Types of Variables:

- Variable is a named memory location.
- The purpose of variable is to store the data.

In python variables are classified into 5 types.

1. Local variables.
2. Global Variables.
3. Parameterized Variables.
4. Instance Variables.
5. Static Variables.

1. Local variables.

The variables which are declared inside of a method declaration is called as local variables.

Local variables we can access within the method only.

2. Global Variables.

The variables which are declared outside of a class declaration is called as global variables.

Global variables we can access in entire program.

3. Parameters Variables.

- The variables which are declared at the time of method declaration is called as Parameterized variables.
- Parameter variables we can access with in that method only.

Example: Local variables + Global variables + Paramer variables program

```
b = 20 # global variable
def f1(c): # parameter variable
```

```
    a = 10  #  local  variable
    print('a value is :',a)
    print('b value is :',b)
    print('c value is :',c)
    print()

f1(15)
# print('a value is :',a)  # NameError: name 'a' is not defined
print('b value is :',b)
# print('c value is :',c)  # NameError: name 'c' is not defined
```

Output:

```
a value is : 10
b value is : 20
c value is : 15
b value is : 20
```

Example Bank:

```
a  = 10 #  global variable
class  Customer:
    # static | class variables
    ifsc  =  'SBIN0001'
    bname = 'SBI'

    def read(self):
        # non-static | instance variables
        self.cid = 1001
        self.amt = 5000

    def  withdraw(self):
        wamt  =  10000  #  local variable
        self.amt = self.amt - wamt

    def   deposite(self, damt): # parameter variable
```

```
        self.amt = self.amt + damt
        #print('Total available amount is :',self.amt)

    def show(self):
        print('Customer Id :', self.cid )
        print('Available Amount is :',self.amt)
        print('Customer ifsc code :', Customer.ifsc)
        print('Customer Bank name :', Customer.bname)
        print('global variable value is :', a )

c1 = Customer()
c1.read()
c1.show()
print()

c1.deposite(3000)
c1.show()
print()
c1.withdraw()
c1.show()
```

Output:

```
Customer Id : 1001
Available Amount is : 5000
Customer ifsc code : SBIN0001
Customer Bank name : SBI
global variable value is : 10
Customer Id : 1001
Available Amount is : 8000
Customer ifsc code : SBIN0001
Customer Bank name : SBI
global variable value is : 10

Customer Id : 1001
```

Available Amount is : -2000
Customer ifsc code : SBIN0001
Customer Bank name : SBI
global variable value is : 10

Class and Instance Variables in Python (Explained with Examples)

--->> In this session we will understand the concept of class and instance variables in object-oriented programming within Python.

--->> Python being an object-oriented programming language allows variables to be used at class level or at the instance level.

--->> When the variables are declared at the class level they are referred to as class variables whereas when they are declared at instance level they are referred to as instance variables.

--->> We declare variables at class level when we expect all the objects (instances of a class) to share a single copy of a variable, whereas we declare variables at instance level when each object needs to have a separate copy of the variable so that they can have different values stored in them.

Class Variable :

--->> Class variables are shared by all the instances of the class (objects) because they are owned by the class.

--->> If it gets changed by any single object it changes for all objects.

--->> Class variables are declared within class outside of any methods, usually just below the class header.

Example:

```
class Book:

    book_type = "programming"
```

Here book_type is a class variable assigned with “programming”.

Let’s create an instance of the class and print the value of class variable.

```
class Book:
    book_type = "programming"
python_book = Book()
print(python_book.book_type)
```

Output:

Programming

We successfully get the value of the variable as the output. Now let’s add some more instances of the class to the program.

Example:

```
class Book:
   book_type = "programming"

python_book = Book()
java_book = Book()
c_book = Book()

print(python_book.book_type)
print(java_book.book_type)
```

```
print(c_book.book_type)
```

Output:

```
programming

programming

programming
```

You can clearly see that all the instances have the same value because book_type is a class variable.

Just like book_type, we can declare multiple class variables inside of the class and each of them will be accessible to each instance of the class.

Instance Variable:

Instance variables are the properties of objects. This means that every object or instance of the class maintains a separate copy of the variable.

Instance variables are declared within the method.

Example:

```
class Book:

    def __init__(self, bt):

        self.book_type = bt
```

Here book_type is an example of instance variable. When we create an object of the class we need to define the values of instance variable hence we have to pass the values for them during object declaration.

Let's see an example.

```
class Book:

    def __init__(self, bt):
```

```
        self.book_type = bt

python = Book("programming")

print(python.book_type)
```

Output:

```
programming
```

We have successfully got the output of the instance variable as what we have assigned at object declaration.

Now let’s create some more instances of the class.

Example:

```
class Book:

    def __init__(self, bt):

        self.book_type = bt

python = Book("programming")

Recepie = Book("Cooking")

Maharanapratap = Book("Biography")

print(python.book_type)

print(Recepie.book_type)

print(Maharanapratap.book_type)
```

Output:

```
programming
```

Cooking

Biography

You can see that each instance has a different value for its book_type property, which is passed during the declaration.

Hence it is proved that the instance variables are owned by the objects.

Using Class and Instance variable Together:

Now as we have some idea of class and instance variable let's see how they can be used together.

Example:

```
class Book:

    publication = "Pencil_Programmer_Publication"

    def __init__(self, bt):

        self.book_type = bt

python = Book("programming")

Maharanapratap = Book("Biography")

print("Python: " + python.book_type + " - " + python.publication)

print("Maharanapratap: " + Maharanapratap.book_type + " - " +
Maharanapratap.publication)
```

Output:

Python: programming - Pencil_Programmer_Publication

Maharanapratap : Biography - Pencil_Programmer_Publication

---->> In the above example, we have created two ‘Book’ objects of the same publication but of different categories.

---->> Since their publication are same but the type is different therefore we have defined publication as a class variable and book_type as instance variable so that it can hold different value for different instances of the class.

Conclusion:

--->> In conclusion, use a class variable when you want to have same value for all objects otherwise use an instance variable.

--->> Hope you have understood the concept of class and instance variables.