Examples

Example - 1 29-07-2020

In [ ]:

```
# basic print statement
print("Hello World!")
```

In [ ]:

```
# I am single line comment
# How to write comments in Python
# suppose there are multiple lines of text
# in python and you want to comment
# all lines in one go
# do we have any short cut ??
```

In [ ]:

```
# Indent is important for us
print("I am from comment section")
print("I am also from comment section")
```

In [ ]:

```
#Multi line comments, is there any way out !
'''I am from multi line section
Me too
Me too'''
```

Example - 2 30-07-2020

In [ ]:

```
# how to use variable in python, lets talk about addition of two nums
'''
int var1, var2, summ;
var1=100;
var2=200;
summ=var1 + var2;
printf("sum is %d", summ)
'''

var1 = 100
var2 = 200
summ = 0
summ= var1 + var2
#print("Sum is" + summ) this will generate error because sum is of type string and summ
# op1
print("Sum is", summ)
#op2
print("Sum is", str(summ))
```

In [ ]:

```python
# can we check what is the type of variable
var1=100
var2="200"
var3=100.5
var4=True
var5=["a","b","c"]
var6=("a","b","c")
var7={"a","b","c"}
var8={"name": "python",
      "spec": "PL"}

print(type(var1))
print(type(var2))
print(type(var3))
print(type(var4))
print(type(var5))
print(type(var6))
print(type(var7))
print(type(var8))
```

In [ ]:

```python
# how to take multiple variables at one time
# var1=100
# var2=100
# var3=100

var1=var2=var3=100  # short notation - 1
print(var1) #100
print(var2) #100
print(var3) #100
print("========")
var1=100
var2=200
var3=300

var1, var2, var3= 100, 200, 300 #shortnotation-2
print(var1) #100
print(var2) #200
print(var3) #300
```

Example - 3 03-08-2020

In [ ]:

```
# how to change the type of data or how to set type of data

var1=100
var2="200"
var3=["a","b","c"]
var4=("a","b","c")
var5=10.7
var6=20.7
var7=True
var8=False

print("Default Types")
print(var1, type(var1))
print(var2, type(var2))
print(var3, type(var3))
print(var4, type(var4))
print(var5, type(var5))
print(var6, type(var6))
print(var7, type(var7))
print(var8, type(var8))

var1=str(var1)
var2=int(var2)
var3=tuple(var3)
var4=list(var4)
var5=int(var5)
var6=str(var6)
var7=int(var7)
var8=str(var8)

print("\nTypes After Updation")
print(var1, type(var1))
print(var2, type(var2))
print(var3, type(var3))
print(var4, type(var4))
print(var5, type(var5))
print(var6, type(var6))
print(var7, type(var7))
print(var8, type(var8))
```

In [ ]:

```
# different ways to present the data

var1=1000.0 # traditional float representation
var2=10e2 # variation - 1


print(var1, type(var1))
print(var2, type(var2))
```

In [ ]:

```
# complex numbers
var1=10+4j
var2=10+4j
var3=10

print(var1, type(var1))
print(var2, type(var2))
summ=var1+var2
print(summ)

print(var3, type(var3))

var3=complex(var3) # Accepted !
print(var3, type(var3))

#var1=int(var1) # TypeError: can't convert complex to int

```

Example - 4 04-08-2020

In [ ]:

```
# Strings - Introduction - Printing
var1 = "I am the first String" # printing of string with ""

var2 = 'I am the second String'  # printing of string with ''

var3="""I
        am
        multiline
        string""" # printing of string in multiple line with ""

var4='''I
        am
        multiline
        string''' # printing of string in multiple line with ''

print(var1,type(var1))
print(var2,type(var2))
print(var3,type(var3))
print(var4,type(var4))
```

