CORE JAVA Questions

core java interview questions for freshers experienced developers

1. What are static blocks and static initalizers in Java ?

Static blocks or static initializers are used to initalize static fields in java. we declare static blocks when we want to intialize static fields in our class. Static blocks gets executed exactly once when the class is loaded . Static blocks are executed even before the constructors are executed.

2. How to call one constructor from the other constructor ?

With in the same class if we want to call one constructor from other we use this() method. Based on the number of parameters we pass appropriate this() method is called.

Restrictions for using this method :

- this must be the first statement in the constructor
- we cannot use two this() methods in the constructor

3. What is method overriding in java ?

If we have methods with same signature (same name, same signature, same return type) in super class and subclass then we say subclass method is overridden by superclass.

When to use overriding in java

- If we want same method with different behaviour in superclass and subclass then we go for overriding.

- When we call overridden method with subclass reference subclass method is called hiding the superclass method.

4. What is super keyword in java ?

Variables and methods of super class can be overridden in subclass . In case of overriding , a subclass object call its own variables and methods. Subclass cannot access the variables and methods of superclass because the overridden variables or methods hides the methods and variables of super class.

But still java provides a way to access super class members even if its members are overridden. Super is used to access superclass variables, methods, constructors.

Super can be used in two forms :

- First form is for calling super class constructor.
- Second one is to call super class variables,methods.

Super if present must be the first statement.

5. Difference between method overloading and method overriding in java ?

No.	Method Overloading	Method Overriding
1)	Method overloading is used *to increase the readability* of the program.	Method overriding is used *to provide the specific implementation* of the method that is already provided by its super class.
2)	Method overloading is performed *within class*.	Method overriding occurs *in two classes* that have IS-A (inheritance) relationship.
3)	In case of method overloading, *parameter must be different*.	In case of method overriding, *parameter must be same*.
4)	Method overloading is the example of *compile time polymorphism*.	Method overriding is the example of *run time polymorphism*.
5)	In java, method overloading can't be performed by changing return type of the method only. *Return type can be same or different* in method overloading. But you must have to change the parameter.	*Return type must be same or covariant* in method overriding.

6. Difference between abstract class and interface ?

Abstract class	Interface
1) Abstract class can **have abstract and non-abstract** methods.	Interface can have **only abstract** methods. Since Java 8, it can have **default and static methods** also.
2) Abstract class **doesn't support multiple inheritance**.	Interface **supports multiple inheritance**.
3) Abstract class **can have final, non-final, static and non-static variables**.	Interface has **only static and final variables**.
4) Abstract class **can provide the implementation of interface**.	Interface **can't provide the implementation of abstract class**.
5) The **abstract keyword** is used to declare abstract class.	The **interface keyword** is used to declare interface.
6) An **abstract class** can extend another Java class and implement multiple Java interfaces.	An **interface** can extend another Java interface only.
7) An **abstract class** can be extended using keyword "extends".	An **interface** can be implemented using keyword "implements".
8) A Java **abstract class** can have class members like private, protected, etc.	Members of a Java interface are public by default.

7. Why java is platform independent?

The most unique feature of java is platform independent. In any programming language source code is compiled in to executable code . This cannot be run across all platforms. When javac compiles a java program it generates an executable file called .class file.

class file contains byte codes. Byte codes are interpreted only by JVM's . Since these JVM's are made available across all platforms by Sun Microsystems, we can execute this byte code in any platform. Byte code generated in windows environment can also be executed in linux environment. This makes java platform independent.

8. What is method overloading in java ?

A class having two or more methods with same name but with different arguments then we say that those methods are overloaded. Static polymorphism is achieved in java using method overloading. Method overloading is used when we want the methods to perform similar tasks but with different inputs or values.

When an overloaded method is invoked java first checks the method name, and the number of arguments ,type of arguments; based on this compiler executes this method.

Compiler decides which method to call at compile time. By using overloading static polymorphism or static binding can be achieved in java.

