In [2]:

```
print("Hello")
```

```
Hello
```

what is ".ipynb" extension ¶

ipynb file extension is used for computational notebooks that can be open with Jupyter Notebook. It stands for Interactive Python Notebook.

) Python is developed by- GUIDO VAN RASSUM

) at- Research Institute for Mathematics and CS in Netherlands.

) its source code is available under the GNU General Public Lisense(GPL)

Basics

I) Comment

In [7]:

```
#use this for single line comment
"""
this
is multiline
comment
"""
print("use tripple quoates for multiline comments ")
```

```
use tripple quoates for multiline comments
```

In [8]:

```
# this is a comment
text= "#this is not a comment"
print(text)
```

```
#this is not a comment
```

Type Casting

In [19]:

```
print(int("123"))

int("123abc") #does not work
```

```
123

---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
~\AppData\Local\Temp\ipykernel_5432\1160456250.py in <module>
      1 print(int("123"))
      2 
----> 3 int("123abc") #does not work

ValueError: invalid literal for int() with base 10: '123abc'
```

Boolian

In [22]:

```
not False
```

Out[22]:

```
True
```

In [24]:

```
not True
```

Out[24]:

```
False
```

In [33]:

```
a=500
if bool(a)==True:
    print("True")
elif bool(a)==False:
    print("False")   ###because bool of 0 is False and bool of 1 is True
else:
    print("nothing")
```

```
True
```

Dynamic Typing

In dynamic typing we dont have to decide before hand the data type of the variable. On the runtime compiler will decide the variable type

In [55]:

```
a=56
b=92
a="abcd" #this will replace the the above value of the variable
print(type(a))
print()
print(type(b))
```

```
<class 'str'>

<class 'int'>
```

Concatination

In [56]:

```
### concatenation between different types
```

In [60]:

```
int("1")+ 1
```

Out[60]:

```
2
```

In [59]:

```
int("1")+ "1"
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
~\AppData\Local\Temp\ipykernel_5432\1031076855.py in <module>
----> 1 int("1")+ "1"

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

In [65]:

```
a ="123"*10
type("123"*10)
print(a)
```

```
123123123123123123123123123123
```

Print Statement

In [76]:

```
age = 25
name = "asit"
print("My age is:",age)
```

```
My age is: 25
```

Fstring method

In [85]:

```
print(f"My name is: {name} and " + "\n" + f"I'am {age} years old.")
```

```
My name is: asit and
I'am 25 years old.
```

.Fromat method

In [86]:

```
print("My name is {} and i'am {} years old".format(name,age))
```

```
My name is asit and i'am 25 years old
```

In [ ]:

```
##or
print("My name is {firstname} and i'am {your_age} years old".format(firstname=name,your_a
```

Input Function

In [92]:

```
"""
by default value of input function is string
so in order to convert it to string we have to do type casting
"""
typ=input()
```

```
123
```

In [93]:

```
type(typ)
```

Out[93]:

```
str
```

In [ ]:

```

```