In [ ]:
```
#Types of Variables in Functional Programming
Python supports two types of variable
1.Global
2.Local
```

In [ ]:
```
#Global variables : are those variables that are declared oootside the function
(above the function declaration)
```

In [20]:
```
#Examples of Each Type of Variable
#Global Variable

def f1():
    print(a)
def f2():
    print(b)
a=10
b=20
print(a)
f1()
f2()
```

```
10
10
20
```

In [ ]:
```
#Local variable -->variables that we declared inside the function that
variable are known as local variable
```

In [24]:
```
def add():
    abc=10
    b=20
    print(abc)
    return abc+b
x=10
y=20
print(add(x,y))
```

```
10
30
```

In [ ]:
```
Scope of a Variable:
1.Global Variable --> global variable can be accessed at any place of the program
(functions , or outside the function)
2.Local variable --> Within the function. You cannot acces it outside the function
```

In [26]:
```
a=20
def f1():
    b=10
    print(a)
f1()
#Note: If a variable is present inside the function then the first priority is given
to that variable only if in that function element is not present then global
variable will be considered
```

```
20
```

In [27]:
```
a=10
def f1():
    a=777
    print(a)
def f2():
    print(a)


f1()
f2()
```

```
777
10
```

In [ ]:
```
#Global Keyword
Gloabal keyword is used for two purposes:
    1.To declare global variable inside the function
    2.to make the global varibalen available to the function
```

In [36]:
```
#Note: if local and global variable names are same then if we want to access global variables
#inside the function then you can access with the help of globals functions
b=10
a=20
def f1():
    a=777
    print(a)
    print(b)
f1()
```

```
777
10
```

In [40]:
```
sum1="Hello"
def sums():
    sum1="world"
    print(sum1)
    print(globals()["sum1"])
sums()
```

```
world
Hello
```

In [ ]:

In [31]:
```
a=10
def f1():
    print(a)
def f2():
    global a
    a=777
    print(a)   #777


f2()
f1()
```

```
777
777
```

In [ ]:
```
#Practice Problems Based on Functions
```

In [42]:
```
#Factorial Problem
def fact(num):
    fact=1
    for i in range(1,num+1):
        fact=fact*i
    return fact
num=int(input("Enter the number"))
ans=fact(num)
print(ans)
```

```
Enter the number90
1485715964481761497309552273362082573788855699612846887669422168637049853930940658765459921313708840596456172344699781120000000000000000000
```

In [50]:
```
#Prime Number
def prime(number):
    if number<0:
        return "Invalid Input"
    elif number==1 or number==0:
        return "Neither prime nor composite"
    else:
        for i in range(2,number):
            if number%i==0:
                return "Number is not Prime"
    return "Number is Prime"
number=int(input("Enter the number"))
ans=prime(number)
print(ans)
```

```
Enter the number12
Number is not Prime
```

In [ ]:
```
#Armstrong Number --> 153-->sum of cube of numbers == original number
```

In [53]:
```
#Armstrong and Strong
def Armstrong(n):
    sum=0
    number=n
    while n!=0:
        rem=n%10  #-->
        sum=sum+(rem*rem*rem)
        n=n//10
    if sum==number:
        return "Armstrong Number"
    else:
        return "Not Armstrong Number"

number=int(input("Enter the number"))
ans=Armstrong(number)
print(ans)
```

```
Enter the number370
Armstrong Number
```

In [56]:
```
#Strong Number --> sum of factorial of each digit == Original Number
#example 145
#Armstrong and Strong
def fact(num):
    fact=1
    for i in range(1,num+1):
        fact=fact*i
    return fact
def Strong(n):
    sum=0
    number=n
    while n!=0:
        rem=n%10  #-->
        sum=sum+fact(rem)
        n=n//10      --> 153  -->15
    if sum==number:
        return "Strong Number"
    else:
        return "Not Strong Number"

number=int(input("Enter the number"))
ans=Strong(number)
print(ans)
```

```
Enter the number371
Not Strong Number
```