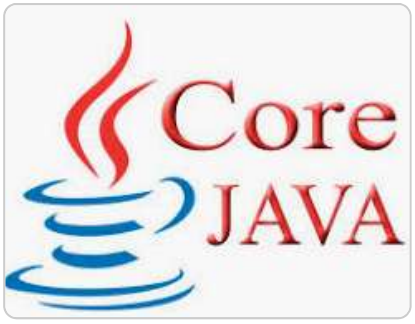
Note : Return type is not part of method signature. we may have methods with different return types but return type alone is not sufficient to call a method in java.

9. What is difference between c++ and Java ?

Java	C++
1) Java is platform independent	C++ is platform dependent.
2) There are no pointers in java	There are pointers in C++.
3) There is no operator overloading in java overloading.	C++ has operator
4) There is garbage collection in java collection	There is no garbage
5. Supports multithreading multithreading	Does'nt support
6) There are no templates in java C++.	There are templates in
7) There are no global variables in java in c++	There are global variables

10. What is JIT compiler ?

JIT compiler stands for Just in time compiler. JIT compiler compiles byte code in to executable code . JIT a part of JVM .JIT cannot convert complete java program in to executable code it converts as and when it is needed during execution.



CORE JAVA Interview Questions

11. What is bytecode in java ?

When a javac compiler compiles a class it generates .class file. This .class file contains set of instructions called byte code. Byte code is a machine independent language and contains set of instructions which are to be executed only by JVM. JVM can understand this byte codes.

12. Difference between this() and super() in java ?

this() is used to access one constructor from another with in the same class while super() is used to access

superclass constructor. Either this() or super(. Exists it must be the first statement in the constructor.

13. What is a class ?

Classes are fundamental or basic unit in Object Oriented Programming .A class is kind of blueprint or template for objects. Class defines variables, methods. A class tells what type of objects we are creating. For example take Department class tells us we can create department type objects. We can create any number of department objects.

All programming constructs in java reside in class. When JVM starts running it first looks for the class when we compile. Every Java application must have atleast one class and one main method.

Class starts with class keyword. A class definition must be saved in class file that has same as class name. File name must end with .java extension.

```
public class FirstClass {public static void main(String[] args)
{System.out.println("My First class");}

}
```

If we see the above class when we compile JVM loads the FirstClass and generates a .class file(FirstClass.class).

When we run the program we are running the class and then executes the main method.

14. What is an object ?

An Object is instance of class. A class defines type of object. Each object belongs to some class.Every object contains state and behavior. State is determined by value of attributes and behavior is called method. Objects are also called as an instance.

To instantiate the class we declare with the class type.

```
public classFirstClass {

public static voidmain(String[] args) {

FirstClass f=new FirstClass();

System.out.println("My First class");

}

}
```

To instantiate the FirstClass we use this statement

```
FirstClass f=new FirstClass();
```

f is used to refer FirstClass object.

15. What is method in java ?

It contains the executable body that can be applied to the specific object of the class.

Method includes method name, parameters or arguments and return type and a body of executable code.

Syntax : type methodName(Argument List){}

ex : public float add(int a, int b, int c)

methods can have multiple arguments. Separate with commas when we have multiple arguments.

16. What is encapsulation ?

The process of wrapping or putting up of data in to a single unit class and k eeps data safe from misuse is called encapsulation .

Through encapsulation we can hide and protect the data stored in java objects.Java supports encapsulation through access control. There are four access control modifiers in java public , private ,protected and default level.

For example take a car class , In car we have many parts which is not required for driver to know what all it consists inside. He is required to know only about how to start and stop the car. So we can expose what all are required and hide the rest by using encapsulation.

17. Why main() method is public, static and void in java ?

public : “public” is an access specifier which can be used outside the class. When main method is declared public it means it can be used outside class.

static : To call a method we require object. Sometimes it may be required to call a method without the help of object.

Then we declare that method as static. JVM calls the main() method without creating object by declaring keyword static.

void : void return type is used when a method does’nt return any value . main() method does’nt return any value, so main() is declared as void.

