTER THE ALPHABET KEYn
ENTER THE VALUE INTEGER11
ENTER THE ALPHABET KEYm
ENTER THE VALUE INTEGER66
ENTER THE ALPHABET KEYv
ENTER THE VALUE INTEGER88
dictionary1: {'a': '12', 'b': '345', 'i': '876', 'r': '121', 'g': '675'}
dictionary2: {'n': '11', 'm': '66', 'v': '88'}
Merged dictionary: {'a': '12', 'b': '345', 'i': '876', 'r': '121', 'g': '675', 'n': '11', 'm': '66', 'v': '88'}
```

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

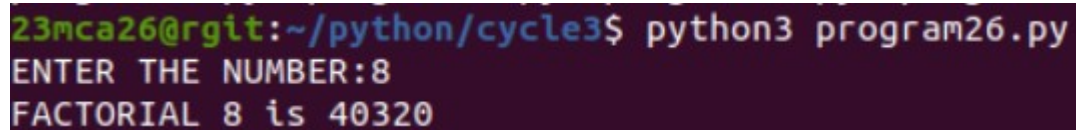**Experiment No : 26**

**Date : 26/10/2023**

**Aim:**

Write a program to find the factorial of a number

**Pseudocode :**

Input number n
Initialize f=1
 For i in 1 to n+1:
 f=f*i
 Next i
Print fact

**Source Code :**

```
num=int(input("ENTER THE NUMBER"))
fact=1
for i in range(1,num+1):
  fact*=i
print("FACTORIAL of ",num,"is:",fact)
```

**Output:**



```
23mca26@rgit:~/python/cycle3$ python3 program26.py
ENTER THE NUMBER:8
FACTORIAL 8 is 40320
```

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

**Experiment No : 27**

**Date :  26/10/2023**

**Aim:**

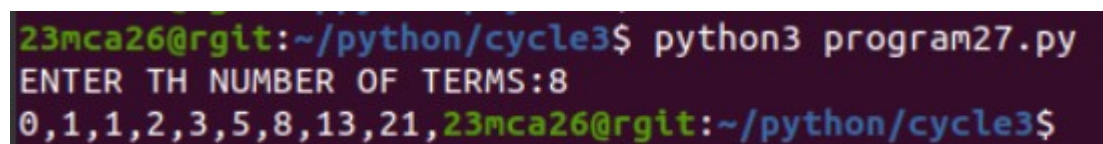Write a program to generate Fibonacci series of N terms

**Pseudocode :**

Input number x
Initialize n1=0, n2=1, count=0
If x<=0:
 Print "Enter positive no"
Else if x==1:
 Print n1 as Fibonacci sequence
Else:
 While count<x:
 Print n1
 nt=n1+n2
 n1=n2
 n2=nt
 count=count+1
End if

**Source Code :**

```
n=int(input("ENTER TH NUMBER OF TERMS"))
first_term=0
second_term=1
i=0
while(i<=n):
  print(first_term,end=",")
  nth=first_term+second_term
  first_term=second_term
  second_term=nth
  i=i+1
```

**Output:**



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

**Experiment No : 28**

**Date :  26/10/2023**

**Aim:**

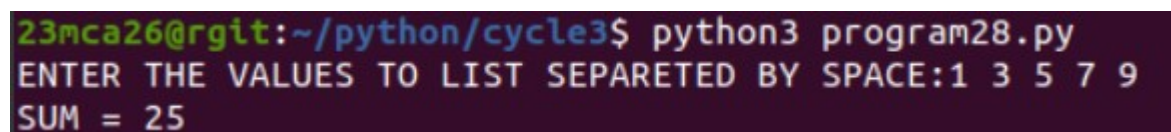Write a program to find the sum of all items in a list [Using for loop]

**Pseudocode :**

Input a list of comma separated integers str
For (x in split(str)):
 l=int(x)
Next x
Print l
Initialize sum=0
For i in l:
 sum = sum + i
Next i
Print sum

**Source Code :**

```
num_list=input("ENTER THE VALUES TO LIST SEPARETED BY SPACE:").split()
num_list1=[int(num)for num in num_list]
total=0
for num in num_list1:
  total+=num
print("SUM =",total)
```

**Output:**



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

35

**Experiment No : 29**

**Date : 6/11/2023**

**Aim:**

Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square

**Pseudocode :**

Initialize an empty list 'list'
Input limit l
If l>=1000 and l<10000:
 For i=1000 to l:
 If (floor(i^0.5)^2)=i:
 If all(int(b)%2=0 for b in str(i)):
 Insert i to list
 End If  End If
 Next i
Else: Print "Invalid range"
End If
Print list

**Source Code :**

```
start=int(input("Enter the starting number : "))
end=int(input("Enter the ending number : "))
fourdigit=[]
if start<1000 or start>9999 or end<1000 or end>9999:
     print("Please enter a four digit number !! " )
else:
     for num in range(start,end+1):
          tmp=num
          if num%2==0 and int(num**0.5)**2==num:
               num=int(num/10)
               if num%2==0:
                    num=int(num/10)
                    if num%2==0:
                         num=int(num/10)
                         if num%2==0:
                              fourdigit.append(tmp)
print(fourdigit)
```
**Output:**

```
23mca26@rgit:~/python/cycle3$ python3 program29.py
Enter the starting number : 1000
Enter the ending number : 9999
[4624, 6084, 6400, 8464]
```

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

36

**Experiment No : 30**

**Date : 6/11/2023**

**Aim:**

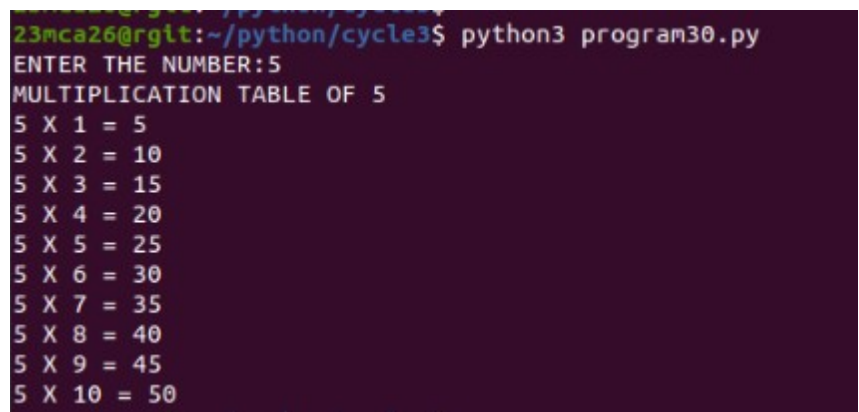Write a program using a for loop to print the multiplication table of n, where n is entered by the user.

**Pseudocode :**

Input number num
For i=1 to 10:
 Print 'i'*'num'='num*i'
Next i

**Source Code :**

```
n=int(input("ENTER THE NUMBER"))
print("MULTIPLICATION TABLE OF",n)
for i in range(1,11):
  print(n,"X",i,"=",n*i)
```

**Output:**

