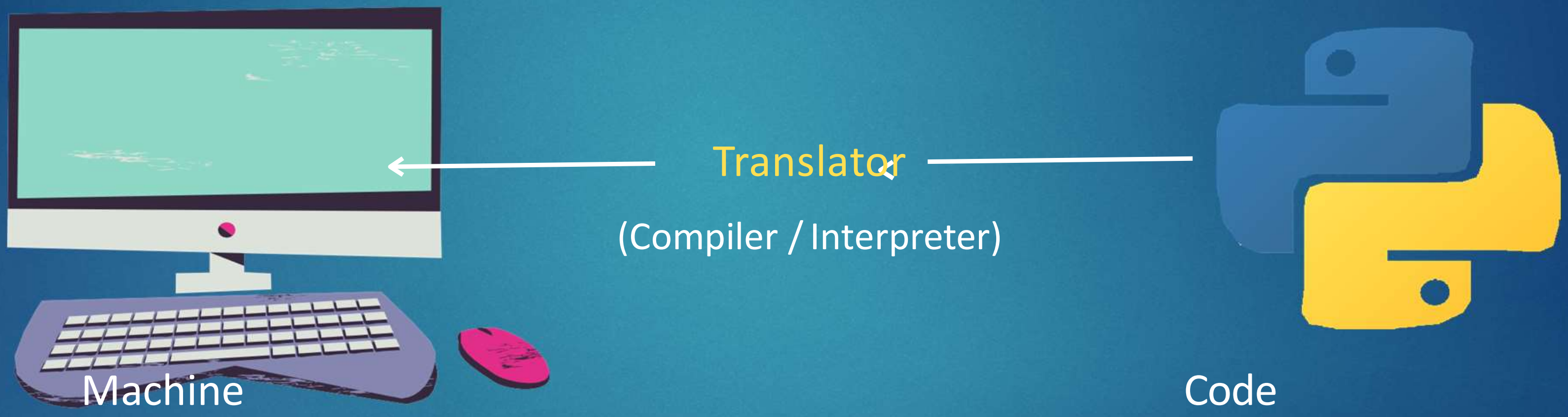
Programming



Why Python?

- Python is simple & easy
- Free & Open Source
- High Level Language
- Portable

Our First Program

```
print("Hello IIIT !")
```

Python Character Set

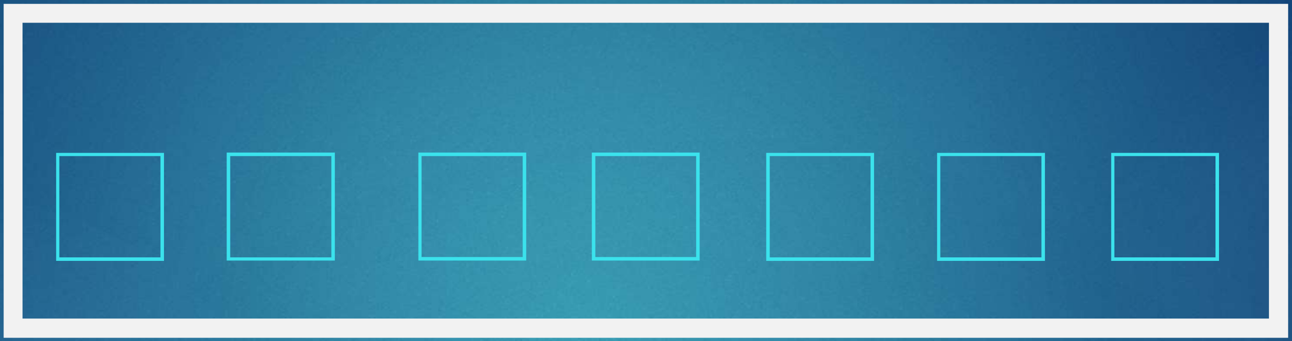
- Letters – A to Z, a to z
- Digits – 0 to 9
- Special Symbols - + - * / etc.

Variables

A variable is a name given to a memory location in a program.

```
name = "Abhijeet"
age = 25
Designation = "Research Scholar"
Institute = “IIIT Naya Raipur"
```

Memory



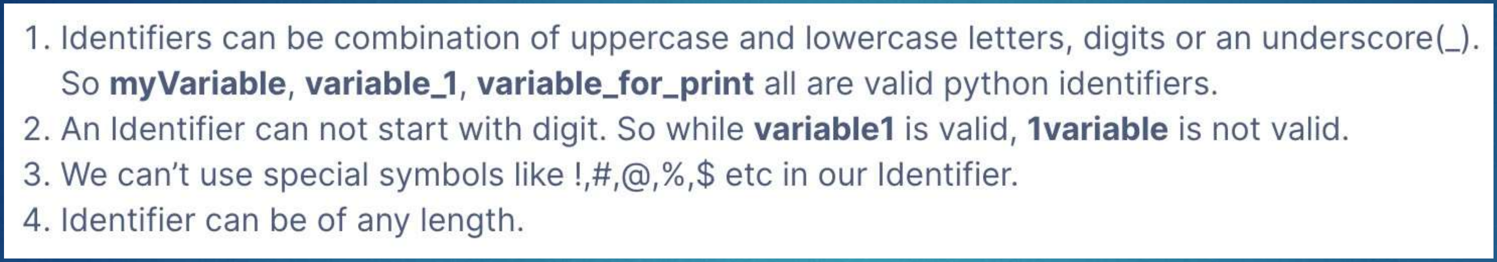
name = "Abhijeet"

age = 25

Designation = "Research Scholar"

