Enum:

Enumerations serve the purpose of representing a group of named constants in a programming language. Enums are used when we know all possible values at **compile time**, such as choices on a menu, rounding modes, command line flags, etc.

In Java, we can also add variables, methods and constructors to it. The main objective of enum is to define our own data types (Enumerated Data Types).

Declaration of enum in java:

- Enum declaration can be done outside a Class or inside a Class but not inside a Method.
- First line inside enum should be list of constants and then other things like methods, variables and constructor.
- According to Java naming conventions, it is recommended that we name constant with all capital letters

Important points of enum:

- Every enum internally implemented by using Class.

```
/* internally above enum Color is converted to

class Color

{

   public static final Color RED = new Color();

   public static final Color BLUE = new Color();

   public static final Color GREEN = new Color();

}*/
```

- Every enum constant represents an **object** of type enum.
- enum type can be passed as an argument to **switch** statement.
- Every enum constant is always implicitly **public static final**. Since it is **static**, we can access it by using enum Name. Since it is **final**, we can’t create child enums.
- We can declare **main() method** inside enum. Hence we can invoke enum directly from the Command Prompt.

Enum and Inheritance:

- All enums implicitly extend **java.lang.Enum class**. As a class can only extend **one** parent in Java, so an enum cannot extend anything else.
- **toString() method** is overridden in **java.lang.Enum class**,which returns enum constant name.
- enum can implement many interfaces.

enum and constructor :

- enum can contain constructor and it is executed separately for each enum constant at the time of enum class loading.
- We can’t create enum objects explicitly and hence we can’t invoke enum constructor directly.