```
23mca26@rgit:~/python/cycle3$ python3 program30.py
ENTER THE NUMBER:5
MULTIPLICATION TABLE OF 5
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45
5 X 10 = 50
```

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

**Experiment No : 31**

**Date :  6/11/2023**

**Aim:**

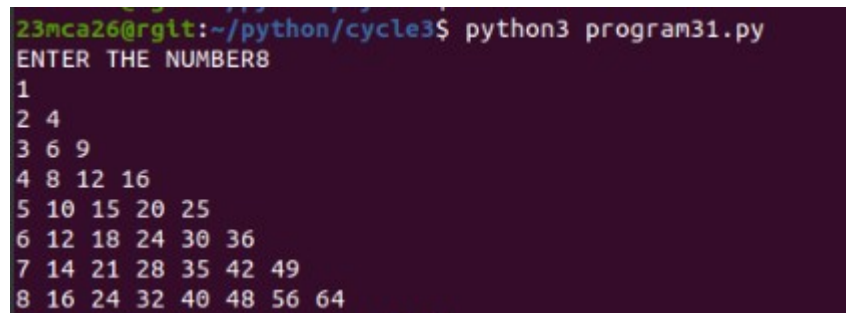Display the given pyramid with the step number accepted from the user.

**Pseudocode :**

Input num
For i=1 to num:
 For j=1 to i:
 Print(j*i, end=" ")
 Print()
 Next j

**Source Code :**

```
n=int(input("ENTER THE NUMBER"))
for i in range(1,n+1):
  for j in range(i):
    print(i*(j+1),end="")
  print()
```

**Output:**



```
23mca26@rgit:~/python/cycle3$ python3 program31.py
ENTER THE NUMBER8
1
2 4
3 6 9
4 8 12 16
5 10 15 20 25
6 12 18 24 30 36
7 14 21 28 35 42 49
8 16 24 32 40 48 56 64
```

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

38

**Experiment No : 32**

**Date :  6/11/2023**

**Aim:**

Write a program to generate all factors of a number [use while loop].

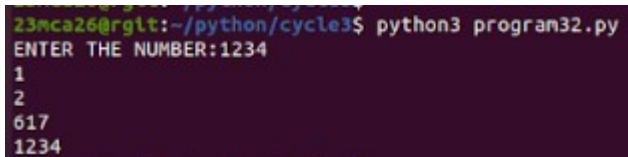**Pseudocode :**

Input num
For i=1 to num:
 If num%i==0:
 Print i
 End if
Next i

**Source Code :**

```
num=int(input("ENTER THE NUMBER"))
divisor=1
while divisor<=num:
  if num % divisor==0:
    print(divisor)
  divisor+=1
```

**Output:**



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

**Experiment No : 33**
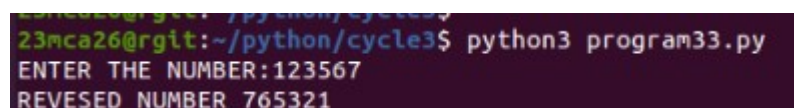
**Date : 6/11/2023**

**Aim:**

Write a program to print reverse of a number [use while loop].

**Pseudocode :**

Input number num
Initialize rev=0
while num!=0:
 remainder=num%10
 rev=rev*10+remainder
 num=num//10
End While
Print rev

**Source Code :**

```
num=int(input("ENTER THE NUMBER"))
reverse_num=0
while num>0:
  digit=num%10
  reverse_num=reverse_num*10+digit
  num=num//10
print("REVESED NUMBER",reverse_num)
```

**Output:**



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

**Experiment No : 34**

**Date : 6/11/2023**

**Aim:**

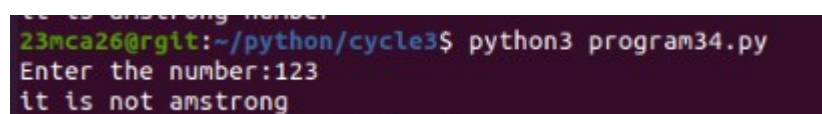Write a program to find whether the given number is an Armstrong or not [use while loop].

**Pseudocode :**

Input num
Store num to x
Initialize sum=0
While num>0:
 remainder=num%10
 sum=sum+(remainder)^3
num=num//10
End while
If sum==x:
 Print x is Armstrong number
Else:
 Print x is not Armstrong number
End if

**Source Code :**

```
num=int(input("Enter the number:"))
num_cpy=num
num_dig=0
while num_cpy>0:
  num_cpy //=10
  num_dig+=1
num_cpy=num
amstrong_sum=0
while num_cpy >0:
  digit=num_cpy%10
  amstrong_sum+=digit**num_dig
  num_cpy //=10
if amstrong_sum==num:
  print("it is amstrong number")
else:
  print("it is not amstrong")
```

**Output:**



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

**Experiment No : 35**

**Date :  6/11/2023**

**Aim:**

Display star pattern using nested loop

**Pseudocode :**

Input number of rows n
For i=0 to n-1:
 For j=0 to i:
 Print *," "
 Print " "
 Next j
Next i
For i=n to 1:
 For j=0 to i-2:
 Print *," "
 Print " "
 Next j
 i--
Next I

**Source Code :**

```
num=5
for i in range(num):
  for j in range(i+1):
    print("*",end=" ")
  print()
for i in range(num-1,0,-1):
  for j in range(i):
    print("*",end=" ")
  print()
```
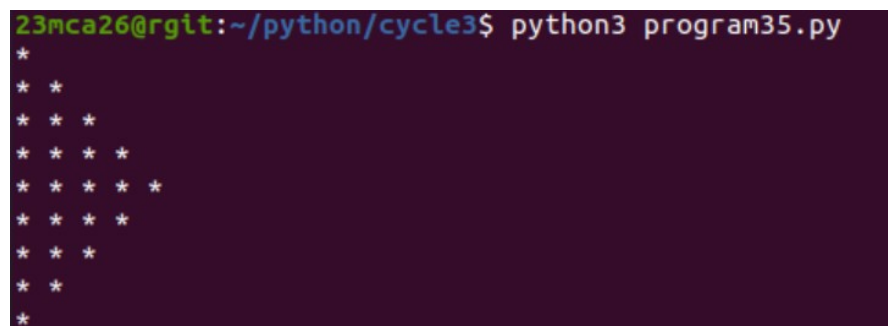**Output:**



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

**Experiment No : 36**

**Date :  6/11/2023**

**Aim:**

Write a program using functions to calculate the simple interest. Suppose the customer is a senior citizen. He is being offered a 12 percent rate of interest, for all other customers, the rate of interest is 10 percent.

**Pseudocode :**

Function si (p,r,t):
 Print (p*r*t)/100
End Function
Input name
Input age
Input principal amount, p
Input no.of years, t
If age>60:
 r=12
Else:
 r=10
End If
Print r
si (p,r,t)

**Source Code :**

```
def senior():
    amount=int(input("Enter the amount:"))
    time=int(input("Enter the duration:"))
    interest=(amount*12*time)/100
    print("Simple interest=",interest)
def other():
    amount=int(input("Enter the amount:"))
    time=int(input("Enter the duration:"))
    interest=(amount*10*time)/100
    print("Simple interest=",interest)

age=int(input("Enter the age:"))
if age>50:
    print("YOU ARE SENIOR CITIZEN")
    senior()
else:
    print("YOU ARE NOT SENIOR CITIZEN")
    other()
```

43

**Output:**

