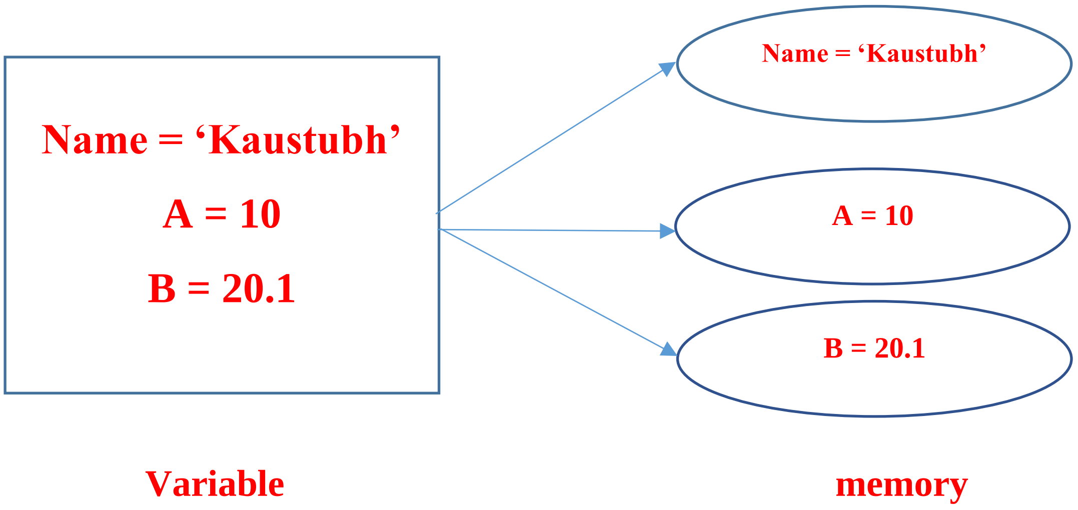
# 2. Types of Variables

Welcome to the topic "Types of variables in Python" language. Now before we move on directly to the topic. It is important to have basic introduction to Variables , how to assign values to them ,what are some rules to follow and some basic methods.

## 1. Introduction to variables:

Variables and data types in python as the name suggests are the values that vary. In a programming language, a variable is a memory location where you store a value. The value that you have stored may change in the future according to the specifications.

A Variable in python is created as soon as a value is assigned to it. It does not need any additional commands to declare a variable in python.



Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

### 1.1 Rules for variable name:

There are a certain rules that we have to keep in mind while declaring a variable:

1. The variable name cannot start with a number. It can only start with a character or an underscore.
2. Variables in python are case sensitive.
3. They can only contain alpha-numeric characters and underscores.
4. No special characters are allowed.

***Example:***

***Correct :*** name

X

***Incorrect :*** @name

&

## 2. Assigning values to variables:

### 2.1 Assigning a single value

Python is not “statically typed”. We do not need to declare variables before using them, or declare their type. A variable is created the moment we first assign a value to it. Assigning values to a variable in python is very straight forward, which means we can directly assign values to any variable with help of Assignment operator (=). Provided the rules of naming a variable are kept in mind while naming any variable. *(as mentioned in 1.2)*

***Example:***

***Correct :*** name = “Jimmy”

X = 55

***Incorrect :*** @name = “Jimmy”

& = 74

### 2.2 Assigning a Single value to multiple variables :

Python allows to assign a single value to several variables simultaneously. Hence it is possible to assign a single value to multiple variable.

***Example:***

a = b = c = 10

p = q = r = 25

In the above examples, all three respective variables are assigned a single value simultaneously.

## 3. Python Scope of Variables

The location where we can find a variable and also access it if required is called the ***scope of a variable***.

### 3.1 Global variables:

***Global variables*** are the ones that are defined and declared outside any function and are not specified to any function. They can be used by any part of the program.

***Example:*** # This function uses global variable s

```
def f( ):
    print(s)

# Global scope
s = "Hello world"
f( )
```

***Output :*** Hello world

### 3.2 Local variables

***Local variables*** are the ones that are defined and declared inside any function and are not specified to any function. They can be used by any part of the program.

***Example :***

```
def f():
    s = "Hello World" #local scope
    print(s)
f()
print(s)
```

***Output :*** Hello World

## 4. Instance variables & Class(Static) variables

Now in Object oriented programming we have two different types of variables.

### 4.1 Instance variable :

***Instance variable*** declared inside the constructor method of class **(the __init__ method).** They are tied to the particular object instance of the class, hence the contents of an instance variable are completely independent from one object instance to the other.

***Example :***

```
class Car:
    wheels = 4    #  Class variable
    def __init__(self, name):
        self.name = name    # <- Instance variable
```

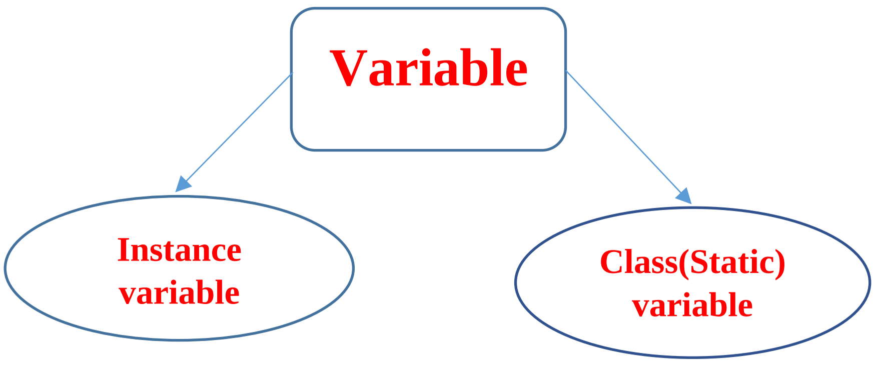


*Fig. Types of Variables*

### 4.2 Class variable

***Class Variables*** are declared inside the class definition (but outside any of the instance methods). They are not tied to any particular object of the class, hence shared across all the objects of the class. Modifying a class variable affects all objects instance at the same time.

***Example :***

```
class Car:
    wheels = 4    #  Class variable
    def __init__(self, name):
        self.name = name    # <- Instance variable
```

Now please refer the following **YouTube video**: https://youtu.be/RSQjxL5WRNM

## References :

**https://medium.com/python-features/class-vs-instance-variables-8d452e9abcbd**

**https://www.edureka.co/blog/variables-and-data-types-in-python**

**https://www.learnpython.org/en/Variables_and_Types**

**https://www.geeksforgeeks.org/python-scope-of-variables/?ref=rp**

**https://www.geeksforgeeks.org/private-variables-python/?ref=rp**