In [ ]:

```
# Strings - Introduction - Accessing as Arrays
var1 = "I am the first String" #
        #012345678910........

print(var1[0]) # Strings are represented as Arrays  - char I
print(var1[1]) # Strings are represented as Arrays  - char space
print(var1[2]) # Strings are represented as Arrays  - char a
print(var1[3]) # Strings are represented as Arrays  - char m
print(var1[4]) # Strings are represented as Arrays  - char space

print(var1[2:4]) # Strings can be printed or accessed upto specific range also am " Inc

print(var1[2:]) # Strings can be printed or accessed upto specific range also am

print(var1[:4]) # Strings can be printed or accessed upto specific range also am

print(var1[-4:-2]) # Strings can be printed or accessed upto specific range also ri "Ne

print(var1[-6:]) # Strings can be printed or accessed upto specific range also String

print(var1[:-2]) # Strings can be printed or accessed upto specific range also "I am th
```

Example - 5 05-08-2020

In [ ]:

```
# String manipulation methods

varstr = "I am the new string on new day"
print("\nLength of the entered string is:",len(varstr)) # to find the Length of string

varstr="I Am SECOND oNe"
print("\nString in lowercase is: ",varstr.lower()) # to change the case of string from

varstr="I Am third oNe"
print("\nString in uppercase is: ",varstr.upper()) # to change the case of string from

varstr="I am fourth in a row"
chartobechanged="a"
chartobechangedto="z"
print("\nNew String is: ",varstr.replace(chartobechanged,chartobechangedto)) # to repl
#print("\nNew String is: ",varstr.replace("a","z")) # to replace char "a" with "z"

varstr="Here, I come as a 5th, string"
print(varstr)
print("\nSplitted string is: ",varstr.split(",")) # , space, normal char --> used to sp

varstr="    Here, I come as a 6th, string   "
print(varstr, len(varstr))
varstr=varstr.strip() # to remove any space in beginning and at the end of string
print(varstr,len(varstr))

```

Example - 6 06-08-2020

In [ ]:

```
# String manipulation methods - 2

varstr1 = "I am the new string"
varstr2 = "on new day"
finalstring=varstr1 + varstr2
print("\nConcatenated string is:", finalstring) # to concatenate two strings


varstr3 = "I am the new string - See you again"
varstr4="new"
check=varstr4 in varstr3  # to check whether entered word is present in the string
print(check)

varstr3 = "I am the new string - See you again"
varstr4="NEW"  # NEW is not equal to new
check=varstr4 not in varstr3  # to check whether entered word is not present in the str
print(check)
```

In [ ]:

```
# Tuples , Introduction

firsttuple=("a","b","a") # basic tuple
print(firsttuple)

print(firsttuple[1]) # accessing tuple by its positive index ( from left to right)

print(firsttuple[-1]) # accessing tuple by its negative index ( from right to left)

print(firsttuple[0:2]) # accessing tuple by its range


findele="a" # trying to find element b in the Tuple
srch=firsttuple.index(findele) # Search is based on its index
print(srch) # print results - index


findele="da" # trying to find element da in the Tuple which is not present
#srch=firsttuple.index(findele) # Search is based on its index
#print(srch) # print results - value error

findele="a" # trying to find element a in the Tuple which is present at 2 locations
srch=firsttuple.index(findele) # Search is based on its index
print(srch) # print results - index

```

Example - 7 10-08-2020

In [ ]:

```python
## Operators - Arithmetic

#way 1

# v1=input('enter first no.')
# v2=input('enter second no.')
# print(int(v1)+int(v2))

#way 2
a= int(input("Enter First Number "))
b= int(input("Enter Second Number "))

# arithmetic operations
c= a+b
d= a*b
e= b-a
f= a/b
g= a%b
h= a**b
i= a//b

#print("Sum of",a,"and",b,"is",c)

print("Sum of {} and {} is {}".format(a,b,c))
print("Multiplication of {} and {} is {}".format(a,b,d))
print("Subtraction of {} and {} is {}".format(a,b,e))
print("Division of {} and {} is {}".format(a,b,f))
print("Modulus of {} and {} is {}".format(a,b,g))
print("Exponentiation of {} and {} is {}".format(a,b,h))
print("Floor division of {} and {} is {}".format(a,b,i))
```