```
23mca26@rgit:~/python/cycle4$ python3 program36.py
Enter the age:30
YOU ARE NOT SENIOR CITIZEN
Enter the amount:12000
Enter the duration:8
Simple interest= 9600.0
```

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

44

**Experiment No : 37**
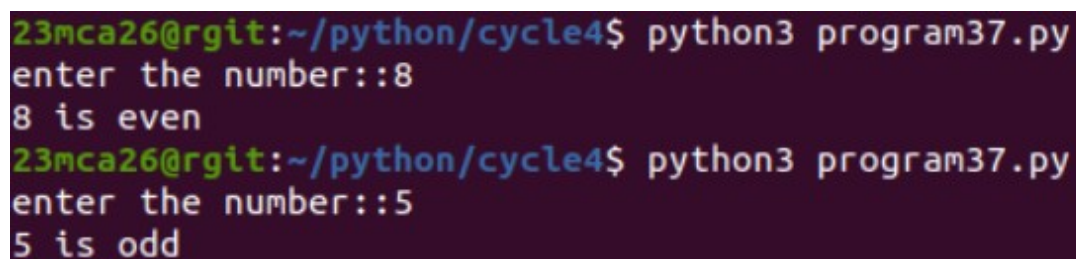
**Date : 16/11/2023**

**Aim:**
Write a program using functions and return statements to check whether a number is even or odd.

**Pseudocode :**

```
Function evenodd(num):
 If num%2=0:
 Return even
 Else:
 Return odd
 End If
End Function
Input num
Print evenodd(num)
```

**Source Code :**

```python
def evenorodd(number):
  if(number%2==0):
    return 1
  else:
    return 0
number=int(input("enter the number::"))
n=evenorodd(number)
if(n==1):
 print(number,"is even")
else:
 print(number,"is odd")
```

**Output:**



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

**Experiment No : 38**

**Date : 16/11/2023**

**Aim:**
Write a function called compare which takes two strings S1 and S2 and an integer n as arguments.
The function should return True if the first n characters of both the strings are the same else the
function should return False.

**Pseudocode :**

Function compare (s1,s2,n):
 If s1[:n]=s2[:n]:
 Return True
 Else:
 Return False
 End If
End Function
Input strings s1, s2
Input comparison limit, n
Print compare(s1,s2,n)

**Source Code :**

```
def compare(s1,s2,n):
  for x in range(0,n):
   if s1[x]==s2[x]:
    return "true"
   else:
    return "false"
s1=input("Enter first string:")
s2=input("Enter second string:")
n=int(input("enter n:"))
res=compare(s1,s2,n)
print(res)
```

**Output:**



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

**Experiment No : 39**

**Date : 16/11/2023**

**Aim:**

Write a program to print the Fibonacci series using recursion
.
**Pseudocode :**
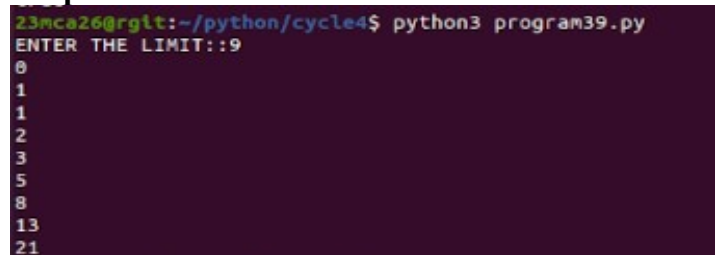
Function fibo(n):
If n<=1:
 Return n
 Else:
 Return (fibo(n-1) + fibo(n-2))
 End If
End Function

Input no. of terms, nterms
If nterms <= 0:
Print "Enter positive no."
Else:
 Print "Fibonacci Sequence"
 For i = 0 to nterms:
 Print Fibo(i)
 Next i
End If

**Source Code :**

```
def fibonacci(first,second,limit):
    if limit>=1:
     print(first)
     nth=second+first
     first=second
     second=nth
     limit=limit-1
     fibonacci(first,second,limit)
limit=int(input("ENTER THE LIMIT::"))
fibonacci(0,1,limit)
```
**Output:**



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

**Experiment No : 40**

**Date : 16/11/2023**

**Aim:**

Write a program to add variable length integer arguments passed to the function.

**Pseudocode :**

```
Function add (*args):
 sum=0
 For i in args:
 sum = sum + i
 Next i
 Print sum
add (num1, num2, num3)
add (num1, num2, num3, num4, num5)
```
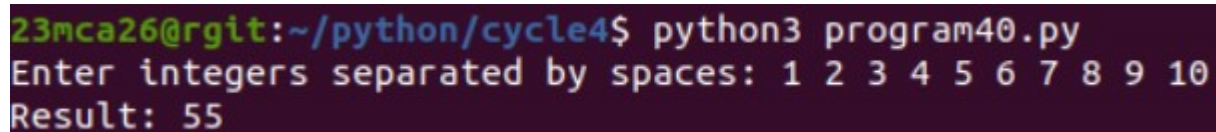
**Source Code :**

```
def add_integers(*args):
    """
    Adds variable-length integer arguments.

    Parameters:
    *args (int): Variable number of integer arguments.

    Returns:
    int: Sum of the integer arguments.
    """
    total = 0
    for num in args:
        if isinstance(num, int):
            total += num
        else:
            raise TypeError("Arguments must be integers.")
    return total
result = add_integers(1, 2, 3, 4, 5)
print(f"Result: {result}")
```

**Output:**



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

**Experiment No : 41**

**Date : 16/11/2023**

**Aim:**

Write lambda functions to find the area of square, rectangle and triangle.

**Pseudocode :**

Initialize ar1 as lambda x:x*x
Initialize ar2 as lambda x,y:x*y
Initialize ar3 as lambda x,y:0.5*x*y
Input length of square, a
Print ar1(a)
Input length of rectangle, l
Input breadth of rectangle, b
Print ar2(l, b)
Input base of triangle, b
Input height of triangle, h
Print ar3(b, h)

**Method:**

| Function | Description | Syntax |
|----------|-------------|--------|
| Lambda | A lambda function can take any number of arguments, but can only have one expression. | lambda arguments: expression |

**Source Code :**

```
triangle=lambda b,h:.5*b*h
rectangle=lambda l,b:l*b
square=lambda s:s*s
base=int(input("Enter Base"))
height=int(input("Enter Height"))
print("AREA OF TRIANGLE")
print(triangle(base,height))
length=int(input("Enter Length"))
breadth=int(input("Enter Breadth"))
print("AREA OF RECTANGLE")
print(rectangle(length,breadth))
side=int(input("Enter Side"))
print("AREA OF SQUARE")
print(square(side))
```

**Output:**

```
23mca26@rgit:~/python/cycle4$ python3 program41.py
Enter Base:6
Enter Height:7
AREA OF TRIANGLE
21.0
Enter Length:15
Enter Breadth:9
AREA OF RECTANGLE
135
Enter Side7
AREA OF SQUARE
49
```

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

**Experiment No : 42**

**Date : 23/11/2023**

**Aim:**

Write a program to display powers of 2 using anonymous function

**Pseudocode :**

Input no. of terms, n
Store map (lambda x:2**x, range(n)) as list into result
Print n
For i = 0 to n:
 Print result[i]
Next i

**Source Code :**

