Structure of Java

Classes in Java :

Class is a collection of objects of similar type. Objects are variables of the type class. Once a class has been defined, we can create any number of objects belonging to that class. Eg: grapes, bananas and orange are the member of class fruit.

Example:

Fruit orange;

In the above statement object mango is created which belong to the class fruit.

Constructors:

A constructor in Java is a special method that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes. A Constructor has the same name as the Class.

Methods:

A method is a block of code which only runs when it is called.

You can pass data, known as parameters, into a method.

Methods are used to perform certain actions, and they are also known as functions.

Use of methods is to reuse code: define the code once, and use it many times.

Access Specifier:

Java Access Specifiers (also known as Visibility Specifiers) regulate access to classes, fields and methods in Java.These Specifiers determine whether a field or method in a class, can be used or invoked by another method in another class or subclass. Access Specifiers can be used to restrict access. Access Specifiers are an integral part of object-oriented programming.

There are four types of Java access modifiers:

1. **Private:** The access level of a private modifier is only within the class. It cannot be accessed from outside the class.
2. **Default:** The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.
3. **Protected:** The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.

4. **Public:** The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

Static Members: In Java, static members are those which belongs to the class and you can access these members without instantiating the class.

Comments:

In a program, comments take part in making the program become more human readable by placing the detail of code involved and proper use of comments makes maintenance easier and finding bugs easily. Comments are ignored by the compiler while compiling a code.

Variables :

A variable in simple terms is a storage place which has some memory allocated to it. Basically, a variable used to store some form of data. Different types of variables require different amounts of memory, and have some specific set of operations which can be applied on them.

Method Overloading:

Overloading allows different methods to have the same name, but different signatures where the signature can differ by the number of input parameters or type of input parameters or both. Overloading is related to compile-time (or static) polymorphism**.**

Method Overriding:

In any object-oriented programming language, Overriding is a feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its superclasses or parent classes. When a method in a subclass has the same name, same parameters or signature and return type(or sub-type) as a method in its super-class, then the method in the subclass is said to *override* the method in the super-class.

Packages : Package in Java is a mechanism to encapsulate a group of classes, sub packages and interfaces.

Wrapper Class:

A Wrapper class is a class whose object wraps or contains a primitive data types. When we create an object to a wrapper class, it contains a field and in this field, we can store a primitive data types. In other words, we can wrap a primitive value into a wrapper class object.

Autoboxing : Converting a primitive value into an object of the corresponding wrapper class is called Autoboxing.

Unboxing: Converting an object of a wrapper type to its corresponding primitive value is called unboxing.

Strings: Strings are defined as an array of characters. The difference between a character array and a string is the string is terminated with a special character '\0'.

StringBuffer: StringBuffer is a peer class of String that provides much of the functionality of strings. String represents fixed-length, immutable character sequences while StringBuffer represents growable and writable character sequences.

StringTokenizer : StringTokenizer class in Java is used to break a string into tokens.

Exception : An exception is an unwanted or unexpected event, which occurs during the execution of a program i.e at run time, that disrupts the normal flow of the program's instructions.

Error : An Error indicates serious problems that a reasonable application should not try to catch.

Inheritance : It is the process by which objects of one class acquire the properties or features of objects of another class. The concept of inheritance provides the idea of reusability means we can add additional features to an existing class without Modifying it

Encapsulation: Combining data and functions into a single unit called class and the process is known as **Encapsulation**.

Data Abstraction: The insulation of the data from direct access by the program is called Data Hiding or information hiding. Hiding the complexity of program is called **Data Abstraction** .

Polymorphism : A greek term means ability to take more than one form. An operation may exhibit different behaviours in different instances.

JVM : A Java virtual machine (JVM), an implementation of the Java Virtual Machine Specification, interprets compiled Java binary code (called bytecode) for a computer's processor (or "hardware platform") so that it can perform a Java program's instructions.

JRE: The Java Runtime Environment (JRE), also known as Java Runtime, is the part of the Java Development Kit (JDK) that contains and orchestrates the set of tools and minimum requirements for executing a Java application. The JDK, along with the Java Virtual Machine (JVM) and the JRE, can be used by developers to program and run Java applications.

JDK : The JDK is a software package that contains a variety of tools and utilities that make it possible to develop, package, monitor and deploy applications that build for any standard Java platform

Super Class(Parent Class): The Class that inherits from another class.

Subclass(Child Class): The Class being inherited from.

Dynamic Method Dispatch : Dynamic method dispatch is a mechanism by which a call to an overridden method is resolved at runtime. This is how java implements runtime polymorphism. When an overridden method is called by a reference, java determines which version of that method to execute based on the type of object it refers to. In simple words the type of object which it referred determines which version of overridden method will be called.

Interface : Like a class, an interface can have methods and variables, but the methods declared in an interface are by default abstract (only method signature, no body).

Streams : Streams are the sequence of bits(data).

Input Streams: Input streams are used to read the data from various input devices like keyboard, file, network, etc.

Output Streams: Output streams are used to write the data to various output devices like monitor, file, network, etc

Byte Stream: used to read or write byte data.

Character Stream: used to read or write character data.

Byte Input Stream: These are used to read byte data from various input devices. InputStream is an abstract class and it is the super class of all the input byte streams.

Byte Output Stream: These are used to write byte data to various output devices.Output Stream is an abstract class and it is the superclass for all the output byte streams.

Character Input Stream: These are used to read char data from various input devices. Reader is an abstract class and is the super class for all the character input streams.

Character Output Stream: These are used to write char data to various output devices.Writer is an abstract class and is the super class of all the character output streams.

FileWriter : FileWriter is meant for writing streams of characters.FileWriter is used to write to character files. Its write() methods allow you to write character(s) or strings to a file.

FileReader : FileReader class is used to read data from the file. It returns data in byte format like FileInputStream class.It is character-oriented class which is used for file handling in java.

BuffererReader : BufferedReader class is used to read the text from a character-based input stream. It can be used to read data line by line by readLine() method. It makes the performance fast.

BufferedWriter: BufferedWriter class is used to provide buffering for Writer instances. It makes the performance fast. It inherits Writer class. The buffering characters are used for providing the efficient writing of single arrays, characters, and strings.