```
def MyFunction():
  print("Hello, I m a function")
```

```
print("Main starts")
print("Function call")
MyFunction()
print("Main ends")
```

```
Main starts
Function call
Hello, I m a function
Main ends
```

first line after function def is document string

```
def max(no1, no2):
  ''' takes 2 inputs and prints maximum of 2'''
  if no1>no2:
    print(no1, "is max")
  else:
    print(no2, "is max")
```

```
max()
```

```
max(10, 20)
```

```
20 is max
```

```
max(4.5, 9.9)
```

```
9.9 is max
```

```
max("amar", "zamar")
```

```
zamar is max
```

```
max(True, False)
```

```
True is max
```

```
def max(no1, no2):
  ''' takes 2 inputs and prints maximum of 2'''
  if no1>no2:
    return no1
  else:
    return no2
```

```
print("Maximun is:", max(int(input("Enter a number:")), int(input("Enter another number:"))))
```

```
Enter a number:12
Enter another number:44
Maximun is: 44
```

```
no1=int(input("Enter a number:"))
no2=int(input("Enter another number:"))
ans=max(no1, no2)
print("Maximun is", ans)
```

```
Enter a number:12
Enter another number:44
Maximun is 44
```

```
# use above code to print max of 4 numbers
no1=int(input("Enter 1st number: "))
no2=int(input("Enter 2nd number: "))
no3=int(input("Enter 3rd number: "))
no4=int(input("Enter 4th number: "))
print("maximum is", max(max(no1, no2), max(no3, no4)) )
```

```
Enter 1st number: 12
Enter 2nd number: 4
Enter 3rd number: 55
Enter 4th number: 1
maximum is 55
```

```
def doubler(no):
  print("in function before changes no:", no)
  no*=2
  print("in function after changes no:", no)
```

```
print("in main")
no=10
print("in main before changes no:", no)
doubler(no)
print("in main after changes no:", no)
```

```
in main
in main before changes no: 10
in function before changes no: 10
in function after changes no: 20
in main after changes no: 10
```

```
def doubler(no):
  print("in function before changes no:", no)
  no[0]*=2
  print("in function after changes no:", no)
```

```
print("in main")
no=[10]
print("in main before changes no:", no)
doubler(no)
print("in main after changes no:", no)
```

```
in main
in main before changes no: [10]
in function before changes no: [10]
in function after changes no: [20]
in main after changes no: [20]
```

```
def fact(no):
  if no<=1:
    return 1
  else:
    return no*fact(no-1)
```

```
fact(4)
```

```
24
```

```
# if no1%no2==0 then no2 is gcd, else let no1=no2 and no2 be no1%no2
def gcd(no1, no2):
  if no1%no2==0:
    return no2
  else:
    return gcd(no2, no1%no2)
```

```
print(gcd(12, 18))
```

```
6
```

```
def Ascending(no1, no2):
  ''' takes 2 inputs and prints maximum of 2'''
  if no1>no2:
    return no2, no1
  else:
    return no1, no2
```

```
ans=Ascending(33, 2)
print(type(ans), ans)
```

```
<class 'tuple'> (2, 33)
```

```
def Descending(no1, no2, no3):
  ''' takes 2 inputs and prints maximum of 2'''
  if no1<no2 and no2<no3:
    return no3, no2, no1
  elif no2<no1 and no1<no3:
    return no3, no1, no2
  elif no1<no3 and no3<no2:
    return no2, no3, no1
  elif no3<no1 and no1<no2:
    return no2, no1, no3
  elif no3<no2 and no2<no1:
    return no1, no2, no3
```

```
  else :
    return no1, no3, no2
```

```
print(Descending(23, 43, 21))
```

```
(43, 23, 21)
```

```
def Descending(no1, no2, no3):
  ''' takes 2 inputs and prints maximum of 2'''
  if no1>no2 and no1>no3:
    if no2>no3:
      return no1, no2, no3
    else:
      return no1, no3, no2
  elif no2>no1 and no2>no3:
    if no1>no3:
      return no2, no1, no3
    else:
      return no2, no3, no1
  elif no3>no1 and no3>no2:
    if no1>no2:
      return no3, no1, no2
    else:
      return no3, no2, no1

```

```
print(Descending(23, 43, 23))
```

```
def intro(name, native):
  print("Hi I am", name, "and from", native)
```

```
intro("amar", "mumbai")
```

```
Hi I am amar and from mumbai
```

```
intro("gujarat", "jayesh")
```

```
Hi I am gujarat and from jayesh
```

```
intro(native="gujarat", name="jayesh") # key-value parameter
```

```
Hi I am jayesh and from gujarat
```

```
def intro(name="unnamed", native="unknown"):
  print("Hi I am", name, "and from", native)
```

```
intro()
```

```
Hi I am unnamed and from unknown
```

```
intro("nitin")
```

```
Hi I am nitin and from unknown
```

```
intro(native="kokan")
```

```
Hi I am unnamed and from kokan
```

```
def Hobbies(*hlist):
  print(type(hlist))
  print("Total Hobbies:", len(hlist))
  for i in hlist:
    print("->", i)
```

```
Hobbies("code", "travel", "sketch", "read")
```

```
<class 'tuple'>
Total Hobbies: 4
-> code
-> travel
-> sketch
-> read
```

```
Hobbies("Eat", "Sleep")
```

```
<class 'tuple'>
Total Hobbies: 2
-> Eat
-> Sleep
```

```
# accept n parameters from user and return total
def totaling(*ele):
  summ=0
  for i in range(len(ele)):
    summ=summ+int(ele[i])
  print(summ)
totaling(1, 2, 3, 4, 5, 6)
```

```
21
```

```
# calculate x^n using power function where default value of x & n is 1 if not given
def pwr(x=1, n=1):
  return x**n
print(pwr(2,3))
print(pwr())
```

```
8
1
```