```
display_powers_of_2 = lambda n:list(map(lambda x: 2**x, range(n)))
num_of_powers =int(input("Enter the number of powers needed:"))
result = display_powers_of_2(num_of_powers)
print(f"Powers are:",result)
```

**Output:**



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

**Experiment No : 43**

**Date : 23/11/2023**

**Aim:**

Write a program to sum the series $1/1! + 4/2! + 27/3! + ..... +$ nth term.

**Pseudocode :**

Function fact(n):
 If n=1:
 Return 1
 Else:
 Return n*fact(n-1)
 End If
End Function
Input number of terms, n
Initialize result as 0
For i = 1 to n:
 f = fact(i)
 result = result + (Power(i,i)/f)
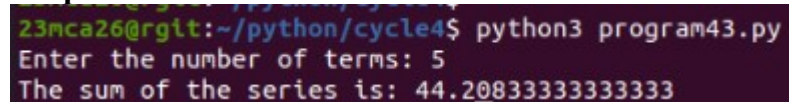Next i
Print result

**Source Code :**

```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)

def series_sum(n):
    result = 0
    for i in range(1, n + 1):
        term = i ** i / factorial(i)
        result += term
    return result
n = int(input("Enter the number of terms: "))
result = series_sum(n)
print(f"The sum of the series is: {result}")
```

**Output:**



```
23mca26@rgit:~/python/cycle4$ python3 program43.py
Enter the number of terms: 5
The sum of the series is: 44.20833333333333
```

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

52

**Experiment No : 44**

**Date : 23/11/2023**

**Aim:**

Write a program to determine whether a given year is a leap year [ Use Calendar Module].

**Pseudocode :**

Import calendar module
Input year
If calendar.isleap(year):
 Print 'year' is a leap year
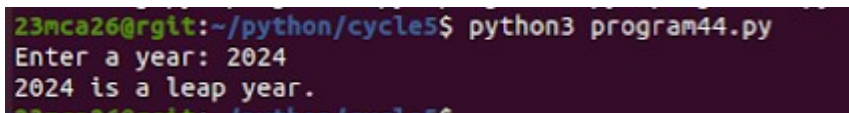Else:
 Print 'year' is not a leap year
End If

**Method**

| Function | Description | Syntax |
|----------|-------------|--------|
| Calendar | Built-in module in Python which allows you to perform date, month, and calendar-related operations. | import calendar |
| Isleap | Year to be tested leap or not. | isleap(year) |

**Source Code :**

```
import calendar
def is_leap_year(year):
    return calendar.isleap(year)
year = int(input("Enter a year: "))
if is_leap_year(year):
    print(f"{year} is a leap year.")
else:
    print(f"{year} is not a leap year.")
```

**Output:**



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

**Experiment No : 45**

**Date : 23/11/2023**

**Aim:**

Write a python script to display
a) Current date and time
b) Current Year
c) Month of the year
d) Week number of the year
e) Weekday of the week
f) Day of year
g) Day of the month
h) Day of week [ Use time and datetime Module]

**Pseudocode :**

Import time module
Import datetime module
Print Current date and time: datetime.datetime.now()
Print Current year: tdy.strftime("%Y")
Print Month of the year: tdy.strftime("%B")
Print Week number of the year: tdy.strftime("%W")
Print Weekday of the week: tdy.strftime("%w")
Print Day of year: tdy.strftime("%j")
Print Day of the month: tdy.strftime("%d")

**Source Code :**

```
import time
import datetime
current_time = datetime.datetime.now()
print("a) Current date and time:", current_time)
current_year = current_time.year
print("b) Current Year:", current_year)
month = current_time.strftime("%B")
print("c) Month of the year:", month)
week_number = current_time.strftime("%U")
print("d) Week number of the year:", week_number)
weekday = current_time.strftime("%A")
print("e) Weekday of the week:", weekday)
day_of_year = current_time.strftime("%j")
print("f) Day of year:", day_of_year)
day_of_month = current_time.strftime("%d")
print("g) Day of the month:", day_of_month)
day_of_week = current_time.strftime("%w")
print("h) Day of week (0 - Sunday, 1 - Monday, ...):", day_of_week)
```

**Output:**

```
23mca26@rgit:~/python/cycle5$ python3 program45.py
a) Current date and time: 2023-12-15 11:45:36.259200
b) Current Year: 2023
c) Month of the year: December
d) Week number of the year: 50
e) Weekday of the week: Friday
f) Day of year: 349
g) Day of the month: 15
h) Day of week (0 - Sunday, 1 - Monday, ...): 5
```

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

**Experiment No : 46**

**Date : 30/11/2023**

**Aim:**

Write a python program to print yesterday, today and tomorrow.

**Pseudocode :**

Import datetime module
Initialize today as datetime.date.today()
Store today - datetime.timedelta(days = 1) into yesterday
Store today + datetime.timedelta(days = 1) into tomorrow
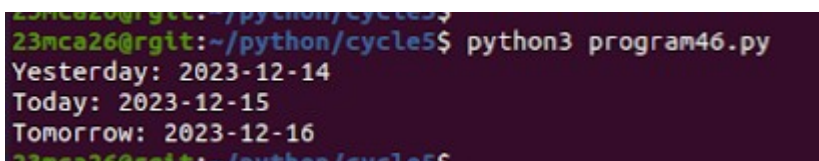Print yesterday
Print today
Print tomorrow

**Method**

| Function | Description | Syntax |
|----------|-------------|--------|
| Timedelta | It is built in function in datetime module which is used for calculating differences in dates and also can be used for date manipulations in Python. | datetime.timedelta(days=0, seconds=0,microseconds=0, milliseconds=0, minutes=0, hours=0, weeks=0) |

**Source Code :**

```
import datetime
today = datetime.date.today()
yesterday = today - datetime.timedelta(days=1)
tomorrow = today + datetime.timedelta(days=1)
print("Yesterday:", yesterday)
print("Today:", today)
print("Tomorrow:", tomorrow)
```

**Output:**



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

**Experiment No : 47**

**Date : 30/11/2023**

**Aim:**

Write a function in file Armstrong.py to check whether a number is an Armstrong number. Import the module to generate Armstrong numbers between two limits.

**Pseudocode :**

Function Amstrong(num):
Sum=0 and temp=num
While temp!=0
rem=temp%10
sum=sum+(rem**3)
temp=temp/10
If sum equals to num
Print num is amstrong number
Else
Print num is not amstrong number
 End If
End Function
Read lower and upper limits
For i from lower limit to upper limit+1
 If Amstrong(i) equals true
 Print i is amstrong
 End If
Next i

**Source Code :**
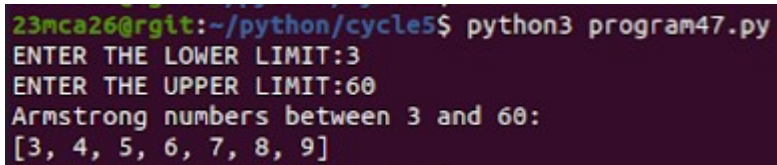
```
from Armstrong import is_armstrong

def generate_armstrong_numbers(lower_limit, upper_limit):
    armstrong_numbers = []
    for num in range(lower_limit, upper_limit + 1):
        if is_armstrong(num):
            armstrong_numbers.append(num)
    return armstrong_numbers

lower_limit =int(input("ENTER THE LOWER LIMIT:"))
upper_limit =int(input("ENTER THE UPPER LIMIT:"))
result = generate_armstrong_numbers(lower_limit, upper_limit)
print(f"Armstrong numbers between {lower_limit} and {upper_limit}:")
print(result)
```

**Armstrong.py**

