- ***import java.util.Scanner***
  - **import** – Statement
  - **java** - package, (A mechanism to control class name collision and visibility)
  - **util** - subpackage, (contains many class that has various functionalities that use can utilize)
  - **Scanner** – Class that contain streams to read input
- ***Access specifiers*** – It allows programmer to control the visibility of the member (class, variable, method)
  - **Public** – The member can be used in same class or other class within same project or other project
  - **Private** – The member can only be used in same class.
  - **Protected** – The member can be used by class which has inherited
  - **Default** – The member can be used by class within same package
- ***public static void main(String args[])***
  - **public** – Access Specifier
  - **static** - In any code, the execution starts from the main() method. As java is a 'Object Oriented', a method can only be invoked by 'Object' of the class. It is necessary that java interpreter calls main() method before objects are created, hence in order to invoke main() without an object, we use the keyword 'static'
  - **void** - return type of the method, which doesn't return any values
  - **main()** – Method where the program starts, Java Interpreter searches for the main() method to start execution
  - **String args[]** – It's a string type array, which passes the command line arguments during execution.
    - C:\Users\User_Name\Java>javac filename.java
    - C:\Users\User_Name\Java>java filename Strings given in command line are passed using args[], when we print String args[], then we get those strings that were entered in command line
- ***Scanner Obj = new Scanner(System.in);***
  - **Scanner** – class that allows to read user inputs
  - **Obj** - Identifier for object
  - **new** - keyword to create java object and allocate memory on heap.
  - **System.in** – Input stream connected to keyboard input.
  - **Scanner Obj = new Scanner(System.in);** will invoke Constructor of Scanner class with argument System.in, and will return a reference to newly constructed object Obj.
- ***While loop***
  - While loop executes the statements as long as the Boolean condition is true.
  - while(count<5) { …statements; } // to print count value
- ***do-while***
  - do-while loop executes the statements as long as the Boolean condition is true.
  - While loop checks the condition and then execute the block, hence in some cases the block won't be executed. In the program sometimes it's necessary to execute the block at least once before checking the condition. do-while works in similar to while loop, but it executes one time before checking the condition.
  - Applications in command prompt, online shopping, ATM enter pin, animation.
- ***For loop***
  - For loop executes the block for given number of iterations.
  - For loop can be used for entering array elements, printing array, and pyramid.
- ***User defined function***
  - Function is a part of program which is fragments of logic block to reduce complexity.
  - **User-Defined Function**
    - The function or method which is defined by user is called as User-Defined Functions. User defined method is usually written below main method of Code.
    - printArray(), calculateInterest().
- ***Parameters***
  - Function Parameters is a variable that are passed as arguments to the method either by "pass by value" or by "pass by reference".

- ***Different way of passing function parameters***
  - **Pass by value** – It holds the actual data
  - **Pass by reference** – When a reference type variable is declared, then compiler **allocates** only space where the memory address of the object can be stored. The space for the object is not allocated until the time of object creation using new keyword. It holds the address of object which it refers to and not the actual data.
- ***Java Package***
  - Java Package is used to prevent naming conflict to control access to make searching locating and usage of classes, interfaces, enumeration, annotations... easily
    - Eg. Java.util. Java.math., Java.lang.
- ***Inheritance***
  - Inheritance in java is a mechanism in which one object acquires all the properties and behaviours of parent object. The idea behind inheritance in java is that you can create new classes that are built upon existing classes. It provides way for code reusability.
  - There are 3 level of inheritances: Single level inherence, multi-level inherence, multiple level inherence.
- ***Encapsulation***
  - Encapsulation is the process of wrapping code and data together into a single unit. We can create fully encapsulated class in java by using all the data members of the class as private.
  - It makes class read-only, write-only
  - It provides control over data
- ***Abstraction***
  - The process of hiding the implementation details of the object and show only the outer functionality.
  - Eg. Composing mail without being involved in background process.
- ***Polymorphism***
  - Polymorphism is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.
  - Any Java object that can pass more than one IS-A test is considered to be polymorphic
  - Polymorphism is the capability of a method to do different things based on the object that it is acting upon. In other words, polymorphism allows you define one interface and have multiple implementations.
    - **Method Overloading:** In Java, it is possible to define two or more methods of same name in a class, provided that there argument list or parameters are different. This concept is known as Method Overloading.
    - **Method Overriding**: Child class has the same method as of base class. In such cases child class overrides the parent class method without even touching the source code of the base class. This feature is known as method overriding.
- ***Data type casting***
  - Assigning a value of one type to a value of another type. It is the conversion of data from one data type to another.
  - Implicit Casting (byte>short>int>long>float>double)
  - Explicit Casting (double>float>long>int>short>byte)
- ***Nested looping***
  - A loop inside another loop is called as Nested looping.
- ***Boxing and Unboxing***
  - In Java programming language, there are eight primitive types and each of these has a corresponding library class of reference type.
  - For example, there is a class java.lang.Integer that corresponds to primitive type int. These kinds of classes are called wrappers.
  - The wrapper classes are used whenever a primitive type needs to be treated as an Object.
  - Conversion of a primitive type to the corresponding reference type is called boxing, such as an int to a java.lang.Integer.
  - Conversion of the reference type to the corresponding primitive type is called unboxing, such as Byte to byte.

- ***String Manipulation***
  - tocharArray(), toString()
  - length()
  - concat()
  - charAt(), indexOf(), lastIndexOf()
  - compareTo(), compareToIgnoreCase()
  - contentEquals()
  - startsWith(), endsWith()
  - replace(), replaceAll()
  - split()
  - substring(), subSequence()
  - toUpperCase(), toLowerCase(), toCharArray()
  - equals(), equalsIgnoreCase()
- ***String Tokenizer***
  - The **java.util.StringTokenizer** class allows you to break a string into tokens. It is simple way to break string.

```
StringTokenizer st = new StringTokenizer ("string tokenizer is used to break strings"," ");

while (st.hasMoreTokens())

{

System.out.println(st.nextToken());
```