In [ ]:

```python
# Operators - Assignment

var1=int(input("enter any number"))

#var1=var1+10
var1+=10 #var1=var1 + 10 short notation to perform calculations
print(var1)

var1-=10 #var1=var1 - 10 short notation to perform calculations
print(var1)

var1*=10 #var1=var1 * 10 short notation to perform calculations
print(var1)

var1/=10 #var1=var1 / 10 short notation to perform calculations
print(var1)

var1%=10 #var1=var1 % 10 short notation to perform calculations
print(var1)

var1//=10 #var1=var1 // 10 short notation to perform calculations
print(var1)
```

Example - 8 11-08-2020

In [ ]:

```python
# Operators - Assignment Comparison

a= int(input("Enter First Number: "))
b= int(input("Enter Second Number: "))

# ==, >, <, >=, <=, !=

print("\nStatus of a==b is",a==b) # TRUE or FALSE
print("\nStatus of a>b is",a>b) # TRUE or FALSE
print("\nStatus of a<b is",a<b) # TRUE or FALSE
print("\nStatus of a>=b is",a>=b) # TRUE or FALSE
print("\nStatus of a<=b is",a<=b) # TRUE or FALSE
print("\nStatus of a!=b is",a!=b) # TRUE or FALSE


```

In [ ]:

```python
# Operators - Logical Comparison

a= int(input("Enter value of a: "))
b= int(input("Enter value of b: "))

# and, or, not

print("\nStatus of and is ",a > b and a < 100 ) # TRUE or FALSE

print("\nStatus of or is",a > b or a < 100) # TRUE or FALSE

print("\nStatus of not is",not(a > b and a < 100)) # TRUE or FALSE

```

In [3]:

```
# Operators - Identity specification

var_list_1=["a","b","c"]

var_list_3=["a","b"]

print(var_list_1)
#print(var_list_2) this line will generate error
print(var_list_3)

var_list_2=var_list_1 # var list 2 here is exactly same as list 1 because it is createc

var_list_3=var_list_2 # var list 2 here is exactly same as list 1 because it is createc

# is, is not

print(var_list_1 is var_list_2) #True

print(var_list_2 is not var_list_3) #False # var list 3 is not created from list 1

print(var_list_1)
print(var_list_2)
print(var_list_3)
```

```
['a', 'b', 'c']
['a', 'b']
True
False
['a', 'b', 'c']
['a', 'b', 'c']
['a', 'b', 'c']
```

In [6]:

```
# Operators - Membership specification

var_list_1=["a","b","c"]

var_list_3=["a","b"]

print("a" in var_list_1) # we can check whether particular element exists in the list c

print("ab" in var_list_1)  # we can check whether particular element exists in the list

print("a" not in var_list_1)  # we can check whether particular element does not exist

```

```
True
False
False
```

Example - 9 12-08-2020

In [8]:

```
# Program Flow Control using if, if else

a= int(input("Enter First Number: "))
b= int(input("Enter Second Number: "))

if a > b:
    print("a is greater than b") # we are checking if a > b

elif a==b: # short form to write if else ( else if)
    print("a and b are equal")

else: # else statement must match with indentation of if statement
    print("b is greater than a") # we are not checking anything, if first is not true,


#3 combinations
#1. a=10, b=20
#2. a=20, b=10
#3. a=10, b=10
```

```
Enter First Number: 10
Enter Second Number: 10
a and b are equal
```

In [9]:

```
# Program Flow Control using if, if else - short hand notation - 1

a= int(input("Enter First Number: "))
b= int(input("Enter Second Number: "))

if a > b: print("a is greater than b") # we are checking if a > b
```

```
Enter First Number: 10
Enter Second Number: 3
a is greater than b
```

In [11]:

```
# Program Flow Control using if, if else - short hand notation - 2

a= int(input("Enter First Number: "))
b= int(input("Enter Second Number: "))

print("a is greater than b") if a > b else print("b is greater than a") # we are check
```

```
Enter First Number: 5
Enter Second Number: 10
b is greater than a
```

In [15]:

```
# Program Flow Control using if, if else - short hand notation - 3

a= int(input("Enter First Number: "))
b= int(input("Enter Second Number: "))

print("a is greater than b") if a > b else print("a and b are equal")  if a == b else p
```

```
Enter First Number: 10
Enter Second Number: 10
a and b are equal
```

