**JAVA**

1. Platform Independent:
2. User writes the source code
3. **Javac** compiler of java takes the source code *(saves it as .class file)* and generates bytecode *(in .class file)*
4. **JVM**(Java Virtual Machine) executes the bytecode

Every OS might have different JVM, but the output of execution of bytecode by the JVM is same for every OS. This makes Java a Platform Independent language.

1. **Java Development Kit (JDK)** includes everything including compiler, Java Runtime Environment (JRE), java debuggers, java docs, etc.

For the program to execute in java, we need to install JDK on our computer in order to create, compile and run the java program.

1. **Java Runtime Environment (JRE):**JRE is a part of JDK.

JRE installation on our computers allows the java program to run, however, we cannot compile it.

JRE includes a browser, JVM, applet supports, and plugins. For running the java program, a computer needs JRE.



1. **OOPL (Object Oriented Programming Language)**: Java is an Object Oriented Programming Language.

**Note:** *However, it is not considered as pure object-oriented as it provides support for primitive data types (like int, char, etc).*

1. **Class:** The class is a blueprint (plan) of the instance of a class (object). It can be defined as a template that describes the data and behaviour associated with its instance.

*Example*: Blueprint of the house is class.

1. **Object**: The object is an instance of a class. It is an entity that has behaviour and state.

*Example*: A car is an object whose **states**are: brand, colour, and number plate.

**Behaviour:** Running on the road.

1. **Method**: The behaviour of an object is the method.

*Example*: The fuel indicator indicates the amount of fuel left in the car.

1. **Garbage Collector:** It takes unreferenced objects and destroys them. Thereby freeing up the memory space.
2. A class in java contains:
3. Member functions(methods)
4. Data members(properties)
5. **Java Keywords**

|  |  |
| --- | --- |
| **Key word** | **Meaning** |
| Static | Indicates that the class member can be accessible without instantiating the class |

1. Complexities

|  |  |
| --- | --- |
| **Notation** | **Meaning** |
| O(1) | Means that the algorithm takes the same number of steps no matter how much data is there |
| O(N) | It’s the “Big O” way of saying that for N elements inside an array, the algorithm would take N steps to complete |
| O(log N) | * It’s the Big O way of describing an algorithm that ***increases one step each time the data is doubled*** * P.S.: Shorthand for saying O (log2 N). * O (log N) means that the algorithm takes as many steps as it takes to keep halving the data elements until we remain with one. |

1. Java String Format Specifiers

Here, we are providing a table of format specifiers supported by the Java String.

|  |  |  |
| --- | --- | --- |
| **Format Specifier** | **Data Type** | **Output** |
| %a | Floating point (except *BigDecimal*) | Returns Hex output of floating point number. |
| %b | Any type | "true" if non-null, "false" if null |
| %c | character | Unicode character |
| %d | integer (incl. byte, short, int, long, bigint) | Decimal Integer |
| %e | floating point | decimal number in scientific notation |
| %f | floating point | decimal number |
| %g | floating point | decimal number, possibly in scientific notation depending on the precision and value. |
| %h | any type | Hex String of value from hashCode() method. |
| %n | none | Platform-specific line separator. |
| %o | integer (incl. byte, short, int, long, bigint) | Octal number |
| %s | any type | String value |
| %t | Date/Time (incl. long, Calendar, Date and TemporalAccessor) | %t is the prefix for Date/Time conversions. More formatting flags are needed after this. See Date/Time conversion below. |
| %x | integer (incl. byte, short, int, long, bigint) | Hex string. |