```
def is_armstrong(number):
    num_str = str(number)
    num_digits = len(num_str)
    sum_of_digits = sum(int(digit) ** num_digits for digit in num_str)
    if sum_of_digits == number:
        return True
    else:
        return False
```

**Output:**

```
23mca26@rgit:~/python/cycle5$ python3 program47.py
ENTER THE LOWER LIMIT:3
ENTER THE UPPER LIMIT:60
Armstrong numbers between 3 and 60:
[3, 4, 5, 6, 7, 8, 9]
```

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

**Experiment No : 48**

**Date : 8/12/2023**

**Aim:**

Create a package graphics with modules rectangle, circle and sub-packagethreeDgraphics with modules cuboid and sphere. Include methods to findarea and perimeter of respective figures in each module. Write programsthat find the area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements)

**Pseudocode :**

Read length and width
 Print rectangle.perimeter(length,width)
 Print rectangle.area(length,width)
Read radius
 Print circle.perimeter(radius)
 Print circle.area(radius)
Read length, width and height
 Print cuboid.area(length,width,height)
 Print cuboid.perimeter (length,breadth,height)
Read radius
 Print sphere.area(radius)
 Print sphere.perimeter(radius)

Create package graphics with modules circle and rectangle
**circle.py**
Function area(r):
 return 3.14*r*r
Function perimeter(r):
 return 2*3.14*r
**rectangle.py**
Function area(l,w):
 return l*w
Function perimeter(l,w):
 return 2*(l+w)
Create package threeDgraphics with modules cuboid and sphere
**cuboid.py**
Function area(l,w,h):
 return ((2*(l*w))+(2*(l*h))+(2*(w*h)))
Function perimeter(l,b,h):
 return 4*(l+b+h)
**sphere.py**
Function area(r):
 return 4*3.14*r*r
Function perimeter(r):
 return (2*3.14*r)

**Source Code :**

```python
from rectangle import area as rectArea
from rectangle import perimeter as rectPerimeter
from circle import perimeter as circlePerimeter
from circle import area as circleArea
from ThreeD.cuboid import *
from ThreeD.sphere import *
length=int(input("Enter the length of rectangle:"))
breadth=int(input("Enter the breadth of rectangle:"))
radius=int(input("Enter radius of circle:"))
l=int(input("Enter the length of cuboid:"))
w=int(input("Enter the width of cuboid:"))
h=int(input("Enter the height of cuboid:"))
r=int(input("Enter radius of sphere:"))
a1=rectArea(length,breadth)
a2=circleArea(radius)
p1=rectPerimeter(length,breadth)
p2=circlePerimeter(radius)
sa1=surface_area(l,w,h)
v1=volu_me(l,w,h)
sa2=surfaceArea(r)
v2=volume(r)
print("Area of rectangle",a1)
print("Perimeter of rectangle:",p1)
print("Area of circle",a2)
print("Perimeter of circle:",p2)
print("Surface Area of cuboid",sa1)
print("volume of cuboid",v1)
print("Surface Area of sphere",sa2)
print("volume of sphere",v2)
```

**rectangle.py**

```python
def area(length, width):
    return length * width

def perimeter(length,width):
    return 2 * (length + width)
```

**circle.py**

```python
def area(radius):
    return 3.14 * radius ** 2

def perimeter(radius):
    return 2 * 3.14 * radius
```
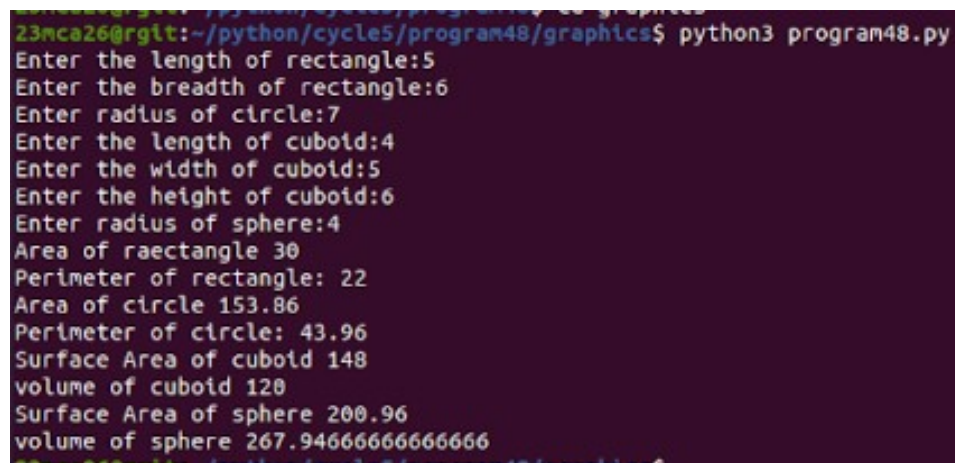
**cuboid.py**

```python
def surface_area(length, width, height):
    return 2 * ((length * width) +( width * height) +( height * length))

def volu_me(length, width, height):
    return length * width * height
```

**sphere.py**

```python
def surfaceArea(radius):
    return 4 *3.14 * radius ** 2

def volume(radius):
    return (4 / 3) *3.14 * radius ** 3
```

**Output:**



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

**Experiment No : 49**

**Date : 8/12/2023**

**Aim:**

Define a class to represent a bank account. Include the following details like name of the depositor, account number, type of account, balance amount in the account. Write methods to assign initial values, to depositan amount , withdraw an amount after checking the balance, to display details such as name, account number, account type and balance.

**Pseudocode :**

create a class to represent bank account with instance variable
name, account no, type and balance
Create method deposit()
Update balance=balance+amount
Create method withdrawal ()
If(balance>=amount)
Update balance=balance-amount

Create method display ()
Print name, account number, type and balance
Create object obj of class account
Call the methods
Obj.deposit(amount)
Obj.withdraw(amount)
Obj.display()

**Source Code:**

```
class BankAccount:
    def __init__(self, name, account_number, account_type, initial_balance=0):
        self.name = name
        self.account_number = account_number
        self.account_type = account_type
        self.balance = initial_balance

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"Deposit of ${amount} successful.")
        else:
            print("Deposit amount should be greater than zero.")

    def withdraw(self, amount):
        if amount > 0:
```

```python
        if self.balance >= amount:
            self.balance -= amount
            print(f"Withdrawal of ${amount} successful.")
        else:
            print("Insufficient funds.")
    else:
        print("Withdrawal amount should be greater than zero.")

