JAVA.LANG.PACKAGE

OBJECT CLASS

- Object class is the parent class for all the classes in java.
- It can also be said as mother of all classes in the java.
- Even the user-defined classes are inherited from the object class.
- There are important methods in java:
  - **clone( ):** creates a clone of itself and give the same object.
  - **equals(object obj):** it will compare two objects, and will return true if both the references are holding the same objects.
  - **finalize( ):** this method is called by a garbage collector when ever a object of a class is being taken away by the garbage collector.

    It is same as destructor method in c++.
  - **Class( ):** returns the runtime class of the object.
  - **hashCode( ):** returns the hash code value of the object, there are no two programs created by java program which have same hashcodes.
  - **notify( ):** wakes up a single thread.
  - **notifyAll( ):** wakesup all the threads.
  - **toString( ):** returns the string representation of the object, when we print any object it calls this method by itself, in any class if you want the object to be printed by s.o.p then over ride toString( ) method.
  - **wait( ):** causes the current thread to wait until the thread invokes the notify.
  - **wait(long timeout):** causes the current thread to wait until either thread invokes the notify.
  - **wait(long timeout, int nanos):** causes the current thread to wait until the thread invokes the notify certain amount of real time has elapsed.

WRAPPER CLASSES

- Java provides wrapper classes around primitives so they can be used as classes and their objects can be created.
- Wrapper classes are available for every data type.
- Wrapping is also known as boxing.
- All these classes are present inside the java.lang package.
- Number, Character and Boolean classes are child classes of object class.
- **Number class** have methods like:
    - byteValue( ): returns the value as byte.
    - doubleValue( ): returns the value as double.
    - floatValue( ): returns the value as float.
    - intValue( ): returns the value as int.
    - longValue( ): returns the value as long.
    - shortValue( ): returns the value as short.
- **Integer class:**
    - It is the child class of number.
    - The Integer class wraps a value of the primitive type int in an object. An object of type Integer contains a single field whose type is int.
    - It inherits the methods from the number class as it is the parent class.
- **Byte Class:**
    - The Byte class wraps a value of primitive type byte in an object. An object of type Byte contains a single field whose type is byte.
- **Long Class:**

- The Long class wraps a value of the primitive type long in an object. An object of type Long contains a single field whose type is long.

➢ **Short Class:**

- The Short class wraps a value of primitive type short in an object. An object of type Short contains a single field whose type is short.

➢ **Float Class:**

- The Float class wraps a value of primitive type float in an object. An object of type Float contains a single field whose type is float.

➢ **Double Class:**

- The Float class wraps a value of primitive type float in an object. An object of type Float contains a single field whose type is float.

➢ **Character Class:**

- The Character class wraps a value of the primitive type char in an object. An object of class Character contains a single field whose type is char.

➢ **Boolean Class:**

- The Boolean class wraps a value of the primitive type boolean in an object. An object of type Boolean contains a single field whose type is boolean.

➢ **AutoBoxing and AutoUnBoxing:**

- *Autoboxing* is the automatic conversion that the Java compiler makes between the primitive types and their corresponding object wrapper classes. For example, converting an int to an Integer, a double to a Double, and so on. If the conversion goes the other way, this is called *unboxing*.

STRING v/s STRING BUFFER v/s STRING BUILDER

- Java provides other classes for handling strings those are string buffer and builder.
- String holds the collection of characters.
- The string class objects are immutable.
- String class have lots of methods to perform different operations.
- String buffer is similar to string but it is mutable
    - It does not return a new string for the object.
    - More characters can be added to the end or the characters can be appended.
    - Any string or substring can be inserted at the particular index.
    - The above methods are applicable as the string is mutable.
    - Length of a string inside the string buffer may be different or less than the capacity.
    - The capacity of string buffer by default is 16.
    - The capacity may increase or decrease by itself or by the String buffer depending upon the string we store.
    - More than two threads cannot occupy string buffer Simultaneously which means than the string buffer have theThread safe.
    - As the two threads ca not occupy the methods it means that the methods of the string buffer are synchronized.
- String builder is same as the string buffer but is not thread-safe.
    - String builder is faster than the string buffer.
    - instead of the string buffer thee string builder can be used if only single thread is used.

MATH CLASS

- The math class is for controlling the mathematical operations.
- All the mathematical operations are provided in the static method.
- All the methods can be called by using only the class name.
- There are different methods for math class:
  - Math.abs( ):

    It gives the positive value of the given value.
  - StrictMat.abs( ):

    It gives the perfect result and is indirectly used by math class.
  - Math.cbrt( ):

    It gives the root value of the given value precisely.
  - Math.decrementExact( ):

    It gives the exact decremented value of the given number but it does not overflow to the max_value while decrementing.
  - Math.getExponent( ):

    If the float or double value is given then the above method stores the data in mantissa and exponent form and gives the exponent as the result.
  - Math.floorDiv( ):

    It gives the truncated result of the value given that is by removing the decimal and giving the approximate value.
  - Math.exp( ):

    It gives the function of e^x in return where x is the argument and the e is the euler's number.
  - StrictMath.exp( ):

    It returns the value of e raised to the power of double value.

➔ Math.log( ):

It returns the natural log as a double value of a parameter.

➔ Math.tan( ):

It returns the trigonometric function.

It takes only the radian value.

It returns the precise value.

➔ Math.toRadians( ):

Returns the value to convert the degree to the radians.

➔ Math.atan( ):

Returns the value in radians of the arc-Tangent of a number.

➔ Math.random( ):

It returns a random number between 0 and 1.

It returns a float number.

It returns a pseudorandom double type number.

➔ Math.pow( ):

Returns the value of first argument raised to the power of second argument.

➔ Math.multiplyExact( ):

It is a built-in math function in java which returns the exact product of the arguments.

The result may go beyond the given integer value.

So, it may give overflow results.

➔ Math.nextAfter( ):

Returns the floating point of a number adjacent to the first argument in the direction of the second argument.

If both the arguments are equal then the value of second argument is returned.

```
enum Dept {
    // constant must be capital
    ECE("Dr. Smith", "Building A"),
    CSE("Dr. Jones", "Building B"),
    MECH("Dr. Doe", "Building C");

    // Variables to hold the data
    private String headName;
    private String location;

    Dept(String headName, String location) {
        this.headName = headName;
        this.location = location;
    }

    public String getHeadName() {
        return headName;
    }

    public String getLocation() {
        return location;
    }
}
```

In Java, every Enum constant you define is implicitly public, static, and final.

ENUM

- It is used to define our own data types or to define an enumerated data type.
- We can have pre-defined Identifier using enum in java.
- Enums are defined just like classes.
- There can be final and static members in the ENUM
- Enum will be directly inherited by the enum object in the lang package.
- It can also have other methods as well as the constructors.

  Where constructor cannot be public or protected.
  . The constructor is called only by the enum constants
- All the identifiers given are created when the enum class is loaded.

You can use all four access modifiers (private, public, protected, default) for variables and methoc inside an enum, just like a regular class.

If you define static variables, they must be declared after the list of Enum constants. If you try to put them before the constants, you will get a compile-time error

REFLECTION PACKAGE

- Java have a package called java.lang.reflect which have different set of classes.
- The set of classes that are there in reflection package will help us get the information or the definition about the class.