**PACKAGES:**

* Packages are the collection of similar types of classes and interfaces.
* Package name must be specified in lowercases and in single word.
* We can import any static variable or method using static import.
* Import statement will import all the classes but not the sub-classes
* printstacktrace() will show you the error exactly where it occurs.

Access Specifiers:

* If I define any data member without any access specifier it will take it as default.
  + Private
  + Default
  + Protected
  + Default
* Private data member will be accessed only within the same class.
* Default data member will be accessed only within the same package.
* Protected data member will be accessed within the same package and the subclasses(inherited).
* Public data member can be accessed from anywhere.

**Inner class:**

* There are four types of inner class
  + Static inner class.
  + Non – static inner class.
  + Anonymous inner class.
  + Normal inner class.
* **Non – static inner class.**
  + Non static inner class should be called by creating the object for the normal class and using that object to initialize the object of inner class.
* **Static inner class:**
  + We have to create the object by using normal class name.innerstatic class name.
* **Local inner class:**
  + Local inner class is nothing but the class inside a method.
  + We can create an object directly to access the data members of that class.
* **Anonymous inner class:**
  + Anonymous interface needs at least one interface to execute.
  + We must create an object while declaring a class.
  + We can access the methods inside the class using the created object.
  + Only in the anonymous inner class will we create an object for the interface.

**Multithreading:**

* **Multi-processing:**
  + Multiple processors will work in the background.
* **Multi-Tasking:**
  + Running multiple tasks in a single processor.
* **Multi – threading:**
  + One task will be divided into multiple Sub – tasks. Each task is called thread. Running multiple threads simultaneously is knows as multi – threading.
* **Thread lifecycle:**
  + New 🡪 ready 🡪 runnable 🡪 Terminated state
  + To start the thread we can use start() and to kill them we can use stop().
  + While running there are chances to a thread that may go to waiting state.
  + As programmers we can able to move the thread to sleep state for certain period of time.
  + And we can also move the thread to suspended state using suspend() and we can resume them using resume ().
  + If I move a thread to wait state, I must use a method notify() or notifyall() to get back to its execution.
  + If a method is in waiting state and I must get it back to the ready state we can use yield().
  + Thread is predefined class from java.lang package.
  + Wait (), notify (), notifyall () are the predefined method from object class.
  + There are two ways to create a own thread.
    - By extending the Thread class
    - By implementing runnable interface.
  + In both ways we have to override the run ().
  + There are certain methods to access thread.
  + getPriority(), setpriority(), getname(), setname(), currentthread(), getstate(), join().
  + These are the extra method from multi-threading.
  + The thread priority can be between one to ten.
  + Priority(1) is the minimum priority.
  + Priority(5) is the normal priority.
  + priority(10) is the maximum priority.
  + Thread class have three final static priority
    - Thread.MIN\_PRIORITY 1
    - Thread.NORM\_PRIORITY 5
    - Thread.MAX\_PRIORITY 10

Runnable interface:

* Functional interfaces are nothing, but the interface has only one abstract class.
* Runnable interface has only one method inside that so runnable interface is an example of functional interface.
* It is good to go with implementing runnable interface other than extending Thread class because in future if we want to extend any other class that is possible only using implanting runnable interface.