    def display_details(self):
        print(f"Account Details\nName: {self.name}\nAccount Number: {self.account_number}\
nAccount Type: {self.account_type}\nBalance: ${self.balance}")


name = input("Enter depositor's name: ")
account_number = input("Enter account number: ")
account_type = input("Enter account type: ")
initial_balance = float(input("Enter initial balance: "))

account = BankAccount(name, account_number, account_type, initial_balance)
account.display_details()

deposit_amount = float(input("Enter amount to deposit: "))
account.deposit(deposit_amount)

withdraw_amount = float(input("Enter amount to withdraw: "))
account.withdraw(withdraw_amount)

account.display_details()
```

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/Desktop/qqqq/cycle 6$ python3 6.1.py
Enter depositor's name: Alen Roy
Enter account number: 231425246
Enter account type: Current
Enter initial balance: 10000
Account Details
Name: Alen Roy
Account Number: 231425246
Account Type: Current
Balance: $10000.0
Enter amount to deposit: 5000
Deposit of $5000.0 successful.
Enter amount to withdraw: 400
Withdrawal of $400.0 successful.
Account Details
Name: Alen Roy
Account Number: 231425246
Account Type: Current
Balance: $14600.0
```

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

**Experiment No : 50**

**Date : 8/12/2023**

**Aim:**
Create a class Publisher with attributes publisher id and publisher name. Derive class Book from Publisher with attributes title and author.Derive class Python from Book with attributes price and no_of_pages.Write a program that displays information about a Python book. Use base class constructor invocation and method overriding.

**Pseudocode :**

Create a class publisher with instance variable
Publisher_id and Publisher_name
Create method display()
Print publisher_id and publisher_name
Create class book and inherit class publisher with instance variable
Title and Author
Create method display()
Print title and author
Create a class python by inheriting class book with instance
Variable, Price and No of pages
Create method display()
Print price and no of pages
Create object obj of class python
Obj.display()

**Source Code:**

```python
class Publisher:
    def __init__(self, publisher_id, publisher_name):
        self.publisher_id = publisher_id
        self.publisher_name = publisher_name

class Book(Publisher):
    def __init__(self, publisher_id, publisher_name, title, author):
        super().__init__(publisher_id, publisher_name)
        self.title = title
        self.author = author

    def display_info(self):
        print(f"Title: {self.title}\nAuthor: {self.author}\nPublisher ID: {self.publisher_id}\nPublisher
Name: {self.publisher_name}")

class Python(Book):
    def __init__(self, publisher_id, publisher_name, title, author, price, no_of_pages):
        super().__init__(publisher_id, publisher_name, title, author)
        self.price = price
```

```python
        self.no_of_pages = no_of_pages

    def display_info(self):
        super().display_info()
        print(f"Price: ${self.price}\nNumber of Pages: {self.no_of_pages}")


publisher_id = input("Enter Publisher ID: ")
publisher_name = input("Enter Publisher Name: ")
title = input("Enter Title of the book: ")
author = input("Enter Author of the book: ")
price = float(input("Enter Price of the book: "))
no_of_pages = int(input("Enter Number of Pages: "))

python_book = Python(publisher_id, publisher_name, title, author, price, no_of_pages)
print("\nInformation about the Python book:")
python_book.display_info()
```
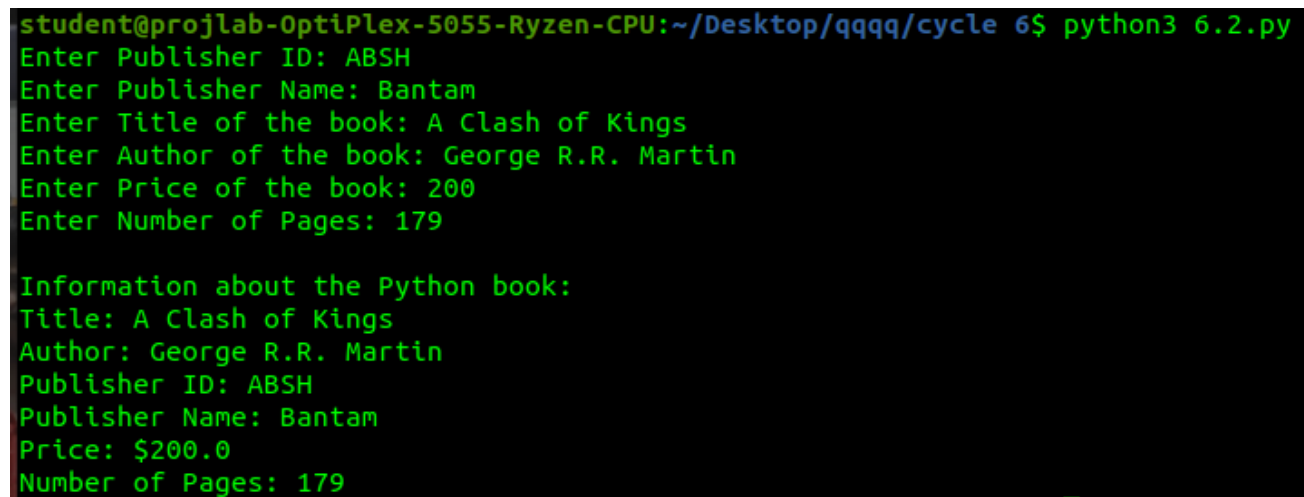**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/Desktop/qqqq/cycle 6$ python3 6.2.py
Enter Publisher ID: ABSH
Enter Publisher Name: Bantam
Enter Title of the book: A Clash of Kings
Enter Author of the book: George R.R. Martin
Enter Price of the book: 200
Enter Number of Pages: 179

Information about the Python book:
Title: A Clash of Kings
Author: George R.R. Martin
Publisher ID: ABSH
Publisher Name: Bantam
Price: $200.0
Number of Pages: 179
```

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

65

**Experiment No : 51**

**Date : 15/12/2023**

**Aim:**

Write a program that has an abstract class Polygon. Derive two classes Rectangle and Triangle from Polygon and write methods to get the details of their dimensions and hence calculate the area.

**Pseudocode :**

Create an abstract class polygon
Create abstract method area()
Pass

Create clays Triangle by inheriting polygon with instance
Variable hand b.
Create method area ()
Print h &b

Create class Rectangle by inheriting polygon with instance variable
Create method area ()
Print word

Read choice.
If choice equals 1,
Read base and height of triangle
Create object obj of Triangle
Obj. Area ()
Else if choice equals 2
Read length and width of rectangle
create object obj of Rectangle obj. Area ()

**Source Code:**

```
from abc import ABC, abstractmethod

class Polygon(ABC):
    def __init__(self, num_sides):
        self.num_sides = num_sides