In [19]:

```
# Program Flow Control using if, if else - more examples ( for all three numbers as sep

a= int(input("Enter First Number: "))
b= int(input("Enter Second Number: "))
c= int(input("Enter Third Number: "))

# some if statement we can add here to check whether input of a,b and c are equal or no

if a >= b and a >= c:
    print("a is greater than b and c") # we are checking if a > b

elif b >= a and b >= c: # short form to write if else ( else if)
    print("b is greater than a and c")

else: # else statement must match with indentation of if statement
    print("c is greater than a and b") # we are not checking

```

```
Enter First Number: 10
Enter Second Number: 20
Enter Third Number: 5
b is greater than a and c
```

In [23]:

```python
#code provided by Cheki
a=int(input("Enter the First Number:"))
b=int(input("Enter the Second Number:"))
c=int(input("Enter the Third Number:"))

if a>b and a>c:
    print("a is greater than b and c")

elif a==b and a==c:
    print("a is equal to b and c")

elif a==b and a>c:
    print("a and b is equal but greater than c")

elif b>c:
    print("b is greater than a and c")

elif b==c:
    print("b and c is equal but greater than a")

else:
    print("c is greater than a and b")
```

```
Enter the First Number:10
Enter the Second Number:9
Enter the Third Number:9
a is greater than b and c
```

Example - 10 13-08-2020

In [5]:

```python
# Program Flow Control using if, if else - cont.. pass

a= int(input("Enter First Number: "))
b= int(input("Enter Second Number: "))
if a > b:
    # how to skip the execution of this section i.e. True case
    pass
else:
    print("b is greater than a")

```

```
Enter First Number: 9
Enter Second Number: 8
```

In [8]:

```python
# Program Flow Control using if, if else - cont.. nested if

a= int(input("Enter a: "))
b= int(input("Enter b: "))

if a > b:
    print("a is greater than b") # we are checking if a > b

    if a > 100: #example of nested if else
        print("yes, a is greater than 100")

    else: # else statement must match with indentation of if statement
        print("no, a is not greater than 100")

else:
    print("b is greater than a")

```

```
Enter a: 100
Enter b: 100
b is greater than a
```

In [9]:

```python
#even or odd by jayant

a = int(input("Enter any number"))
if a % 2 == 0:
    print("Even")
else:
    print("Odd")

```

```
Enter any number8
Even
```

In [10]:

```
#code by bharat

# no. of classes held
# no. of classes attended
# percentage of classess attended
cls_held=int(input('enter classes held'))
cls_attended=int(input('enter classes attended'))
if cls_attended <= cls_held:

    p=int((cls_attended/cls_held)*100)
    print('percentage is:',p)
    if p >=75:
        print('u r eligible to write exames')
    else:
         print('u r not eligible to write exames')

else:
    print('attended classes should be less then classes held')
```

```
enter classes held9
enter classes attended9
percentage is: 100
u r eligible to write exames
```

Example - 11 17-08-2020

In [1]:

```
# Loops - while - 3 condition, increment/decrement, initial value of loop

initial = 1 # initial value
while initial < 10: # termination condition
    print(initial) # printing of results
    initial=initial+1 # increment factor
```

```
1
2
3
4
5
6
7
8
9
```

In [2]:

```
# Loops - while - 3 condition, increment/decrement, initial value of loop

initial = 10 # initial value
while initial > 0: # termination condition
    print(initial) # printing of results
    initial=initial-1 # decrement factor
```

```
10
9
8
7
6
5
4
3
2
1
```

In [3]:

```
# Loops - while - 3 condition, increment/decrement, initial value of loop

initial = 10 # initial value
while initial > 0: # termination condition
    print(initial) # printing of results - indefinite times
```

```
10
10
10
10
10
10
10
10
10
10
10
10
10
10
10
10
10
10
10
```