Signature : public static void main(String[] args) {

18. What about main() method in java ?

main() method is starting point of execution for all java applications.

public static void main(String[] args) {}

String args[] are array of string objects we need to pass from command line arguments.

Every Java application must have atleast one main method.

19)What is constructor in java ?

A constructor is a special method used to initialize objects in java.

we use constructors to initialize all variables in the class when an object is created. As and when an object is created it is initialized automatically with the help of constructor in java.

We have two types of constructors

No argument constructor

Parameterized Constructor

Signature : public classname()

{

}

Signature : public classname(parameters list)

{

}

20. What is difference between length and length() method in java ?

length() : In String class we have length() method which is used to return the number of characters in string.

Ex : String str = “Hello World”;

System.out.println(str.length());

Str.length(. Will return 11 characters including space.

length : we have length instance variable in arrays which will return the number of values or objects in array.

For example :

String days[]={” Sun”,”Mon”,”wed”,”thu”,”fri”,”sat”};

Will return 6 since the number of values in days array is 6.

21. What is ASCII Code?

ASCII stands for American Standard code for Information Interchange. ASCII character range is 0 to 255. We can’t add more characters to the ASCII Character set. ASCII character set supports only English. That is the reason, if we see C language we can write c language only in English we can’t write in other languages because it uses ASCII code.

22. What is Unicode ?

Unicode is a character set developed by Unicode Consortium. To support all languages in the world Java supports Unicode values. Unicode characters were represented by 16 bits and its character range is 0-65,535.

Java uses ASCII code for all input elements except for Strings,identifiers, and comments. If we want to use telugu we can use telugu characters for

identifiers.We can enter comments in telugu.

23. Difference between Character Constant and String Constant in java ?

Character constant is enclosed in single quotes. String constants are enclosed in double quotes. Character constants are single digit or character. String Constants are collection of characters.

Ex :'2', 'A'

Ex : "Hello World"

24. What are constants and how to create constants in java?

Constants are fixed values whose values cannot be changed during the execution of program. We create constants in java using final keyword.

Ex : final int number =10;

final String str="java-interview –questions"

25. Difference between '>>' and '>>>' operators in java?

>> is a right shift operator shifts all of the bits in a value to the right to a specified number of times.

int a =15;

a= a >> 3;

The above line of code moves 15 three characters right.

>>> is an unsigned shift operator used to shift right. The places which were vacated by shift are filled with zeroes.

Coding Standards Interview Questions

26. Explain Java Coding Standards for classes or Java coding conventions for classes?

Sun has created Java Coding standards or Java Coding Conventions . It is recommended highly to follow java coding standards.

Classnames should start with uppercase letter. Classnames names should be nouns. If Class name is of multiple words then the first letter of inner word must be capital letter.

Ex : Employee, EmployeeDetails, ArrayList, TreeSet, HashSet

27. Explain Java Coding standards for interfaces?

1) Interface should start with uppercase letters

2) Interfaces names should be adjectives

Example : Runnable, Serializable, Marker, Cloneable

28. Explain Java Coding standards for Methods?

1) Method names should start with small letters.

2) Method names are usually verbs

3) If method contains multiple words, every inner word should start with uppercase letter.

Ex : toString()

4) Method name must be combination of verb and noun

Ex : getCarName(),getCarNumber()

29. Explain Java Coding Standards for variables ?

1) Variable names should start with small letters.

2) Variable names should be nouns

3. Short meaningful names are recommended.

4) If there are multiple words every innerword should start with Uppecase character.

Ex : string,value,empName,empSalary

30. Explain Java Coding Standards for Constants?

Constants in java are created using static and final keywords.

1. Constants contains only uppercase letters.

2) If constant name is combination of two words it should be separated by underscore.

