Python Introduction

→ HLL

→ Interpreter converts python to machine readable code.

→ print("Hello world")

↓

print() → is a function

Python Character Set:-

→ Python can process all ASCII and Unicode characters.

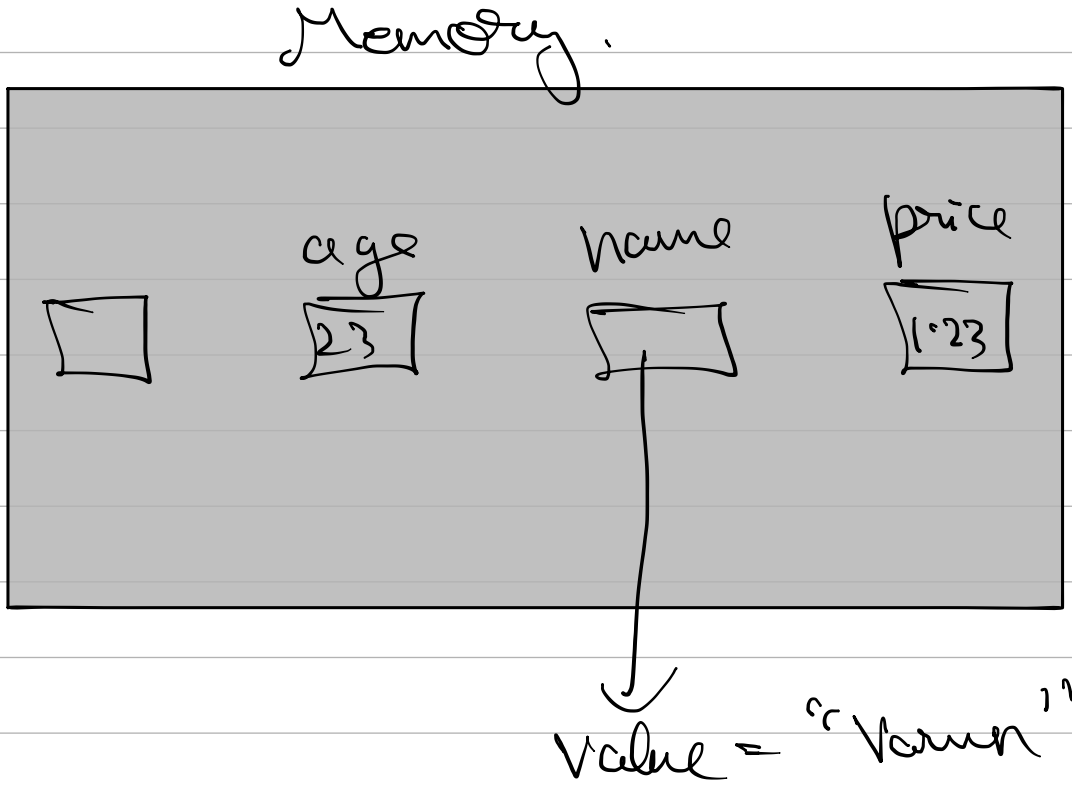
→ A-Z, a-z, 0-9, Special symbols + - * / etc.

→ White spaces, newline, tab allowed.

Variable :-

→ It is a name given to a memory location in a program.

name = "Varun"
age = 23
price = 1.23



*). Identifiers like variable names

can have upper and lower case letters and '_' and digits.

*). Variable names cannot begin with digit.

*). Special symbols are not allowed like #, @, !, etc.

```
print(type(name))
    ↓
prints <str>

print(type(age))
    ↓
prints <int>

print(type(float))
    ↓
prints <float>.
```

Data Types

*). Integer

*). String → ' ' , " " , """ """ } can be written in any ways.

*). Float

*). Boolean. → True, False } : cannot be true and false.

*). None. → a = None
↳ means a variable has no valid value.

Keywords

They are reserved words in python.

→ They cannot be used as identifier.

→ and, for, in, else, if, etc.
lambda, nonlocal, etc.

*). Python is case-sensitive language

Eg:- apple is different from Apple.

Comments in python

→ Used for single line comment.

"""
multi
line
comment
"""

*). Comments do not execute.

Types of Operators

- Arithmetic Operators → +, -, *, /, %, **
- Relational Operators → ==, !=, >, <=, >=, <
- Assignment Operators → =, +=, *=, -=, /=
- Logical Operators → not, and, or.

** → means to the power.

*) Relational operators return either true or false.

a = 10
b = 10

print(a == b)

↓

outputs → True

Logical Operators

not, and, or.

1). not :-

print(not True)

↓

outputs -> False.

a = 50

b = 40

print (not (a>b))

(a>b) -> true

↓

outputs -> false.

2). AND :- a = True

b = True.

print (a and b)

↓

outputs -> True.

a = True

b = False

print (a and b)

↓

outputs -> False.

3). OR :-

a = True

b = False

print (a or b)

↓

outputs -> True.

Type Conversion

*). Conversion occurs automatically in python.

*). Casting is done manually.

Conversion:-

Eg-1:-

a = 1 -> int

b = 2.0 -> float

Sum = a + b.

print (sum)

↓

outputs -> 3.0 -> float.

Eg-2:-

a = "2" -> String

b = 3.0 -> float

Sum = a + b.

print (sum)

↓

Error -> Because can only concatenate string with string. Not

Castings

```
a = int("2")
b = 2.25
sum = a+b.
print(sum) → prints 4.25
```

↓

Does not give error

Input in Python

input()

↓

is used.

```
name = input("Enter your name")
print(name).
```

input() → can take any value (int, float, string, etc).

→ input values are implicitly converted to string.

```
a = input("Number")
a = 10 → If we give, the 10 is stored as string.
```

To take a required datatype as input:-

```
a = int( input("Enter a number") )
```

a = 10 -> we give and is stored as 'int' instead of string.