Institute = “IIIT Naya Raipur"

Rules for Identifiers

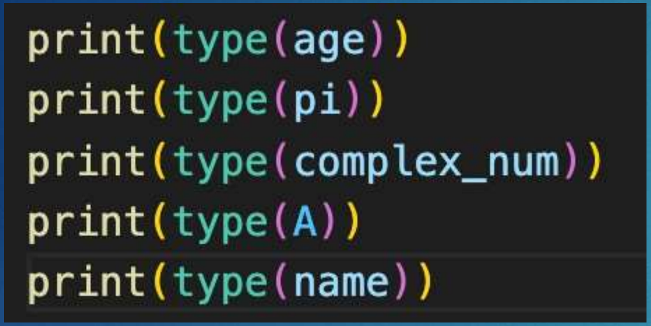


1. Identifiers can be combination of uppercase and lowercase letters, digits or an underscore(_). So **myVariable**, **variable_1**, **variable_for_print** all are valid python identifiers.
2. An Identifier can not start with digit. So while **variable1** is valid, **1variable** is not valid.
3. We can't use special symbols like !,#,@,%,$ etc in our Identifier.
4. Identifier can be of any length.

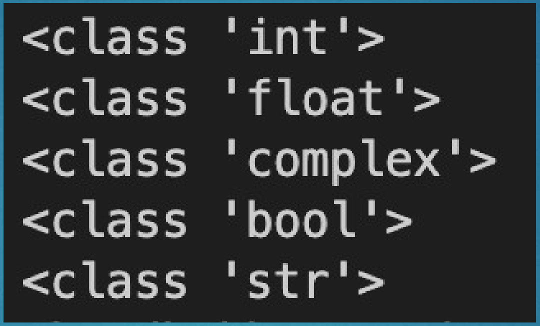
Data Types

- Integers
- String
- Float
- Boolean
- None

Data Types



```
print(type(age))
print(type(pi))
print(type(complex_num))
print(type(A))
print(type(name))
```

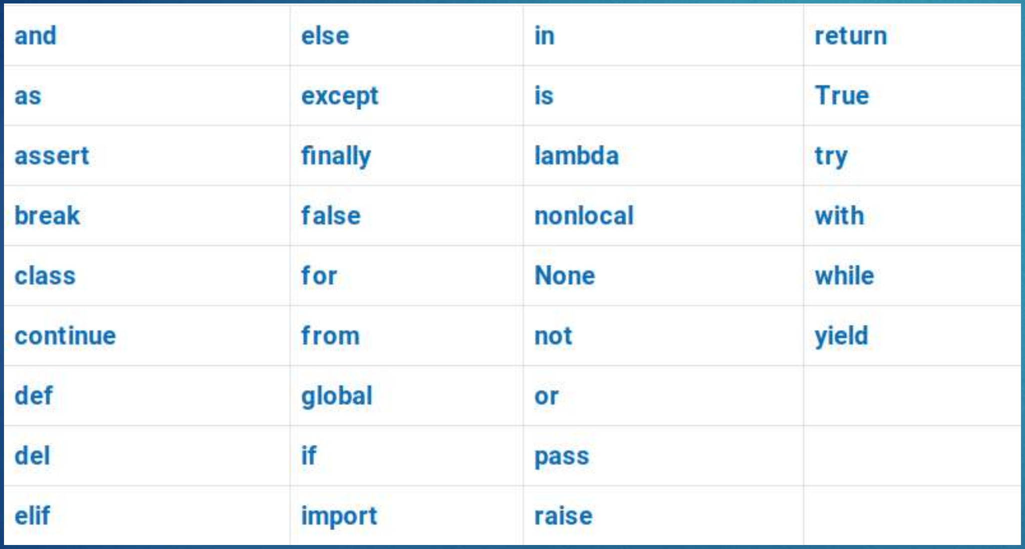


```
<class 'int'>
<class 'float'>
<class 'complex'>
<class 'bool'>
<class 'str'>
```

Keywords

Keywords are reserved words in python.

*False should be uppercase



and	else	in	return
as	except	is	True
assert	finally	lambda	try
break	false	nonlocal	with
class	for	None	while
continue	from	not	yield
def	global	or	
del	if	pass	
elif	import	raise	

Print Sum

Comments in Python

```
# Single Line Comment

"""
Multi Line
Comment
"""
```

Types of Operators

An operator is a symbol that performs a certain operation between operands.

