Java classes / objects.

→ Class is like a blueprint for creating instances/objects.

eg. We have a class named "Student". We can create multiple objects/instances from this blueprint (Class)

This is how you can write a class. →

```
public class Student {
    public static void main (String[] args) {
        Student student1 = new Student();
        Student student2 = new Student();
    }
}
```

← This is how we can create objects from a class

→ You can also have multiple classes

Note: where ever you see the method main(), thats where the code is usually executed from. Main method is usually in default/first class.

e.g

```
public class SchoolRecords {
    public static void main (String[] args) {
        Student std1 = new Student();
    }
}

public class Student {
    String name;
    int age;
}
```

Class 1 ← SchoolRecords

← creating an object of Student class.

Class 2 ← Student

This is how you can organize classes better. Class Student has all the attributes and methods, while SchoolRecords holds the main() method where the code can be executed from.

Attribute:

ex =

```
public class Student {
    String name;
    int age;
    int x = 5;
}
```

→ these are attributes of class Student.

Every time you create an instance/object, that instance will have these attributes

ex. object student1 will have a name ("Ali") and age (25).

→ Accessing attributes:

```
public class SchoolRecords {
    public static void main (String[] args) {
        Student std1 = new Student();
        std1.name = "Ali Zain Ali";
        std1.age = 25;
    }
}

public class Student {
    String name;
    int age = 5;
}
```

← This is how you can access the attributes for a particular student (object).

Note: This shows how Student class is a blueprint. We create std1 object or instance from class Student. Now std1 can access all the attributes from Student and assign name, age.

← note that you can either set a value for all objects that will be created from this class. OR you can not assign a value. You can directly assign a diff value for each object.

Methods : Methods usually perform certain actions.

Ex. Creating a method that calculates and prints age

```
public class SchoolRecords {
    pubric static void printAge (String dob) {
        //do som calculation
        System.out.println( //print age )
    }

    Public Static void main (String[] args) {
        printAge("Feb 16, 1998");
    }
}
```

← Method printAge prints my age.

Constructors:

→ A constructor is a special Java method that is used to initialize objects.
Constructor is called when the object is first created.
It is used to set initial values for object attributes.

Ex:

```
public class Student {
    int age;
    String name;

    public Student() {
        name: "Ali";
        age = 25;
    }

    public Student (String x, int y)
        name = x;
        age = y;
    }

    public Static void main() {
        Student s1 = new Student();
        Student s2 = new Student("Nick", 17);
    }
}
```

← Using this constructor, we can initialize all the Student objects we create to name: Ali and age=25

← using this constructor, we can initialize name and age with x and y parameter which will be provided when the object is created.

← for s1, the name and age will be Ali, 25.

← initializing using second constructor. Providing name and age during execution

Methods (with + without parameters)

You can have methods with and without parameter.

Ex:
```
static void printx () {
        System.out.println("x");
}

static void printText (String txt ) {
        System.out.println(txt);
}

public static void Main(String args[]) {
        printx();          ← Prints x   ← example of calling method without parameter
        printText ("Ali")  ← Prints Ali
}
```
↖ calling methods with parameter.

Summary

Ex:

Student Class

```
Public class Student {
    String name;
    int age;
    String phone;
    String email;

    Public setAge (int a) {
        age = a;
    }

    public setPhone (String ph ) {
        Phone: ph;
    }
}
```

attributes

methods

SchoolRecords class.

```
Public class SchoolRecords {

    Pubric Static void main(String[] args) {
        Student std = new Student();
        std.setAge (55) ;
        std.setPhone ("7738185015");
        System.out.println(std.age);
    }
}
```