In [ ]:
```
def Strong(n):
    sum=0
    number=n
    while n!=0:
        rem=n%10  #-->
        sum=sum+fact(rem)
        n=n//10
    if sum==number:
        return "Strong Number"
    else:
        return "Not Strong Number"

number=int(input("Enter the number"))
ans=Strong(number)
print(ans)
```

In [59]:
```
#153 -->15
#product of digits
def product_of_digits(num):
    product=1
    while num!=0:
        rem=num%10
        product=product*rem
        num=num//10
    return product
number=int(input("Enter the number"))
ans=product_of_digits(number)
print(ans)
```

```
Enter the number153
15
```

In [62]:
```
#Sum of digits
def sum_of_digits(num):    #step3
    sum=0
    while num!=0:
        rem=num%10
        sum=sum+rem
        num=num//10
    return sum
number=int(input("Enter the number"))  #step1
ans=sum_of_digits(number)  #step2

print(ans)      #step5
print("hello")     #step6
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Input In [62], in <cell line: 10>()
      6         num=num//10
      7     return sum
----> 10 number=int(input("Enter the number"))  #step1
     11 ans=sum_of_digits(number)  #step2
     13 print(ans)      #step5

File ~\anaconda3\lib\site-packages\ipykernel\kernelbase.py:1075, in Kernel.raw_input(self, prompt)
   1073 if not self._allow_stdin:
   1074     raise StdinNotImplementedError(
   1075         "raw_input was called, but this frontend does not support input requests."
   1076     )
-> 1077 return self._input_request(
   1078     str(prompt),
   1079     self._parent_ident["shell"],
   1080     self.get_parent("shell"),
   1081     password=False,
   1082 )

File ~\anaconda3\lib\site-packages\ipykernel\kernelbase.py:1120, in Kernel._input_request(self, prompt, ident, parent, password)
   1117             break
   1118 except KeyboardInterrupt:
   1119     # re-raise KeyboardInterrupt, to truncate traceback
-> 1120     raise KeyboardInterrupt("Interrupted by user") from None
   1121 except Exception:
   1122     self.log.warning("Invalid Message:", exc_info=True)

KeyboardInterrupt: Interrupted by user
```

In [64]:
```
def circle(radius):
    x=area(radius)       #3
    y=perimeter(radius)   #5
    print("Area of Circle with given radius",x)  #7
    print("Perimeter of Circle with given radius",y) #8
    return


def area(radius):       #4
    return 3.14*(radius**2)

def perimeter(radius):
    return 2*3.14*radius    #6

radius=int(input("Enter the radius"))       #1
circle(radius)                              #2
```

```
Enter the radius234
Area of Circle with given radius 171933.84
Perimeter of Circle with given radius 1469.52
```

In [ ]:
```
def add(x,y):
    return x+y
```

In [70]:
```
x=10
for i in range(1,x+1):
    for j in range(1,i+1):
        print("*",end=" ")
    print()
```

```
*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
* * * * * * * *
* * * * * * * * *
* * * * * * * * * *
```

In [5]:
```
x=[10,20,30,40,50]
for i in range((len(x))):
    x.pop(i)
    print(x)
```

```
[10, 20, 30, 40, 50]
[10, 20, 30, 40, 50]
[10, 20, 30, 40, 50]
[10, 20, 30, 40, 50]
[10, 20, 30, 40, 50]
```

In [13]:
```
for x in range(6):
    if (x == 3 or x== 6):
        continue
    print(x,end=",")
print("\n")
```

```
0,1,2,4,5,
```

In [ ]:

In [71]:
```
x=[10,20,30,40,50]
for i in range(len(x)):
    x[0],x[-1]=x[-1],x[0]
print(x)
```

```
[50, 20, 30, 40, 10]
```

In [ ]:
```
def ands(*n):
```