Variables

```
In [1]: # variables are the containers which store specific value
        # variable_name = value
```

```
In [2]: variable_name = 10
        variable_name
```

```
Out[2]: 10
```

```
In [3]: type(variable_name)
```

```
Out[3]: int
```

```
In [ ]: # in-build
        # user defined functions
```

```
In [ ]:
```

```
In [ ]:
```

Data-types

```
In [ ]: # integer  ----> 10, 20, 30,22
        # float   -----> 1.2, 1.5, 1.6, 0.22
        # boolean ---> True / False
```

```
In [4]: integer_value = 10
        integer_value
```

```
Out[4]: 10
```

```
In [5]: float_value = 1.2
        float_value
```

```
Out[5]: 1.2
```

```
In [7]: boolean_value = True
        boolean_value
```

```
Out[7]: True
```

In [8]:

```
string_value = 'mehul wankhede'
string_value
```

Out[8]:

```
'mehul wankhede'
```

In [9]:

```
type(integer_value), type(float_value), type(boolean_value), type(string_value
```

Out[9]:

```
(int, float, bool, str)
```

Type Casting

In [ ]:

```

```

In [ ]:

```
# it is process to convert one datatype to another datatype
# integer ---> float
# float ----> interger
```

In [17]:

```
changed_float_value = float(integer_value)
type(changed_float_value), changed_float_value
```

Out[17]:

```
(float, 10.0)
```

In [18]:

```
changed_int_value = int(float_value)
type(changed_int_value), changed_int_value
```

Out[18]:

```
(int, 1)
```

In [23]:

```
# boolean 0 and 1
# any value which is greater than 1 or less than 0 will be considered as True
# 0 ----> False
```

In [28]:

```
float_value = 0.000001
bool(float_value)
```

Out[28]:

```
True
```

In [30]:

```
# string value can be converted into integer or float if it contains only floa
stirng_value = '100'
int(stirng_value)
```

Out[30]:

```
100
```

In [32]:

```
string_value = '20.5'
float(string_value)
```

Out[32]:

```
20.5
```

In [34]:

```
string_value = 'mehul wankhede'
```

In [35]:

```
str(199)
```

Out[35]:

```
'199'
```

In [36]:

```
str(1000.222)
```

Out[36]:

```
'1000.222'
```

In [37]:

```
str(True)
```

Out[37]:

```
'True'
```

User input

In [ ]:

```
#input() ----> use to take input from the user
# input() ----> considers all values as a string value
```

In [50]:

```
# take two numbers from the user and print there multiplication

var_one = input('Enter first number:')
var_two = input('Enter second number:')
var_one =  float(var_one) # type cast to interger  converts string value into
var_two = float(var_two)   # type cast to integer  converts string value into
print('datatype of var_one ', type(var_one))
print('datatype of var_two ', type(var_two))
print('multiplication of two number is:', var_one * var_two)
```

```
Enter first number:1.2
Enter second number:1.2
datatype of var_one  <class 'float'>
datatype of var_two  <class 'float'>
multiplication of two number is: 1.44
```

In [51]:

```
int('100.3')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [51], in <cell line: 1>()
----> 1 int('100.3')

ValueError: invalid literal for int() with base 10: '100.3'
```

In [52]:

```
float('100.3')
```

Out[52]:

```
100.3
```

python operators

```
In [53]: # operator is a special symbol which perform specific operation on given two e
         # *   ----> multiplication
         # + -----> addition
         # -   -----> substraction
         # / ---> division
         # **   ---> exponential
         # // ----> floor division
         # % --->  Modulus
```

```
In [54]: 2 *5

Out[54]: 10
```

```
In [55]: 2 + 5

Out[55]: 7
```

```
In [56]: 2 / 5

Out[56]: 0.4
```

```
In [59]: 12 / 5

Out[59]: 2.4
```

```
In [66]: 19 // 5

Out[66]: 3
```

```
In [67]: 19 / 5

Out[67]: 3.8
```

```
In [61]: 2 ** 5

Out[61]: 32
```

```
In [62]: 2 ** 4

Out[62]: 16
```

```
In [63]: 12 % 5

Out[63]: 2
```

```
In [68]: 35 % 10

Out[68]: 5
```

```
In [69]: # odd number
         # odd number is number which is not divisible by 2
         # even number
         # even number is number which is divisible by 2
```

```
In [70]: 11 % 2

Out[70]: 1
```

Conditional Operators

- Equals: a == b
- Not Equals: a != b
- Less than: a < b
- Less than or equal to: a <= b
- Greater than: a > b
- Greater than or equal to: a >= b

```
In [71]: # operands   --- on which the operations will be performed
         # operators   --- which will perform the operation
```

```
In [72]: # conditional operators gives always boolean (True/ False)
```

```
In [73]: 6 > 5

Out[73]: True
```

```
In [74]: 6 < 5

Out[74]: False
```

```
In [75]: 6 == 6

Out[75]: True
```

```
In [76]: 6 == 6.0

Out[76]: True
```

```
In [77]: 6 == 6.1

Out[77]: False
```

In [83]:

```
# is keyword checks the magnitude and type of a value
6 is 6.0
```

```
<>:2: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:2: SyntaxWarning: "is" with a literal. Did you mean "=="?
C:\Users\admin\AppData\Local\Temp\ipykernel_10060\86231528.py:2: SyntaxWarnin
g: "is" with a literal. Did you mean "=="?
  6 is 6.0
```

Out[83]:

```
False
```

In [84]:

```
6 != 6
```

Out[84]:

```
False
```

In [86]:

```
# and returns True if all conditons are True
# or returns True if any one of the conditions is True
```

In [87]:

```
 4 > 2 and 4 < 5
```

Out[87]:

```
True
```

In [88]:

```
4 > 2 or 2 > 3
```

Out[88]:

```
True
```

In [89]:

```
True and True and True and False
```

Out[89]:

```
False
```

In [90]:

```
True or False or False or False
```

Out[90]:

```
True
```

In [91]:

```
True and True and True or False
```

Out[91]:

```
True
```

if ...else

In [93]:

```
# if conditon1 and/or conditon2 and/or .....
# if conditions always runs when Results is True
# else condition always runs when results is False
```

In [94]:

```
a = 10
b = 5
c  = 3

if (a> b) and (a > c):
    print('a is greater than b and c ')
else:
    print(' a is not greater than b and c')
```

```
a is greater than b and c
```

In [95]:

```
a = 10
b = 15
c  = 8
if (a> b) and (a > c):
    print('a is greater than b and c ')
else:
    print(' a is not greater than b and c')
```

```
 a is not greater than b and c
```

if ...elif ...else ladder

In [97]:

```
a = 4
b = 3
c  = 8

if (a > b) and (a > c):
    print('a is greater b and c')
elif(b > a) and (b > c):
    print('b is greater than a and c')
else:
    print('c is greater than a and b')
```

```
c is greater than a and b
```

In [110]:

```
# A grade marks= 80+

# B grade marks = 65 to 80

# c grade  marks = 45 to 65

# d graee marks  = 35 to 45

# input user
percentage  = input('Enter your percentage:')
percentage = int(percentage)   # type cast convert string to integer
if (percentage > 80):
    print('Grade A')
elif (percentage >= 65) and (percentage < 80):
    print('Grade B')
elif (percentage >= 45) and (percentage < 65):
    print('Grade C')
elif (percentage >= 35) and (percentage < 45):
    print('Grade D')
else :
    print('Fail')
```

```
Enter your percentage:36
Grade D
```

In [ ]: