Core Java: Basic of Java

1) What is difference between JDK, JRE and JVM?

JVM
JVM is an acronym for Java Virtual Machine, it is an abstract machine which provides the runtime environment in which java ***bytecode*** can be executed. It is a specification.
JVMs are available for many hardware and software platforms (so JVM is platform dependent).

JRE
JRE stands for Java Runtime Environment. It is the implementation of JVM.

JDK
JDK is an acronym for Java Development Kit. It physically exists. It contains JRE + development tools.

2) How many types of memory areas are allocated by JVM?

Many types:

1. Class(Method) Area
2. Heap
3. Stack
4. Program Counter Register
5. Native Method Stack

3) What is JIT compiler?

Just-In-Time (JIT) compiler: It is used to improve the performance. JIT compiles parts of the byte code that have similar functionality at the same time, and hence reduces the amount of time needed for compilation. Here the term "compiler" refers to a translator from the instruction set of a Java virtual machine (JVM) to the instruction set of a specific CPU.

4) What is platform?

A platform is basically the hardware or software environment in which a program runs. There are two types of platforms software-based and hardware-based. Java provides software-based platform.

5) What is the main difference between Java platform and other platforms?

The Java platform differs from most other platforms in the sense that it's a software-based platform that runs on top of other hardware-based platforms. It has two components:

1. Runtime Environment
2. API(Application Programming Interface)

6) What gives Java its 'write once and run anywhere' nature?

The bytecode. Java is compiled to be a byte code which is the intermediate language between source code and machine code. This byte code is not platform specific and hence can be fed to any platform.

7) What is classloader?

The classloader is a subsystem of JVM that is used to load classes and interfaces. There are many types of classloaders e.g. Bootstrap classloader, Extension classloader, System classloader, Plugin classloader etc.

8) Is Empty .java file name a valid source file name?

Yes, save your java file by .java only, compile it by **javac .java** and run by **java yourclassname** Lets take a simple example:

```
//save by .java only
class A{
public static void main(String args[]){
System.out.println("Hello java");
}
}
//compile by javac .java
//run by     java A
```

compile it by **javac .java**

run it by **java A**

9) Is delete, next, main, exit or null keyword in java?

No.

10) If I don't provide any arguments on the command line, then the String array of Main method will be empty or null?

It is empty. But not null.

11) What if I write static public void instead of public static void?

Program compiles and runs properly.

12) What is the default value of the local variables?

The local variables are not initialized to any default value, neither primitives nor object references.

Core Java: OOP Concepts

13) What is difference between object oriented programming language and object based programming language?

Object based programming languages follow all the features of OOPs except Inheritance. Examples of object based programming languages are JavaScript, VBScript etc.

14) What will be the initial value of an object reference which is defined as an instance variable?

The object references are all initialized to null in Java.

15) What is constructor?

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.

16) What is the purpose of default constructor?

The default constructor provides the default values to the objects. The java compiler creates a default constructor only if there is no constructor in the class.

17) Does constructor return any value?

Yes, that is current instance (You cannot use return type yet it returns a value).

18) Is constructor inherited?

No, constructor is not inherited.

19) Can you make a constructor final?

No, constructor can't be final.

20) What is static variable?

Static variable is used to refer the common property of all objects (that is not unique for each object) e.g. company name of employees, college name of students etc.

Static variable gets memory only once in class area at the time of class loading.

21) What is static method?

- A static method belongs to the class rather than object of a class.
- A static method can be invoked without the need for creating an instance of a class.
- static method can access static data member and can change the value of it.

22) Why main method is static?

Because object is not required to call static method if It were non-static method, jvm creates object first then call main() method that will lead to the problem of extra memory allocation.

23) What is static block?

- Is used to initialize the static data member.
- It is executed before main method at the time of classloading.

24) Can we execute a program without main() method?

Yes, one of the ways is static block.

25) What if the static modifier is removed from the signature of the main method?

Program compiles. But at runtime throws an error "NoSuchMethodError".

26) What is difference between static (class) method and instance method?

static or class method	instance method
1) A method i.e. declared as static is known as static method.	A method i.e. not declared as static is known as instance method.
2) Object is not required to call static method.	Object is required to call instance methods.
3) Non-static (instance) members cannot be accessed in static context (static method, static block and static nested class) directly.	static and non-static variables both can be accessed in instance methods.
4) For example: public static int cube(int n){ return n*n*n;}	For example: public void msg(){...}.

27) What is this in java?

It is a keyword that that refers to the current object.

28) What is Inheritance?

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

29) Which class is the superclass for every class?

Object class.

30) Why multiple inheritances are not supported in java?