In [7]:

```
# Loops - while - 3 , break

initial = 1 # initial value
while initial <= 10: # termination condition
    print(initial) # printing of results
    if initial == 5:
        break
    initial=initial+1 # increment factor
```

```
1
2
3
4
5
```

In [10]:

```
# Loops - while - 3 , continue

initial = 0 # initial value
while initial <10: # termination condition
    initial=initial+1 # increment factor
    if initial == 5:
        continue
    print(initial)

```

```
1
2
3
4
6
7
8
9
10
```

In [13]:

```
# Loops - while, else

initial = 1 # initial value
while initial <= 10: # termination condition
    print(initial) # printing of results
    initial=initial+1 # increment factor
else:
    print("Loop execution completed!!")
```

```
1
2
3
4
5
6
7
8
9
10
Loop execution completed!!
```

Example - 12 18-08-2020

In [10]:

```
# Loops - for  - 3 -> condition, increment/decrement, initial value of Loop - v1

for initial in range(10):
    print("initial") # printing of results
```

```
initial
initial
initial
initial
initial
initial
initial
initial
initial
initial
```

In [2]:

```
# Loops - for  - 3 -> condition, increment/decrement, initial value of Loop - v2

for initial in range(1,10):
    print(initial) # printing of results
```

```
1
2
3
4
5
6
7
8
9
```

In [8]:

```
# Loops - for  - 3 -> condition, increment/decrement, initial value of Loop - v3

for initial in range(1,10,2):
    print(initial) # printing of results
```

```
1
3
5
7
9
```

In [4]:

```
# Loops - for  - else
for initial in range(1,10,2):
    print(initial) # printing of results
else:
    print("execution terminates!!")
```

```
1
3
5
7
9
execution terminates!!
```

In [11]:

```
# Loops - for  - pass

for initial in range(1,10,2):
    if initial == 7:
        pass # skip the printing of 7
    else:
        print(initial)
```

```
1
3
5
9
```

In [15]:

```
# Loops - for  - break

for initial in range(1,10,2):
    if initial == 7:
        break # stops the execution of Loop
    else:
        print(initial)
```

```
1
3
5
```

In [13]:

```
# Loops - for  - collections - ex 1

country = ["IND", "USA", "PAK"]

for initial in country:
    print(initial)
```

```
IND
USA
PAK
```

In [16]:

```
# Loops - for  - strings - ex 2

for initial in "country":
    print(initial)
```

```
c
o
u
n
t
r
y
```

In [18]:

```
# Loops - for  - using multiple for loops

states = ["PB", "KL"]
caps= ["CHD", "TRV"]

for i in states:
    for j in caps:
        print(i,j)
```

```
PB CHD
PB TRV
KL CHD
KL TRV
```

Example - 13 19-08-2020

In [23]:

```
# Revisiting Strings - method - format, v1
marksObt=98
marksMax=100
var1 = "Great, I have scored {} marks out of {}!"

print(var1.format(marksObt,marksMax))
```

```
Great, I have scored 98 marks out of 100!
```

In [24]:

```
# Revisiting Strings - method - format, v2

var1 = "Great, I have scored {} marks out of {}!"

print(var1.format(98,100))
```

```
Great, I have scored 98 marks out of 100!
```

In [29]:

```
# Revisiting Strings - method - format, v3

marksObt=99
marksMax=100
percen=99.0
var1 = "Great, I have scored {1} marks out of {0} with {2}%"

print(var1.format(marksMax,marksObt,percen))
```

```
Great, I have scored 99 marks out of 100 with 99.0%
```

In [48]:

```
# Revisiting Strings - Escape Sequences \ backslash \n \t \\ \" \'

marksObt=99
marksMax=100
percen=99.0
var1 = '"Great Effort", You have scored {1} marks out of {0} with {2}%\n' # with single

var2 = "\"Great \nEffort\", \tYou have scored {1} marks out of {0} with {2}%" # with es

print(var1.format(marksMax,marksObt,percen))
print(var2.format(marksMax,marksObt,percen))
```

```
"Great Effort", You have scored 99 marks out of 100 with 99.0%

"Great
Effort",        You have scored 99 marks out of 100 with 99.0%
```