3. Constant names are usually nouns.

Ex:MAX_VALUE, MIN_VALUE, MAX_PRIORITY, MIN_PRIORITY

31. Difference between overriding and overloading in java?

Overriding Overloading

1. In overriding method names must be same 1. In overloading method names must be same

2. Argument List must be same 2. Argument list must be different atleast order of arguments.

3. Return type can be same or we can return covariant type. 3. Return type can be different in overloading.

4. We cant increase the level of checked exceptions. No 4. In overloading different exceptions can be thrown.

restrictions for unchecked exceptions

5. A method can only be overridden in subclass 5. A method can be overloaded in same class or subclass

6. Private,static and final variables cannot be overridden. 6. Private , static and final variables can be overloaded.

7. In overriding which method is called is decided at runtime 7. In overloading which method to call is decided at compile

based on the type of object referenced at run time time based on reference type.

8. Overriding is also known as Runtime polymorphism, 8. Overloading is also known as Compile time

dynamic polymorphism or late binding polymorphism, static polymorphism or early binding.

32. What is 'IS-A ' relationship in java?

'is a' relationship is also known as inheritance. We can implement 'is a' relationship or inheritance in java using

extends keyword. The advantage or inheritance or is a relationship is reusability of code instead of duplicating the code.

Ex : Motor cycle is a vehicle

Car is a vehicle Both car and motorcycle extends vehicle.

33. What is 'HAS A" relationship in java?

'Has a ' relationship is also known as "composition or Aggregation". As in inheritance we have 'extends' keyword we don't have any keyword to implement 'Has a' relationship in java. The main advantage of 'Has-A' relationship in java code reusability.

34. Difference between 'IS-A' and 'HAS-A' relationship in java?

1. IS-A relationship 1. HAS- A RELATIONSHIP

2. Is a relationship also known as inheritance 2. Has a relationship also known as composition or aggregation.

3. For IS-A relationship we uses extends keyword 3. For Has a relationship we use new keyword

Ex : Car is a vehicle. Ex : Car has an engine. We cannot say Car is an engine

4. The main advantage of inheritance is reusability of code 4. The main advantage of has a relationship is reusability of code.

35. What about instanceof operator in java?

Instanceof operator is used to test the object is of which type.

Syntax : instanceof

Instanceof returns true if reference expression is subtype of destination type.

Instanceof returns false if reference expression is null.

Example : public classInstanceOfExample {public static voidmain(String[] args) {Integer a = newInteger(5);if (a

instanceof java.lang.Integer) {

System.out.println(true);

} else {

System.out.println(false);

}

}

}

36. What does null mean in java?

When a reference variable doesn't point to any value it is assigned null.

Example : Employee employee;

In the above example employee object is not instantiate so it is pointed no where

37. Can we have multiple classes in single file ?

Yes we can have multiple classes in single file but people rarely do that and not recommended. We can have multiple

classes in File but only one class can be made public. If we try to make two classes in File public we get following

compilation error.

"The public type must be defined in its own file".

38. What all access modifiers are allowed for top class ?

For top level class only two access modifiers are allowed. public and default. If a class is declared as public it is visible everywhere.

If a class is declared default it is visible only in same package.

If we try to give private and protected as access modifier to class we get the below compilation error.

Illegal Modifier for the class only public,abstract and final are permitted.

39 . What are packages in java?

Package is a mechanism to group related classes ,interfaces and enums in to a single module.

Package can be declared using the following statement :

Syntax : package

Coding Convention : package name should be declared in small letters.

package statement defines the namespace.

The main use of package is

1) To resolve naming conflicts

2) For visibility control : We can define classes and interfaces that are not accessible outside the class.

40. Can we have more than one package statement in source file ?

We can't have more than one package statement in source file. In any java program there can be atmost only 1package statement. We will get compilation error if we have more than one package statement in source file.

41. Can we define package statement after import statement in java?

We can't define package statement after import statement in java. package statement must be the first statement in source file. We can have comments before the package statement.

42. What are identifiers in java?

Identifiers are names in java program. Identifiers can be class name, method name or variable name.

Rules for defining identifiers in java:

1) Identifiers must start with letter,Underscore or dollar($. Sign.