- To reduce the complexity and simplify the language, multiple inheritance is not supported in java in case of class.

31) What is composition?

Holding the reference of the other class within some other class is known as composition.

32) What is difference between aggregation and composition?

Aggregation represents weak relationship whereas composition represents strong relationship. For example: bike has an indicator (aggregation) but bike has an engine (composition).

33) Why Java does not support pointers?

Pointer is a variable that refers to the memory address. They are not used in java because they are unsafe(unsecured) and complex to understand.

34) What is super in java?

It is a keyword that refers to the immediate parent class object.

35) Can you use this() and super() both in a constructor?

No. Because super() or this() must be the first statement.

36) What is object cloning?

The object cloning is used to create the exact copy of an object.

37) What is method overloading?

If a class has multiple methods by same name but different parameters, it is known as Method Overloading. It increases the readability of the program.

38) Why method overloading is not possible by changing the return type in java?

Because of ambiguity.

39) Can we overload main() method?

Yes, You can have many main() methods in a class by overloading the main method.

40) What is method overriding?

If a subclass provides a specific implementation of a method that is already provided by its parent class, it is known as Method Overriding. It is used for runtime polymorphism and to provide the specific implementation of the method.

41) Can we override static method?

No, you can't override the static method because they are the part of class not object.

42) Why we cannot override static method?

It is because the static method is the part of class and it is bound with class whereas instance method is bound with object and static gets memory in class area and instance gets memory in heap.

43) Can we override the overloaded method?

Yes.

44) Difference between method Overloading and Overriding.

Method Overloading	Method Overriding
Method overloading increases the readability of the program.	Method overriding provides the specific implementation of the method that is already provided by its super class.
Method overloading is occurs within the class.	Method overriding occurs in two classes that have IS-A relationship.
In this case, parameter must be different.	In this case, parameter must be same.

45) Can you have virtual functions in Java?

Yes, all functions in Java are virtual by default.

46) What is covariant return type?

Now, since java5, it is possible to override any method by changing the return type if the return type of the subclass overriding method is subclass type. It is known as covariant return type.

47) What is final variable?

If you make any variable as final, you cannot change the value of final variable(It will be constant).

48) What is final method?

Final methods can't be overridden.

49) What is final class?

Final class can't be inherited.

50) What is blank final variable?

A final variable, not initialized at the time of declaration, is known as blank final variable.

51) Can we initialize blank final variable?

Yes, only in constructor if it is non-static. If it is static blank final variable, it can be initialized only in the static block.

52) Can you declare the main method as final?

Yes, such as, public static final void main(String[] args){}.

53) What is Runtime Polymorphism?

Runtime polymorphism or dynamic method dispatch is a process in which a call to an overridden method is resolved at runtime rather than at compile-time.

In this process, an overridden method is called through the reference variable of a super class. The determination of the method to be called is based on the object being referred to by the reference variable.

54) Can you achieve Runtime Polymorphism by data members?

No.

55) What is the difference between static binding and dynamic binding?

In case of static binding type of object is determined at compile time whereas in dynamic binding type of object is determined at runtime.

56) What is abstraction?

Abstraction is a process of hiding the implementation details and showing only functionality to the user.

Abstraction lets you focus on what the object does instead of how it does it.

57) What is the difference between abstraction and encapsulation?

Abstraction hides the implementation details whereas encapsulation wraps code and data into a single unit.

58) What is abstract class?

A class that is declared as abstract is known as abstract class. It needs to be extended and its method implemented. It cannot be instantiated.

59) Can there be any abstract method without abstract class?

No, if there is any abstract method in a class, that class must be abstract.

60) Can you use abstract and final both with a method?

No, because abstract method needs to be overridden whereas you can't override final method.

61) Is it possible to instantiate the abstract class?

No, abstract class can never be instantiated.

62) What is interface?

Interface is a blueprint of a class that has static constants and abstract methods. It can be used to achieve fully abstraction and multiple inheritances.

63) Can you declare an interface method static?

No, because methods of an interface are abstract by default, and static and abstract keywords can't be used together.

64) Can an Interface be final?

No, because its implementation is provided by another class.

65) What is marker interface?

An interface that has no data member and method is known as a marker interface. For example Serializable, Cloneable etc.

66) Can we define private and protected modifiers for variables in interfaces?

No, they are implicitly public.

67) What is difference between abstract class and interface?

Abstract class	Interface
An abstract class can have method body (non-abstract methods).	Interfaces have only abstract methods.
An abstract class can have instance variables.	An interface cannot have instance variables.
An abstract class can have constructor.	Interface cannot have constructor.
An abstract class can have static methods.	Interface cannot have static methods.
You can extend one abstract class.	You can implement multiple interfaces.