    @abstractmethod
    def get_dimensions(self):
        pass

    @abstractmethod
    def calculate_area(self):
        pass
```

```python
class Rectangle(Polygon):
    def __init__(self):
        super().__init__(4)
        self.length = 0
        self.breadth = 0

    def get_dimensions(self):
        self.length = float(input("Enter length of the rectangle: "))
        self.breadth = float(input("Enter breadth of the rectangle: "))

    def calculate_area(self):
        return self.length * self.breadth

class Triangle(Polygon):
    def __init__(self):
        super().__init__(3)
        self.base = 0
        self.height = 0

    def get_dimensions(self):
        self.base = float(input("Enter base length of the triangle: "))
        self.height = float(input("Enter height of the triangle: "))

    def calculate_area(self):
        return 0.5 * self.base * self.height

rectangle = Rectangle()
triangle = Triangle()

rectangle.get_dimensions()
triangle.get_dimensions()
area_rectangle = rectangle.calculate_area()
area_triangle = triangle.calculate_area()

print(f"Area of the rectangle: {area_rectangle}")
print(f"Area of the triangle: {area_triangle}")
```
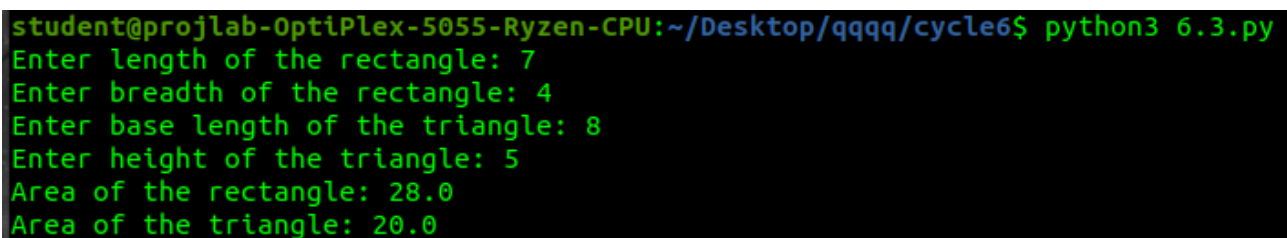
**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/Desktop/qqqq/cycle6$ python3 6.3.py
Enter length of the rectangle: 7
Enter breadth of the rectangle: 4
Enter base length of the triangle: 8
Enter height of the triangle: 5
Area of the rectangle: 28.0
Area of the triangle: 20.0
```

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

**Experiment No : 52**

**Date : 15/12/2023**

**Aim:**

Create a Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area.

**Pseudocode :**

Create class named Rectangle with instance variables
breadth and length.
Create method area ()
return length * breadth
Create method Perimeter()
return 2* length breadth
Create method __gt__ (Self, other)
if (self.area() > other.area()):
return True
else
return False
Read length and breadth for first rectangle
Create object obj1 of rectangle
Read length and breadth for second rectangle
Create object obj2 of rectangle
If (obj1> obj2 )
Print obj1, is greater
Else
Print obj2 is greater.

**Source Code:**

```
class Rectangle:
    def __init__(self, length, breadth):
        self.length = length
        self.breadth = breadth

    def area(self):
        return self.length * self.breadth

    def perimeter(self):
        return 2 * (self.length + self.breadth)


length1 = float(input("Enter length of first rectangle: "))
breadth1 = float(input("Enter breadth of first rectangle: "))
```

```
length2 = float(input("Enter length of second rectangle: "))
breadth2 = float(input("Enter breadth of second rectangle: "))

rectangle1 = Rectangle(length1, breadth1)
rectangle2 = Rectangle(length2, breadth2)


area1 = rectangle1.area()
area2 = rectangle2.area()

if area1 > area2:
    print("Area of first rectangle is greater.")
elif area2 > area1:
    print("Area of second rectangle is greater.")
else:
    print("Both rectangles have the same area.")

print(f"\nArea of first rectangle: {area1}")
print(f"Perimeter of first rectangle: {rectangle1.perimeter()}")

print(f"\nArea of second rectangle: {area2}")
print(f"Perimeter of second rectangle: {rectangle2.perimeter()}")
```
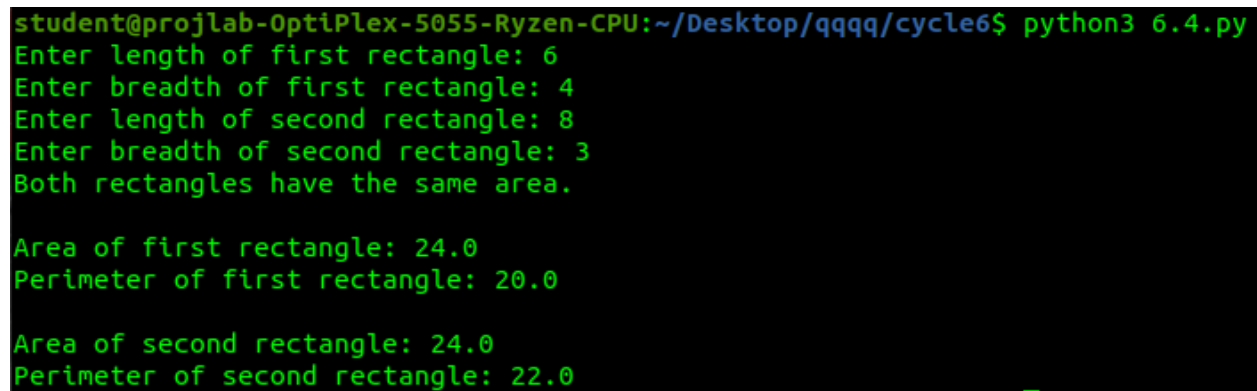
**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/Desktop/qqqq/cycle6$ python3 6.4.py
Enter length of first rectangle: 6
Enter breadth of first rectangle: 4
Enter length of second rectangle: 8
Enter breadth of second rectangle: 3
Both rectangles have the same area.

Area of first rectangle: 24.0
Perimeter of first rectangle: 20.0

Area of second rectangle: 24.0
Perimeter of second rectangle: 22.0
```

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

69

**Experiment No : 53**

**Date : 15/12/2023**

**Aim:**

Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 times.

**Pseudocode :**

Create class named Time with instance variables
Hour, Minute and Seconds
Create method __add__(self,other)
Time1=self.hour+self.minute+self.seconds
Return time1+(other.hour+other.minute+other.seconds)
Read first time as hour minute and second
Create object of Time t1
Read second time as hour minute and second
Create object of Time t2
New_time = t1+t2
print New_time

**Source Code:**