2) Identifiers can't start with numbers .

3) There is no limit on number of characters in identifier but not recommended to have not more than 15 characters

4) Java identifiers are case sensitive.

5) First letter can be alphabet ,or underscore and dollar sign. From second letter we can have numbers .

6. We should'nt use reserve words for identifiers in java.

43. What are access modifiers in java?

The important feature of encapsulation is access control. By preventing access control we can misuse of class, methods and members.

A class, method or variable can be accessed is determined by the access modifier. There are three types of access modifiers in java. public,private,protected. If no access modifier is specified then it has a default access.

44. What is the difference between access specifiers and access modifiers in java?

In C++ we have access specifiers as public,private,protected and default and access modifiers as static, final. But there is no such divison of access specifiers and access modifiers in java. In Java we have access modifiers and non access modifiers.

Access Modifiers : public, private, protected, default Non Access Modifiers : abstract, final, stricfp.

45. What access modifiers can be used for class ?

We can use only two access modifiers for class public and default.

public: A class with public modifier can be visible

1) In the same class

2) In the same package subclass

3) In the same package nonsubclass

4) In the different package subclass

5) In the different package non subclass.

default : A class with default modifier can be accesed

1) In the same class

2) In the same package subclass

3) In the same package nonsubclass

46. What access modifiers can be used for methods?

We can use all access modifiers public, private,protected and default for methods.

public : When a method is declared as public it can be accessed

6) In the same class

7) In the same package subclass

8) In the same package nonsubclass

9) In the different package subclass

10) In the different package non subclass.

default : When a method is declared as default, we can access that method in

1) In the same class

2) In the same package subclass

3) In the same package non subclass

We cannot access default access method in

1. Different package subclass

2. Different package non subclass.

protected : When a method is declared as protected it can be accessed

1. With in the same class

2. With in the same package subclass

3. With in the same package non subclass

4. With in different package subclass

It cannot be accessed non subclass in different package.

private : When a method is declared as private it can be accessed only in that class.

It cannot be accessed in

1. Same package subclass

2. Same package non subclass

3. Different package subclass

4. Different package non subclass.

47. What access modifiers can be used for variables?

We can use all access modifiers public, private,protected and default for variables.

public : When a variables is declared as public it can be accessed

1) In the same class

2) In the same package subclass

3) In the same package nonsubclass

4) In the different package subclass

5) In the different package non subclass.

default : When a variables is declared as default, we can access that method in

1) In the same class

2) In the same package subclass

3) In the same package non subclass

We cannot access default access variables in

4. Different package subclass

5. Different package non subclass.

protected : When a variables is declared as protected it can be accessed

1. With in the same class

2. With in the same package subclass

3. With in the same package non subclass

4. With in different package subclass

It cannot be accessed non subclass in different package.

private : When a variables is declared as private it can be accessed only in that class.

It cannot be accessed in

1. Same package subclass

2. Same package non subclass

3. Different package subclass

4. Different package non subclass.

48. What is final access modifier in java?

final access modifier can be used for class, method and variables. The main advantage of final access modifier is security no one can modify our classes, variables and methods. The main disadvantage of final access modifier is we cannot implement oops concepts in java. Ex : Inheritance, polymorphism.

final class : A final class cannot be extended or subclassed. We ar e preventing inheritance by marking a class as final. But we can still access the methods of this class by composition.

Ex: String class

final methods: Method overriding is one of the important features in java. But there are situations where we may not want to use this feature. Then we declared method as final which will print overriding. To allow a method from being overridden we use final access modifier for methods.

final variables : If a variable is declared as final ,it behaves like a constant . We cannot modify the value of final variable.

Any attempt to modify the final variable results in compilation error. The error is as follows “final variable cannot be assigned.”

49. What about abstract classes in java?

Sometimes we may come across a situation where we cannot provide implementation to all the methods in a class.

We want to leave the implementation to a class that extends it. In such case we declare a class as abstract.To make a class abstract we use key word abstract. Any class that contains one or more abstract methods is declared as abstract. If we don't declare class as abstract which contains abstract methods we get compile time error. We get the following error.

