Creating Variables in Python

```
x=5
print(x)
print(type(x))

str="hello python"
print(str)
print(type(str))

y=4.5
print(y)
print(type(y))
```

Special Data types in Python

Tuple

```
#tuple : takes duplicates, cannot assign a value
cities=('madrid','paris','london','new york','madrid')

#fetch each items based on index
print(cities[0])
print(cities[2])

#print all tuple values
print(cities)
```

```
#Try to assign new value to index 0
#cities[0]='barcelona' - cannot assign values to tuple

List
--------
#list : takes duplicates : can assign the value
cities=['madrid','paris','london','madrid']
print(cities)

print(cities[0]) # madrid
cities[0]='new york'
print(cities[0]) # new york

Set
---------
#set: do not allow duplicates
cities={'madrid','paris','london','madrid'}
print(cities)
print(len(cities)) #3

dictionary
----------
#dictionary : does not take duplicates
employee={
   "empno":"3343",
   "name":"harry potter",
   "city": "london",
   "empno":"3432",
   "project": ["banking","ecom"]
```

```
}
print(employee) #{'empno': '3343', 'name': 'harry potter', 'city': 'london'}
print(employee["city"]) #london

Loops
--------

#for loop

list=['harry','ronald','hermione']

for n in list:
   print(n)

#for using range function

for i in range(0,len(list)):
   print(list[i])

#for using range without 0

for i in range(len(list)):
   print(list[i])

#while loop

x=0
while x<len(list):
   print(list[x])
```

```
  x=x+1
```

bool

```
#bool function : boolean
list=["apple"]

bool(list) #true

x=0
bool(x) #false
```

Working with Complex Numbers

```
#complex numbers

c1=5+4j   #5: real part, 4j: imaginary part
c2=4+3j

print('Complex Number: Real Part is: ' , c1.real)
print('Complex Number: Imaginary Part is: ' , c1.imag)
print('Complex Number: Conjugate Part is: ' , c1.conjugate())

print('Addition of 2 complex numbers is: ', c1+c2)
print('Subtraction of 2 complex numbers is: ', c1-c2)
print('Multiplication of 2 complex numbers is: ', c1*c2)
print('Division of 2 complex numbers is: ', c1/c2)
```

if-elif-else statement

```
price = 50
quantity=5
#if statement
if ((price*quantity))<300:
    print("price*quantity is less than 300")
    print("price: " , price)
    print("quantity " , quantity)

#if-else statement
if price>100:
    print("price is greater than 100")
else:
    print("price is less than 100")

#if-elif-else
if price>100:
    print("price greater than 100")
elif price == 100:
    print("price is = 100")
else:
    print("price is less than 100")
```

Loops

```
#for-while-break-continue

nums=[10,20,30,40,50]



for i in nums:
    print(i)

numst=(10,20,30,40,50)

for i in nums:
    print(i)



#iterating through string

str="python"

for c in str:
    print(c)



#iterating through dictionary
nums = {
    1:"one",
    2:"two",
    3:"three"
}
```

```
for k,v in nums.items():
    print("key: " , k , " value: " , v)



#for with break
for i in range(1,5):
    if i>2:
        break;
    print(i)



#for with continue
for i in range(1,5):
    if i>3:
        continue
    print(i)   #1 2 3



#for with else
for i in range(2):
    print(i)
else:
    print('end..')



#while with else
num=0
while num <3:
```

```
    print(num)
    num = num+1
else:
    print('ends..')
```

Taking Input from the User:

```
marks = input('Enter the marks')
print('you entered ' , marks)
```

String

```
#String in Python

greeting="""Hello User,
How R U?"""

print(greeting)
```

Slicing

```
s="harry potter"

#Slicing
print(s[2:5]) #rry : prints the characters starting from 2 to 4

print(s[:]) #prints the entire string
```

```
print(s[2:]) #prints the string starting from 2 till the end

print(s[-2:]) #prints the last 2 characters

print(s[-4:-2]) # -4 -3 : tt : prints the characters from behind starting from 4 to 2



#String functions

str = " harry potter "
str=str.strip()
print(str) #deletes the spaces from before and after the string, not from the middle

print(str.upper()) #displays the string in upper case
print(str.lower()) #displays the string in lower case

str = str.replace("potter","weasley")
print(str) #replaces "potter" with "weasley"

str = str.replace(" ", "*")
print(str)  #replaces " " with "*"

list = str.split("*")
print(list) #splits string using "*" as delimiter and returns list

print(str.find('w')) #returns the index of the character "w"
print(str.find('i')) #returns -1 if the character "i" is not present
```

```
str ="1234"
print(max(str)) #gives 4 as 4 is the max number in the string
print(min(str)) #gives 1  as 1 is the min number in the string
```

Math Package - working with Numbers

```
import math

ans = math.sqrt(16)
print(ans)

#floor gives the lowers value of the range of the number
# the range of 5.2 is 5 and 6.
#so floor will be 5 and 6 will be ceil.

ans = math.floor(3.4) # 3 : 3.8 : 4
print(ans)

ans = math.ceil(3.4) # 3 : 3.8 : 4
print(ans)

print(math.pow(3,2)) #pow calculates to the power. 3^2 = 9
```

Displaying the Calendar

```
import calendar

cal = calendar.month(2022,3)
```

```
print(cal)



import time

print(time.localtime())
print(time.localtime().tm_hour)

current_time = time.asctime(time.localtime())
print(current_time)



Working with strftime
---------------------

from datetime import datetime
#from module import package
#module and package both have name datetime

now = datetime.now()
print(now)

year = now.strftime("%Y")
month = now.strftime("%m")
date = now.strftime("%d")

time = now.strftime("%H:%M:%S")

print(year,"/",month,"/",date )
```

```
print(time)
```

Object Orientation in Python

Class & Object

```
class A:
    x=5

    def m1(self):
        print('m1')

    def m2(self,x):
        print(x)

a=A()
print(a.x)
a.m1()
a.m2(10)
```

Constructors in Python

```
class A:
    marks=0
    total=0

    def __init__(self,x,y): #this is a constructor
```

```
        self.marks=int(x)
        self.total=int(y)

    def compute(self):
        percent = (self.marks*100) / self.total
        print("you scored: ", percent , " percent")



x=input('Enter the marks')
y=input('Enter the total')
a=A(x,y)

a.compute()
```

Exception Handling

```
class A:
    x=0
    y=0

    def __init__(self,x,y):
        self.x=x
        self.y=y

    def compute(self):
        try:
            z = self.x / self.y
            print('ans is ', z)
```

```
        except ArithmeticError:
            print('divide by zero is not allowed')

a=A(2,0)
a1=A(4,2)
a.compute() #ArithmeticError
a1.compute()
```

Inheritance

```
#inheritance

class A:
    def m1(self):
        print('m1 in A')

class B(A): #B extends A : #B gets m1() from A

    def m2(self):
        print('m2 in B')

b=B()
b.m1()
b.m2()
```

Self Defined Exception

```
#Self defined Exception
class MarksError(Exception):
    pass

class A:
    marks=0
    total=0

    def __init__(self, x,y):
        self.marks = x
        self.total = y

    def compute(self):
        try:
            if self.marks>self.total:
                raise MarksError()
            percent = (self.marks * 100) / self.total
            print('percent: ', percent)
        except MarksError:
            print('Marks cannot be greater than total')
        except ArtithmeticError:
            print('divide by zero not allowed')

a=A(102,100)
a.compute()

a1=A(98,100)
a1.compute()
```