iterator

```
rollno=[1, 2, 3, 4, 5, 6, 7] # iterator
i=iter(rollno)
```

```
next(i)
```

```
1
```

generator

```
def fd(amount, roi): # generator
  y=1
  print("at end of a year")
  yield(amount*(roi/100)*y)
  y+=1
  print("at end of 2nd year")
  yield(amount*(roi/100)*y)
  y+=1
  print("at end of 3rd year")
  yield(amount*(roi/100)*y)
  y+=1
```

```
i=fd(1000, 7)
```

```
next(i)
```

```
at end of a year
70.0
```

```
def intro(name, native):
  print("hi I am", name, "and from", native)
```

```
hi=intro
```

```
hi("amar", "mumbai")
```

```
hi I am amar and from mumbai
```

decorator

Double-click (or enter) to edit

```
def eng(name):
  print("Hi my dear friend", name, "how are you")
def hindi(name):
  print("नमस्ते मेरे प्रिय मित्र", name, "आप कैसे हैं")
```

```
def marathi(name):
  print("नमस्कार माझ्या प्रिय मित्रा", name, "तू कसा आहेस")
```

```
def welcome(name, lang): # decorator
  lang(name)
```

```
welcome("AMAR", eng)
```

```
Hi my dear friend AMAR how are you
```

```
welcome("AMAR", hindi)
```

```
नमस्ते मेरे प्रिय मित्र AMAR आप कैसे हैं
```

```
welcome("AMAR", marathi)
```

```
नमस्कार माझ्या प्रिय मित्रा AMAR तू कसा आहेस
```

Lambda

```
# basic syntax
# lambda parameter/s  : expression
doubler=lambda a:a*2
```

```
doubler(100)
```

```
200
```

```
# lambda function to get a & b, and then return a + b
addtwo = lambda a,b:a+b
```

```
addtwo(10, 20)
```

```
30
```

```
x=10
def test():
  x=100
  print(x)
```

```
test()
```

```
100
```

variable scope

```
# global variable accessed by all
x=10
def test():
  print("inside test x: ", x)
def test2():
  x=100
  print("inside test2 ---> x:", x)

print("outside: x:", x)
test()
test2()
```

```
outside: x: 10
inside test x:  10
inside test2 ---> x: 100
```

```
# global variable accessed by all
x=10
def test():
  print("inside test x: ", x)
def test2():
  x=100
  print("inside test2 ---> x:", x)
def adder():
  global x
  x+=1
  print("inside adder +++++++x:", x)
print("outside: x:", x)
test()
test2()
adder()
```

```
outside: x: 10
inside test x:  10
inside test2 ---> x: 100
inside adder +++++++x: 11
```

```
str1="abc"
def strmod(str2):
  str2=str2+"modded"
  return str2
print(str1) #
print(strmod(str1)) # prints temp modified variable
print(str1)
```

```
abc
abcmodded
abc
```

OOP- Object Oriented Programming

```
class Human:
  def birth(self):
```

```
    self.gender=input("Enter gender: ")
  def naming(self):
    self.name=input("Enter name: ")
  def intro(self):
    print("Hi I am a", self.gender, "called", self.name)
```

```
h=Human()
```

```
h.birth()
```

```
Enter gender: male
```

```
h.gender
```

```
'male'
```

```
h.naming()
```

```
Enter name: amar
Enter name: amar
```

```
h.intro()
```

```
Hi I am a male called amar
```

```
class Human:
  def birth(self):
    print("Reference:", id(self))
    self.gender=input("Enter gender: ")
  def naming(self):
    self.name=input("Enter name: ")
  def intro(self):
    print("Hi I am a", self.gender, "called", self.name)
```

```
h1=Human()
h2=Human()
```

```
h1.birth()
h2.birth()
```

```
Reference: 140629385384912
Enter gender: male
Reference: 140629385387696
Enter gender: female
```

```
h2.naming()
```

```
Enter name: wonder woman
```

```
h1.naming()
```

```
Enter name: superman
```

```
h1.intro()
```

```
Hi I am a male called superman
```

```
h2.intro()
```

```
Hi I am a female called wonder woman
```

```
class Human:
  def __init__(self): # constructor
    print("object created:", id(self))
    self.gender=input("Enter gender: ")
    self.name=input("Enter name: ")
  def intro(self):
    print("Hi I am a", self.gender, "called", self.name)
```

```
h=Human()
```

```
object created: 140630255061504
Enter gender: male
Enter name: man
```

```
h.intro()
```

```
Hi I am a male called man
```

```
print(h)
```

```
<__main__.Human object at 0x7fe70865ca00>
```

```
class Human:
  def __init__(self): # constructor method
    print("object created:", id(self))
    self.gender=input("Enter gender: ")
    self.name=input("Enter name: ")
  def __str__(self): # printer method, returns string
    return "Hi I am a "+self.gender+" called "+self.name
```

```
h=Human()
```

```
object created: 140629385434496
Enter gender: female
Enter name: woman
```

```
print(h)
```

```
Hi I am a female called woman
```

```
1
```