Object Oriented Programming (OOPs) :—

→ OOPs refers to languages that uses objects in programming. Object-oriented programming aims to implement real world entities like inheritance, hiding polymorphism etc in programming.

→ Simula is considered the first object oriented programming language.
The programming paradigm where ~~everyone~~ everything is represented as an object is known as a truly object oriented programming language.
Smalltalk is considered as the first truly object oriented programming language.

→ Procedural programming is about writing procedures or methods that perform operations on the data, while object oriented programming is about creating objects that contain both data and methods.
Object oriented programming has several advantages over procedural programming:
- OOP is faster and easier to execute.
- OOP provides a clear structure for the programs.
- OOP helps to keep the Java code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug.
- OOP makes it possible to create full reusable applications with less code and shorter development time.

Access Modifier :

→ Defines access type of the method i.e from where it can be accessed in your application. In Java, there 4 types of the access specifiers.

For Classes :

Public - The class is accessible by any other class. (accessible in all class in your application)

default - The class is only accessible by classes in the same package. This is used when you don't specify a modifier. [illegible]

For attributes, methods and constructors :

Public : The code is accessible for all classes

Private - The code is only accessible within the declared class.

default - The code is only accessible in the same package. This is used when you don't specify a modifier. [illegible]

protected - The code is accessible in the same package and subclasses.

Non-Access Modifier :

For classes :

final - The class can't be inherited by other classes

abstract - The class can't be used to create objects.

for attributes and methods :

final - Attributes and methods can't be overridden/ modified

static - Attributes and methods belongs to the class rather than an object.

abstract - can only be used in an abstract class, and can only be used on methods. The method does not have a body, for example abstract void run( ); The body is provided by the subclass.

transient - Attributes and methods are skipped when serializing the object containing them.

synchronized - Methods can only be accessed by one thread at a time

volatile - The value of an attribute is not cached thread-locally, and is always read from the main memory.

The return type

→ The data type of the value returned by the method or void if does not return a value.

Method Name

→ The rules for field name apply to method names as well, but the convention is a little different.

Parameter List

→ Comma separated list of the input parameters are defined, preceded with their data type, within the enclosed parenthesis. If there is no parameter, we must use empty parenthesis ( ).

Exception List

→ The exceptions you excect by the method can throw you can specify these exception.

Method body

→ It is enclosed between braces. The code you need to be executed to platform your intended operations.

CLASS :

→ A class is a user defined blueprint or ~~a~~ prototype from which objects are created. It represents the set of propotties or methods that are common to all objects of one type.

→ It is logical entity. It can't be physical.

OBJECT :

→ An entity that has state and behavior is known as an object, It is an instance of class.

→ An object has 3 characteristics.

state - represents the data of an object.

Behavior - represents the behaviour of an object (such as deposit, withdraw etc) effect of data type operation

Identity - An object identity is typically implemented via a unique id. The value of the ID is not visible to external user. It is used internally by the JVM to identify each object uniquely.

→ The state of an object is a value from its data type. The identity of an object distinguishes one object from another. It is useful to think of an object's identity as ~~a~~ the place where it's value is stored in memory.

→ The dot(.) operator links the name of the object with the name of an instance variable.

Constructor :

→ In Java, a constructor is a block of codes similar to ~~be~~ the method.

→ It is called when an instance of the class is created. At the timing of calling constructor, memory for the object is allocated in the memory.

→ It is a special type of method which is used to initialize the object.

Method :

→ A method is a block of code on collection of statements on a set of code grouped together to perform a certain task on operation.
→ It is used to achieve the reusability of code.
→ It also provides the easy modification and readability of code, just by adding or removing a chunk of code.

*How constructors are different from methods of Java :—
→ Constructors must have the same name as the class within which it is defined while it is not necessary for the method in Java.
→ Constructors don't have any return type while method have the return type or void if doesn't return any value.
→ Constructors are called only once at the time of object creation while method can be called any number of time.

*Certain rules for writing Constructor :—
→ Constructors of a class must have the same name as the class name in which it resides.
→ A constructors in Java can't be abstract, final, static and synchronized.
→ Access modifiers can be used in constructors declaration to control it's access i.e which other class can call the constructor.

* Java's primitive types are not implemented as objects. They are implemented as normal variable.

this Keyword :

→ This is a reference variable that refers to the current object.

Usage of Java 'this' keyword :-

→ used to refer current class instance variable

→ used to invoke current class method (implicity)

→ this() can be used to invoke current class constructor

→ passed as an argument in the method call

→ passed as argument in the constructor call

→ used to return the current class instance from the method

final Keyword :

→ The final keyword in java is used to restrict the user. The java final keyword can be used in many context.

Final can be -

(a) Variable

(b) method

(c) class

→ The final keyword can be applied with the variables, a final variable have no value. It is called blank final variable or uninitialized final variable.

→ It can be initialized in the constructor only. The blank final variable can be static also which will be initialized in the static block only.

Garbage Collection :

→ It is a process of reclaiming the runtime unused memory automatically. It is a way to destroy the unused objects.

→ It makes memory efficient because garbage collector removes the unreferenced objects from heap memory.

→ It is automatically done by the garbage collector (a part of JVM) so we don't need to make extra efforts.