```
class Time:
    def __init__(self):
        self.__hour = 0
        self.__minute = 0
        self.__second = 0

    def set_time(self, hour, minute, second):
        self.__hour = hour
        self.__minute = minute
        self.__second = second

    def __add__(self, other):
        total_seconds_self = self.__hour * 3600 + self.__minute * 60 + self.__second
        total_seconds_other = other.__hour * 3600 + other.__minute * 60 + other.__second

        total_seconds_sum = total_seconds_self + total_seconds_other

        hours = total_seconds_sum // 3600
        minutes = (total_seconds_sum % 3600) // 60
        seconds = total_seconds_sum % 60

        result = Time()
        result.set_time(hours, minutes, seconds)
```

```python
        return result

    def display_time(self):
        print(f"Time: {self.__hour}:{self.__minute}:{self.__second}")


hour1 = int(input("Enter hours for time 1: "))
minute1 = int(input("Enter minutes for time 1: "))
second1 = int(input("Enter seconds for time 1: "))

hour2 = int(input("Enter hours for time 2: "))
minute2 = int(input("Enter minutes for time 2: "))
second2 = int(input("Enter seconds for time 2: "))

time1 = Time()
time1.set_time(hour1, minute1, second1)

time2 = Time()
time2.set_time(hour2, minute2, second2)

result_time = time1 + time2

print("\nTime 1:")
time1.display_time()

print("\nTime 2:")
time2.display_time()

print("\nSum of Time 1 and Time 2:")
result_time.display_time()
```
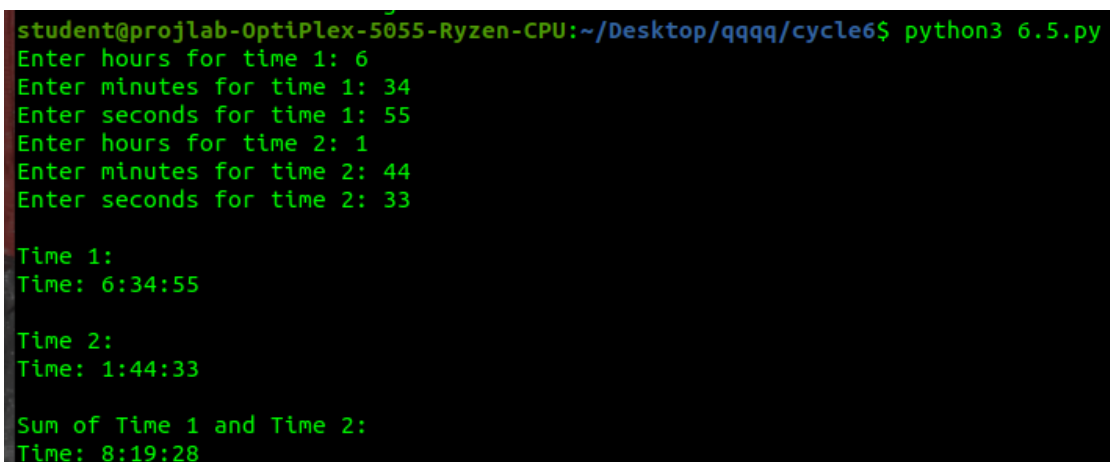
**Output:**



```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/Desktop/qqqq/cycle6$ python3 6.5.py
Enter hours for time 1: 6
Enter minutes for time 1: 34
Enter seconds for time 1: 55
Enter hours for time 2: 1
Enter minutes for time 2: 44
Enter seconds for time 2: 33

Time 1:
Time: 6:34:55

Time 2:
Time: 1:44:33

Sum of Time 1 and Time 2:
Time: 8:19:28
```

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

**Experiment No : 54**

**Date : 15/12/2023**

**Aim:**

Write a program that validates name and age as entered by the user to determine whether the person can cast a vote or not.

**Pseudocode :**

Create class VoteError by inheriting Exception
Pass
Create class InvalidError by inheriting Exception
Pass
Read age and Name
If age <= 0 then
Invoke InvalidError
Else if age<18 then
Invoke VoteError
Else
Print Eligible for voting
Except InvalidError
Print "Enter a valid age"
Except VoteError
Print "Not eligible for voting"

**Source Code:**

```
def validate_name(name):
    return name.isalpha() or ' ' in name

def validate_age(age):
    try:
        age = int(age)
        if age >= 18:
            return True
        else:
            return False
    except ValueError:
        return False

name = input("Enter your name: ")
age = input("Enter your age: ")

if validate_name(name) and validate_age(age):
    print(f"Hello, {name}!")
    print("You are eligible to cast your vote.")
```
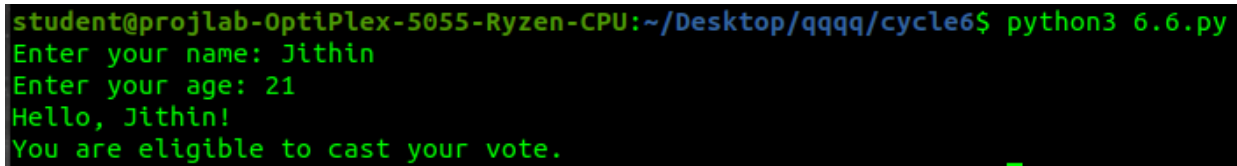
else:
    print("Sorry, you are not eligible to cast your vote.")

**Output:**

```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/Desktop/qqqq/cycle6$ python3 6.6.py
Enter your name: Jithin
Enter your age: 21
Hello, Jithin!
You are eligible to cast your vote.
```

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

**Experiment No : 55**
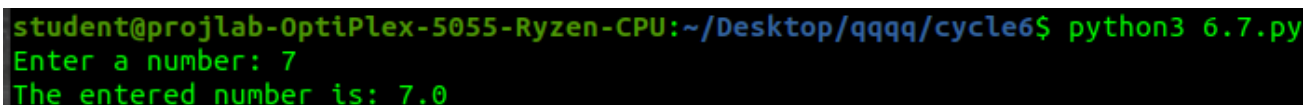
**Date : 4/1/2024**

**Aim:**

Write a program that prompts the user to enter a number. If the number is positive or zero print it, otherwise raise a 'ValueError' Exception.

**Pseudocode :**

Read value
If value < 0 then
Invoke ValueError("This is a negative number")
Else
Print value
Except ValueError as e
Print e

**Source Code:**

```
try:
    number = float(input("Enter a number: "))
    if number >= 0:
        print(f"The entered number is: {number}")
    else:
        raise ValueError("Number is negative")
except ValueError as e:
    print(f"Error: {e}. Please enter a positive number or zero.")
```

**Output:**



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

**Experiment No : 56**

**Date :  4/1/2024**

**Aim:**

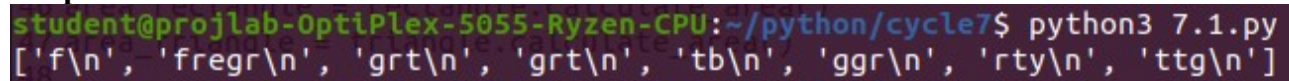Write a Python program to read a file line by line and store it into a list.

**Pseudocode :**

Open file with read access
Lines=[]
For every line in file
Add line to Lines
Print Lines

**Source Code:**

```
file = open("File1.txt","r")
lines=[]
for line in file:
lines.append(line)
print(lines)
file.close()
```

**Output:**



```
student@projlab-OptiPlex-5055-Ryzen-CPU:~/python/cycle7$ python3 7.1.py
['f\n', 'fregr\n', 'grt\n', 'grt\n', 'tb\n', 'ggr\n', 'rty\n', 'ttg\n']
```

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

**Experiment No : 57**

**Date :  4/1/2024**

**Aim:**

Python program to copy odd lines of one file to another .

**Pseudocode :**

Open file1 with read access
Open file2 with write access
Read all lines of file1 and store to Lines
For i from 0 to length of Lines
If i%2!=0
Write Lines[i] to file2

**Source Code:**

```
file = open("File1.txt","r")
file2 = open("File2.txt","w")
line = file.readlines()
for i in range(len(line)):
if(i%2!=0):
file2.write(line[i])
file.close()
file2.close()
print (" successfully copied")
```

**Output:**



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