Functions

- A function is a set of statements that take inputs, and do some specific computaion and produces output.
- Functions are reusable, self-contained pieces of code that are called with a single command.
- Python provides built-in functions like int(), float(), str(), input(), print() etc. but we can also create our own functions. These are called user defined functions.
- In python a function is defined using the "def" keyword.

Parameters

- A parameter is a variable used to difine a value during a function definition.

Arguments

- An argument is a value passed to a function at the time function calling.

  Syntax:

```
def Function_Name(define parameters):
    statement 1
    statement 2
    statement 3
    .
    .
    statement N
Function_Name(pass arguments)
```

In [1]:

```python
def addition(a, b):
    s = a+b
    print(s)
```

In [2]:

```python
addition(5,8)
```

```
13
```

Types of functions in python

1. Without arguments and without return value
2. Without arguments and with return value
3. With arguments and without return value
4. With arguments and with return value

In [3]:

```python
# 1. Without arguments and without return value
def Add():
    a,b = 5,3
    print(a+b)

Add()
```

```
8
```

In [5]:

```python
# 2. Without arguments and with return value

def Mul():
    x,y = 4,5
    mul = x*y
    return mul

print(Mul())
```

```
20
```

In [7]:

```python
# 3. With arguments and without return values

def Mul(a, b):
    print(a*b)

Mul(4,6)
```

```
24
```

In [6]:

```python
#4. With argument and with return value

def Mul(a, b):
    mul = a*b
    return mul

x = int(input())
y = int(input())
print(Mul(x, y))
```

```
6
7
42
```

In [12]:

```python
n = int(input("Enter a number: ")) # n=3
count = 0
for i in range(1, n+1): #i=1, i=2, i=3
    if n%i == 0:
        count += 1 #count=2

if count == 2:
    print(n,"is prime")
else:
    print(n,"is not prime")
```

```
Enter a number: 3
3 is prime
```

In [14]:

```python
n = int(input("Enter a number: ")) # n=3
count = 0
for i in range(2, n+1): #i=2, i=3
    if n%i == 0:
        count += 1 # count = 1

if count == 1:
    print(n,"is prime")
else:
    print(n,"is not prime")
```

```
Enter a number: 3
3 is prime
```

In [15]:

```python
def PrimeOrNot(n):
    c = 0
    for i in range(1, n+1):
        if n%i == 0:
            c += 1
    if c == 2:
        print(n,'is prime')
    else:
        print(n,'is not prime')

x = int(input())
PrimeOrNot(x)
```

```
7
7 is prime
```

In [17]:

```
n = int(input()) # n=7
for i in range(1,n+1): # i=1, i=2, i=3
    count = 0
    for j in range(1,i+1): # j=1, j=2, j=3
        if i%j == 0:
            count += 1 # c=2
    if count == 2:
        print(i, end=" ")
```

```
7
2 3 5 7
```

In [19]:

```
def primeOrNot(n):
    for i in range(1, n+1):
        c = 0
        for j in range(1, i+1):
            if i%j == 0:
                c += 1
        if c == 2:
            print(i,end=" ")

m = int(input())
primeOrNot(m)
```

```
7
2 3 5 7
```

In [20]:

```
def prime_numbers(n):
    count=0
    for i in range(1,n+1): # i=1, i=2
        for j in range(1,i+1): #j=1,j=2
            if i%j==0:
                count+=1 # cournt=1,2,3
        if count==2:
            print(i)

x=int(input())
prime_numbers(x)
```

```
7
```

Types of arguments

- We have two types of arguments in python. Those are
    1. Actual arguments
    2. Formal arguments

```
1. Actual arguments
    i) Position arguments
    ii) Keyword arguments
    iii) Default arguments
    iv) Variable length arguments
```

In [21]:

```python
# Basic program

def add(a, b): # These are the formal arguments
    c = a+b
    print(c)

add(3,5) # These are the actual arguments
```

```
8
```

In [23]:

```python
# 1. Positional arguments

def Person(name, age):
    print("Person's name: ",name)
    print("Person's age: ",age)

Person('kits',20)
```

```
Person's name:  kits
Person's age:  20
```

In [24]:

```python
def Person(name, age):
    print("Person's name: ",name)
    print("Person's age: ",age)

Person(20, 'kits')
```

```
Person's name:  20
Person's age:  kits
```

In [25]:

```
def Person(name, age):
    print("Person's name: ",name)
    print("Person's age: ",age-1)

Person(20, 'kits')
```

```
Person's name:  20

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-25-0b0ac0c3b644> in <module>
      3     print("Person's age: ",age-1)
      4 
----> 5 Person(20, 'kits')

<ipython-input-25-0b0ac0c3b644> in Person(name, age)
      1 def Person(name, age):
      2     print("Person's name: ",name)
----> 3     print("Person's age: ",age-1)
      4 
      5 Person(20, 'kits')

TypeError: unsupported operand type(s) for -: 'str' and 'int'
```

In [27]:

```
# 2. keyword arguments

def Person(name, age):
    print("Person's name: ",name)
    print("Person's age: ",age-1)

Person(age=20, name='kits')
```

```
Person's name:  kits
Person's age:  19
```

In [28]:

```
# 3. Defalut arguments

def Person(name, age=21):
    print("Person's name: ",name)
    print("Person's age: ",age)

Person('kits')
```

```
Person's name:  kits
Person's age:  21
```

In [29]:

```
def Person(name, age=21):
    print("Person's name: ",name)
    print("Person's age: ",age)

Person('kits', 40)
```

```
Person's name:  kits
Person's age:  40
```

In [30]:

```
# 4. Variable length argument

def Add(a, b):
    c = a+b
    print(c)

Add(1,2,3,4,5)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-30-eaf503e7c7f3> in <module>
      5     print(c)
      6 
----> 7 Add(1,2,3,4,5)

TypeError: Add() takes 2 positional arguments but 5 were given
```

In [31]:

```
def Add(a, *b):
    print('A=',a)
    print("B =",b)

Add(1,2,3,4,5)
```

```
A= 1
B = (2, 3, 4, 5)
```

In [32]:

```
def Add(a, *b):
    s = a
    for i in b:
        s += i
    print(s)

Add(1,2,3,4,5,6)
```

```
21
```

In [ ]:

```

```