In [1]:
```
print("Hello vits !!")
print(231)
print(12.5)
```

```
Hello vits !!
231
12.5
```

In [2]:
```
print(1,2,3)
```

```
1 2 3
```

In [3]:
```
print(1,2,3,sep='\n')
```

```
1
2
3
```

Data types:

- int
- float
- str
- complex

Variable naming rules

In [5]:
```
a = 10
b = 'green'
c = 9.5
type(a)
```

Out[5]:
```
int
```

input methods

- static method
  - direct assignment
- dynamic method
  - runtime input given by the user.
  - input("prompt")
    - by default it will take any input in string type.
  - if we want to take an integer as input
    - int(input())
  - for float type input
    - float(input())

In [9]:
```
1 int(input()) # only integer value
```

...

In [11]:
```
1 float(input()) # both int value or float value
```

```
19
```

Out[11]:
```
19.0
```

Type Conversions

- int --> float or str
- float --> int or str
- str --> we can't convert either int or float.

In [12]:
```
1 # int to --> float or str
2 float(a)
```

Out[12]:
```
10.0
```

In [13]:
```
1 str(a)
```

Out[13]:
```
'10'
```

In [14]:
```
1 type(a)
```

Out[14]:
```
int
```

In [15]:
```
1 # float  --> int or str
2 int(c)
```

Out[15]:
```
9
```

In [16]:
```
1 str(c)
```

Out[16]:
```
'9.5'
```

In [17]:
```
1 int(b)
```

...

In [25]:
```
1 float(b)
```

...

In [38]:
```
j = '2022'
type(j)
```

Out[38]: str

In [39]:
```
int(j)
```

Out[39]: 2022

In [40]:
```
float(j)
```

Out[40]: 2022.0

In [41]:
```
# name,age,height
name = input("Enter ur name: ")
age = int(input("Enter ur age: "))
height = float(input("Enter ur height in inches: "))
print(name,age,height,sep='\n')
```

```
Enter ur name: viswa
Enter ur age: 19
Enter ur height in inches: 5.4
viswa
19
5.4
```

In [42]:
```
print(b)
```

```
green
```

In [45]:
```
# multi variable assignment
a,b,c,d = 10,20,30,40
print(b,d,a,c)
```

```
20 40 10 30
```

In [46]:
```
x,y,z = 'white',15,6.4
print(x,y,z)
```

...

In [49]:
```
# multi varibale assignment with same value
p=q=r=200
print(r)
print(p)
print(q)
```

...

In [50]:
```
a,s,d,f = 12,32,43
print(d)
```

...

In [51]:
```
g,h,j = 'green','red','orange','purple'
print(g,j)
```

. . .

Operators in Python

- We have 7 types of operators.
- Operator: a special symbol which carries out a mathematical operation between opearands.

Arithmetic Operators

Operators	Meaning	Example	Result
+	Addition	4 + 2	6
–	Subtraction	4 – 2	2
*	Multiplication	4 * 2	8
/	Division	4 / 2	2
%	Modulus operator to get remainder in integer division	5 % 2	1
**	Exponent	5**2 = 5^2	25
//	Integer Division/ Floor Division	5//2 -5//2	2 -3

In [53]:
```
## Arihtmetic Operators
a = int(input("Enter a number: "))
b = int(input("Enter another number: "))
print(a+b,a-b,b*a,a/b,a//b,a**b,a%b,b%a)
```

. . .

In [55]:
```
9%3 # a>b
5%10 # b>a
```

Out[55]: 5

Assignment Operators

Operator	Example	Equivalent Expression (m=15)	Result
=	y = a+b	y = 10 + 20	30
+=	m +=10	m = m+10	25
-=	m -=10	m = m-10	5
*=	m *=10	m = m*10	150
/=	m /=10	m = m/10	1.5
%=	m %=10	m = m%10	5
**=	m**=2	m = m**2 or $m = m^2$	225
//=	m//=10	m = m//10	1

In [58]:
```
k = 8

k += 5
print(k) # 13

k-=10
k*10
print(k) # 3

k %= 2
print(k) # 1

k += 20
k*=5
print(k) # 105
```

. . .

Comparison Operators

Python Comparison Operators		
Operator	**Name**	**Example**
==	Equal	a == b
!=	Not equal	a != b
>	Greater than	a > b
<	Less than	a < b
>=	Greater than or equal to	a >= b
<=	Less than or equal to	a <= b

In [59]:
```
x = 20
y = 300
print(y>x,x<=y,x<y,x!=y,y==x,x>=y)
```

```
True True True True False False
```

In [60]:
```
x = 20
y = 10+10
print(y>x,x<=y,x<y,x!=y,y==x,x>=y)
```

```
False True False False True True
```

Python Logical Operators

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

A	Not A
True	False
False	True

In [63]:
```
# and,or,not
p,q,r,s = 16,31,12,55
print(p>r and q==s)
```

```
False
```

In [64]:
```
print((p<r or q==s) and (r>s and p>=r) or (not(s==r)))
```

```
True
```

In [ ]:
```
(s==55 or p>21) and (not(r==12) and q!=s) or (r<=p)
```

In [66]:
```
True and False or True
```

Out[66]:
```
True
```

Bitwise Operators

Operator	Meaning
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR / Bitwise XOR
~	Bitwise inversion (one's complement)
<<	Shifts the bits to left / Bitwise Left Shift
>>	Shifts the bits to right / Bitwise Right Shift

Truth Tables

AND Truth Table

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

OR Truth Table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

XOR Truth Table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

NOT Truth Table

A	B
0	1
1	0

In [73]:
```
1 7 & 18
```

Out[73]: 2

In [ ]:
```
1
```

In [72]:
```
1 bin(18)
```

Out[72]: '0b10010'

In [68]:
```
1 int('10101010',2)
```

Out[68]: 170

In [69]:
```
1 oct(8992)
```

Out[69]: '0o21440'

In [70]:
```
1 hex(11)
```

Out[70]: '0xb'

In [71]:
```
int('f',16)
```

...

In [77]:
```
405|203
```

Out[77]: 479

In [75]:
```
bin(203)
```

Out[75]: '0b11001011'

In [ ]:
```
110010101
011001011
---------
010000001

111011111
```

In [76]:
```
int('111011111',2)
```

Out[76]: 479

In [82]:
```
# xor
403^189
```

Out[82]: 302

In [80]:
```
bin(189)
```

Out[80]: '0b10111101'

In [ ]:
```
110010011
010111101
---------
100101110
```

In [81]:
```
int('100101110',2)
```

Out[81]: 302

Type *Markdown* and LaTeX: α^2