1. **Abstraction**: abstraction is a process of hiding the implementation details from the user, only the functionality will be provided to the user
2. In Java, abstraction is achieved using Abstract classes and interfaces.
3. What is abstract class in java :

* A class which is defined with the abstract keyword

Public abstract class MyClass { }

* We can’t create the object of an abstract class
* There can be abstract or non-abstract methods in a class
* To use the abstract class you have to extend it and define the all abstract methods in it
* With the Object of any extending class you can call its methods.

1. true, false, and null might seem like keywords, but they are actually literals; you cannot use them as identifiers in your programs.
2. List of keywords in java

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| abstract | continue | for | new | switch |
| assert\*\*\* | default | goto\* | package | synchronized |
| boolean | do | if | private | this |
| break | double | implements | protected | throw |
| byte | else | import | public | throws |
| case | enum\*\*\*\* | instanceof | return | transient |
| catch | extends | int | short | try |
| char | final | interface | static | void |
| class | finally | long | strictfp\*\* | volatile |
| const\* | float | native | super | while |

1. We can give parameter args[] to the main() method or choose not to. But if we would call any other parameterized method without passing enough arguments, it would give us an error.

Why it is not the case with the main(String[] args) method?

1. Object based programming languages follow all the features of OOPs except Inheritance. Examples of object based programming languages are JavaScript, VBScript etc.
2. an **instance variable** is a **variable** defined in a class (i.e. a member **variable**)
3. Constructor is just like a method that is used to initialize the state of an object. It is invoked at the time of object creation.
4. Does constructor return any value? Yes-object
5. Can constructor perform other tasks instead of initialization?

Yes, like object creation, starting a thread, calling method etc. You can perform any operation in the constructor as you perform in the method.

1. The static can be-

* variable (also known as class variable)
* method (also known as class method)
* block
* nested class
* The static variable gets memory only once in class area at the time of class loading.

**Restrictions for static method**

|  |
| --- |
| There are two main restrictions for the static method. They are: |

|  |
| --- |
| 1. The static method can not use non static data member or call non-static method directly. 2. this and super cannot be used in static context. |

Java static block

* Is used to initialize the static data member.
* It is executed before main method at the time of classloading.
* **class** A2{
* **static**{System.out.println("static block is invoked");}
* **public** **static** **void** main(String args[]){
* System.out.println("Hello main");
* }
* }

Output:static block is invoked

Hello main

14 . Inheritance is a mechanism in which one object acquires all the properties and behaviour of another object of another class. It represents IS-A relationship. It is used for Code Resusability and Method Overriding