"The type must be an abstract class to define abstract methods."

Signature ; abstract class

{

}

For example if we take a vehicle class we cannot provide implementation to it because there may be two wheelers , four wheelers etc. At that moment we make vehicle class abstract. All the common features of vehicles are declared as abstract methods in vehicle class. Any class which extends vehicle will provide its method implementation. It's the responsibility of subclass to provide implementation.

The important features of abstract classes are :

1) Abstract classes cannot be instantiated.

2) An abstract classes contains abstract methods, concrete methods or both.

3) Any class which extends abstract class must override all methods of abstract class.

4) An abstract class can contain either 0 or more abstract methods.

Though we cannot instantiate abstract classes we can create object references . Through superclass references we can point to subclass.

50. Can we create constructor in abstract class ?

We can create constructor in abstract class , it does'nt give any compilation error. But when we cannot instantiate class there is no use in creating a

constructor for abstract class.

51. What are abstract methods in java?

An abstract method is the method which does'nt have any body. Abstract method is declared with keyword abstract and semicolon in place of method body.

Signature : public abstract void ();

Ex : public abstract void getDetails();

It is the responsibility of subclass to provide implementation to abstract method defined in abstract class.

Java Exception Handling Interview questions

52. What is an exception in java?

In java exception is an object. Exceptions are created when an abnormal situations are arised in our program.

Exceptions can be created by JVM or by our application code. All Exception classes are defined in java.lang. In otherwords we can say Exception as run time error.

53. State some situations where exceptions may arise in java?

1) Accesing an element that does not exist in array.

2) Invalid conversion of number to string and string to number.

(NumberFormatException)

3) Invalid casting of class

(Class cast Exception)

4) Trying to create object for interface or abstract class

(Instantiation Exception)

54. What is Exception handling in java?

Exception handling is a mechanism what to do when some abnormal situation arises in program. When an exception is raised in program it leads to termination of program when it is not handled properly. The significance of exception handling comes here in order not to terminate a program abruptly and to continue with the rest of program normally.

This can be done with help of Exception handling.

55. What is an eror in Java?

Error is the subclass of Throwable class in java. When errors are caused by our program we call that as Exception, but some times exceptions are caused due to some environment issues such as running out of memory. In such cases we can't handle the exceptions. Exceptions which cannot be recovered are called as errors in java.

Ex : Out of memory issues.

56. What are advantages of Exception handling in java?

1. Separating normal code from exception handling code to avoid abnormal termination of program.
2. Categorizing in to different types of Exceptions so that rather than handling all exceptions with Exception root class we can handle with specific exceptions. It is recommended to handle exceptions with specific Exception instead of handling with Exception root class.
3. Call stack mechanism : If a method throws an exception and it is not handled immediately, then that exception is propagated or thrown to the caller of that method. This propogation continues till it finds an appropriate exception handler ,if it finds handler it would be handled otherwise program terminates abruptly.

57) In how many ways we can do exception handling in java?

We can handle exceptions in either of the two ways :

1. By specifying try catch block where we can catch the exception.
2. Declaring a method with throws clause .

58. List out five keywords related to Exception handling ?

- Try
- Catch
- throw
- throws
- finally

59. Explain try and catch keywords in java?

In try block we define all exception causing code. In java try and catch forms a unit. A catch block catches the exception thrown by preceding try block. Catch block cannot catch an exception thrown by another try block. If there is no exception causing code in our program or exception is not raised in our code jvm ignores the try catch block.

Syntax :

try

{

}

Catch(Exception e)

{

}

60. Can we have try block without catch block?

Each try block requires atleast one catch block or finally block. A try block without catch or finally will result in compiler error. We can skip either of catch or finally block but not both.

61. Can we have multiple catch block for a try block?

In some cases our code may throw more than one exception. In such case we can specify two or more catch clauses, each catch handling different type of exception. When an exception is thrown jvm checks each catch statement in order and the first one which matches the type of exception is execution and remaining catch blocks are skipped.

