Java Classes and Objects:

Java is an object-oriented programming language. The core concept of the object-oriented approach is to break complex problems into smaller objects.

An object is any entity that has a state and behavior.

Class:

A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type.

```
class ClassName {
  // fields
  // methods
}
```

fields (variables) and methods represent the state and behavior of the object respectively.

- fields are used to store data
- methods are used to perform some operations

```
class Bicycle {

  // state or field
  private int gear = 5;

  // behavior or method
  public void braking () {
    System.out.println("Working of Braking");
  }
}
```

Methods:

A method is a block of code that performs a specific task.

In Java, there are two types of methods:

User-defined Methods: We can create our own method based on our requirements.

Standard Library Methods: These are built-in methods in Java that are available to use.

```
returnType methodName() {
  // method body
}
```

returnType - It specifies what type of value a method returns For example if a method has an int return type then it returns an integer value.

If the method does not return a value, its return type is void.

methodName - It is an identifier that is used to refer to the particular method in a program.

method body - It includes the programming statements that are used to perform some tasks. The method body is enclosed inside the curly braces { }.

Objects

An object is called an instance of a class.

An object consists of:

- State: It is represented by attributes of an object. It also reflects the properties of an object.
- Behavior: It is represented by methods of an object. It also reflects the response of an object with other objects.
- Identity: It gives a unique name to an object and enables one object to interact with other objects.

Creating an Object in Java:

We have to used the new keyword along with the constructor of the class to create an object.

```
className object = new className();

// for Bicycle class
Bicycle sportsBicycle = new Bicycle();

Bicycle touringBicycle = new Bicycle();
```

Constructor:

A constructor in Java is similar to a method that is invoked when an object of the class is created.

Unlike Java methods, a constructor has the same name as that of the class and does not have any return type. For example,

```
class Test {

  Test() {

    // constructor body

  }

}
```

Types of Constructor

In Java, constructors can be divided into

1. No-Arg Constructor
   Similar to methods, a Java constructor may or may not have any parameters (arguments).
   If a constructor does not accept any parameters, it is known as a no-argument constructor.

```
private Constructor() {
   // body of the constructor
}
```

2. Parameterized Constructor
   A Java constructor can also accept one or more parameters. Such constructors are known as parameterized constructors (constructor with parameters).

```
class Main {

  String languages;

  // constructor accepting single value
  Main(String lang) {
    languages = lang;
    System.out.println(languages + " Programming Language");
  }
}
```