68) When can an object reference be cast to an interface reference?

An object reference can be cast to an interface reference when the object implements the referenced interface.

69) What is package?

A package is a group of similar type of classes interfaces and sub-packages. It provides access protection and removes naming collision.

70) Do I need to import java.lang package any time? Why?

No. It is by default loaded internally by the JVM.

71) Can I import same package/class twice? Will the JVM load the package twice at runtime?

One can import the same package or same class multiple times. Neither compiler nor JVM complains about it. But the JVM will internally load the class only once no matter how many times you import the same class.

72) What is static import?

By static import, we can access the static members of a class directly, there is no to qualify it with the class name.

73) What is Exception Handling?

Exception Handling is a mechanism to handle runtime errors.It is mainly used to handle checked exceptions.

74) What is difference between Checked Exception and Unchecked Exception?

Checked Exception

The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions e.g. IOException, SQLException etc. Checked exceptions are checked at compile-time.

Unchecked Exception

The classes that extend RuntimeException are known as unchecked exceptions e.g. ArithmeticException, NullPointerException etc. Unchecked exceptions are not checked at compile-time.

75) What is the base class for Error and Exception?

Throwable.

76) Is it necessary that each try block must be followed by a catch block?

It is not necessary that each try block must be followed by a catch block. It should be followed by either a catch block OR a finally block. And whatever exceptions are likely to be thrown should be declared in the throws clause of the method.

77) What is finally block?

finally block is a block that is always executed.

78) Can finally block be used without catch?

Yes, by try block. finally must be followed by either try or catch.

79) Is there any case when finally will not be executed?

finally block will not be executed if program exits(either by calling System.exit() or by causing a fatal error that causes the process to abort).

80) What is difference between throw and throws?

throw keyword	throws keyword
throw is used to explicitly throw an exception.	throws is used to declare an exception.
checked exceptions cannot be propagated with throw only.	checked exception can be propagated with throws.
throw is followed by an instance.	throws is followed by class.
throw is used within the method.	throws is used with the method signature.
You cannot throw multiple exception	You can declare multiple exception e.g. public void method()throws IOException, SQLException.

81) Can an exception be rethrown?

Yes.

82) Can subclass overriding method declare an exception if parent class method doesn't throw an exception?

Yes but only unchecked exception not checked.

83) What is exception propagation?

Forwarding the exception object to the invoking method is known as exception propagation.

84) What is the meaning of immutable in terms of String?

The simple meaning of immutable is modifiable or unchangeable. Once string object has been created, its value can't be changed.

85) Why string objects are immutable in java?

Because java uses the concept of string literal. Suppose there are 5 reference variables, all refers to one object "softech".If one reference variable changes the value of the object, it will be affected to all the reference variables. That is why string objects are immutable in java.

86) How many ways we can create the string object?

There are two ways to create the string object, by string literal and by new keyword.

87) How many objects will be created in the following code?

```
1. String s1="Welcome";
2. String s2="Welcome";
3. String s3="Welcome";
```

Only one object.

88) Why java uses the concept of string literal?

To make Java more memory efficient (because no new objects are created if it exists already in string constant pool).

89) How many objects will be created in the following code?

```
1. String s = new String("Welcome");
```

Two objects, one in string constant pool and other in non-pool (heap).

90) What is the basic difference between string and stringbuffer object?

String is an immutable object. StringBuffer is a mutable object.

91) What is the difference between StringBuffer and StringBuilder?

StringBuffer is synchronized whereas StringBuilder is not synchronized.

92) How can we create immutable class in java?

We can create immutable class as the String class by defining final class and

93) What is the purpose of toString() method in java ?

The toString() method returns the string representation of any object. If you print any object, java compiler internally invokes the toString() method on the object. So overriding the toString() method, returns the desired output, it can be the state of an object etc. depends on your implementation.

Core Java: Nested Classes and Interface

94) What is nested class?

A class which is declared inside another class is known as nested class. There are 4 types of nested class member inner class, local inner class, anonymous inner class and static nested class.

95) Is there any difference between nested classes and inner classes?

Yes, inner classes are non-static nested classes i.e. inner classes are the part of nested classes.

96) Can we access the non-final local variable, inside the local inner class?

No, local variable must be constant if you want to access it in local inner class.

97) What is nested interface?

Any interface i.e. declared inside the interface or class, is known as nested interface. It is static by default.

98) Can a class have an interface?

Yes, it is known as nested interface.

99) Can an Interface have a class?

Yes, they are static implicitly.