- Arithmetic Operators (+ , - , * , / , % , **)
- Relational / Comparison Operators (== , != , > , < , >= , <=)
- Assignment Operators (= , +=, -= , *= , /= , %= , **=)
- Logical Operators (not , and , or)

Type Conversion

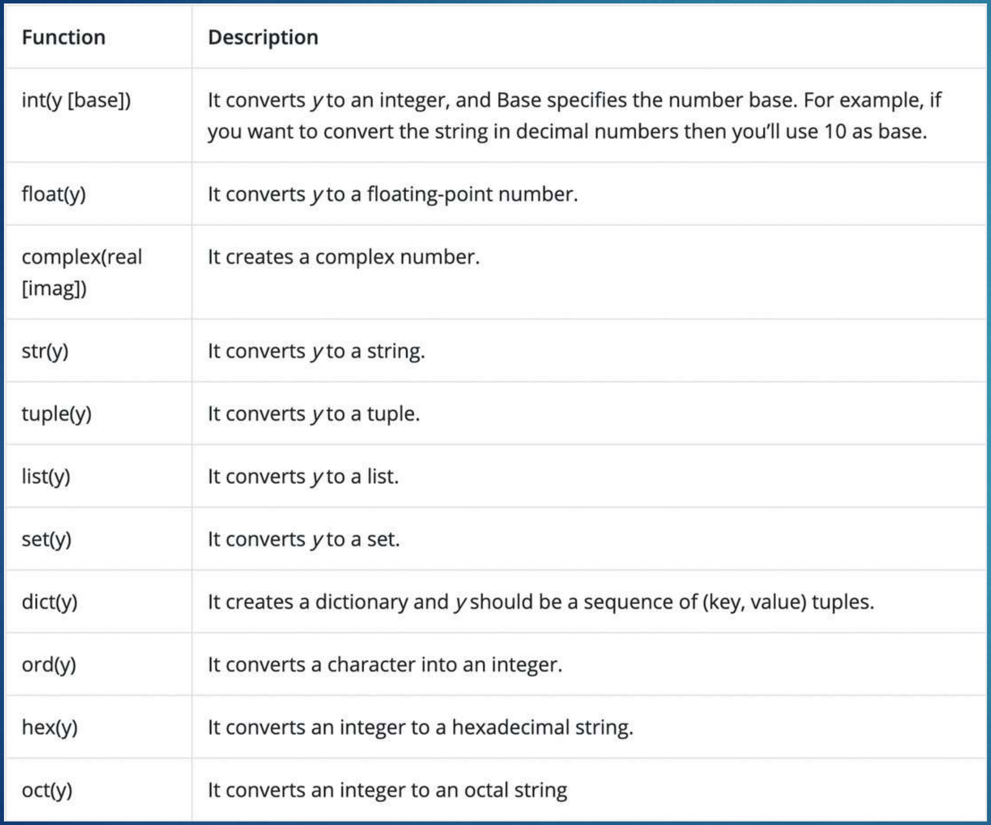
```
a, b = 1, 2.0
sum = a + b
```

```
#error
a, b = 1, "2"
sum = a + b
```

Type Casting

```
a, b = 1, "2"
c = int(b)
sum = a + c
```

Type Casting



Function	Description
int(y [base])	It converts *y* to an integer, and Base specifies the number base. For example, if you want to convert the string in decimal numbers then you'll use 10 as base.
float(y)	It converts *y* to a floating-point number.
complex(real [imag])	It creates a complex number.
str(y)	It converts *y* to a string.
tuple(y)	It converts *y* to a tuple.
list(y)	It converts *y* to a list.
set(y)	It converts *y* to a set.
dict(y)	It creates a dictionary and *y* should be a sequence of (key, value) tuples.
ord(y)	It converts a character into an integer.
hex(y)	It converts an integer to a hexadecimal string.
oct(y)	It converts an integer to an octal string

Input in Python

input( ) statement is used to accept values (using keyboard) from user

```
input( )  #result for input( ) is always a str

int ( input( ) )  #int

float ( input( ) )   #float
```

give code eg of all 3

Let‘s Practice

```
age_dict = {
    "Alice": 25,
    "Bob": 30,
    "Charlie": 22,
    "David": 28,
    "Emma": 35
}

# Ask the user to enter a name
name = input("Enter a name to get their age: ")

# Use get method to retrieve age or default message
print(age_dict.get(name, f"Sorry, the age for '{name}' is not
available."))
```

Let‘s Practice

Write a Program to input 2 numbers & print their sum.

Let‘s Practice

WAP to input side of a square & print its area.

Let‘s Practice

WAP to input 2 floating point numbers & print their average.

Let's Practice

WAP to input 2 int numbers, a and b.

Print True if a is greater than or equal to b. If not print False.