Try with multiple catch blocks is highly recommended in java.

If try with multiple catch blocks are present the order of catch blocks is very important and the order should be from child to parent.

62. Explain importance of finally block in java?

Finally block is used for cleaning up of resources such as closing connections, sockets etc. if try block executes with no exceptions then finally is called after try block without executing catch block. If there is exception thrown in try block finally block executes immediately after catch block.

If an exception is thrown,finally block will be executed even if the no catch block handles the exception.

63. Can we have any code between try and catch blocks?

We shouldn't declare any code between try and catch block. Catch block should immediately start after try block.

```
try{

//code

}

System.out.println("one line of code"); // illegal

catch(Exception e){

//
```

```
}
```

64. Can we have any code between try and finally blocks?

We shouldn't declare any code between try and finally block. finally block should immediately start after catch block. If there is no catch block it should immediately start after try block.

```
try{

//code

}

System.out.println("one line of code"); // illegal

finally{

//

}
```

65. Can we catch more than one exception in single catch block?

From Java 7, we can catch more than one exception with single catch block. This type of handling reduces the code duplication.

Note : When we catch more than one exception in single catch block , catch parameter is implicity final. We cannot assign any value to catch parameter.

```
Ex : catch(ArrayIndexOutOfBoundsException || ArithmeticException e)

{

}
```

In the above example e is final we cannot assign any value or modify e in catch statement.

66. What are checked Exceptions?

1) All the subclasses of Throwable class except error,Runtime Exception and its subclasses are checked exceptions.

2. Checked exception should be thrown with keyword throws or should be provided try catch block, else the program would not compile. We do get compilation error.

Examples :

1) IOException,

2. SQlException,

3) FileNotFoundException,

4) InvocationTargetException,

5. CloneNotSupportedException

6. ClassNotFoundException

7) InstantiationException

67. What are unchecked exceptions in java?

All subclasses of RuntimeException are called unchecked exceptions. These are unchecked exceptions because compiler does not checks if a method handles or throws exceptions.

Program compiles even if we do not catch the exception or throws the exception.

If an exception occurs in the program,program terminates . It is difficult to handle these exceptions because there may be many places causing exceptions.

Example :

1) Arithmetic Exception

2) ArrayIndexOutOfBoundsException

3. ClassCastException

4) IndexOutOfBoundException

5) NullPointerException

6) NumberFormatException

7. StringIndexOutOfBounds

8) UnsupportedOperationException

68. Explain differences between checked and Unchecked exceptions in java?

Unchecked Exception Checked Exception

1) All the subclasses of RuntimeException are called unchecked exception. 1. All subclasses of Throwable class except RuntimeException are called as checked exceptions

2) Unchecked exceptions need not be handled at compile time 2. Checked Exceptions need to be handled at compile time.

3) These exceptions arise mostly due to coding mistakes in our program.

4) ArrayIndexOutOfBoundsException, ClassCastException, IndexOutOfBoundException 4. SqlException,FileNotFoundException,ClassNotFoundException

69. What is default Exception handling in java?

When JVM detects exception causing code, it constructs a new exception handling object by including the following information.

1) Name of Exception

2. Description about the Exception

3. Location of Exception.

After creation of object by JVM it checks whether there is exception handling code or not. If there is exception handling

code then exception handles and continues the program. If there is no exception handling code JVM give the responsibility of exception handling to default handler and terminates abruptly.

Default Exception handler displays description of exception,prints the stacktrace and location of exception and terminates the program.

Note : The main disadvantage of this default exception handling is program terminates abruptly.

70. Explain throw keyword in java?

Generally JVM throws the exception and we handle the exceptions by using try catch block. But there are situations where we have to throw userdefined exceptions or runtime exceptions. In such case we use throw keyword to throw exception explicitly.

Syntax : throw throwableInstance;

Throwable instance must be of type throwable or any of its subclasses.

After the throw statement execution stops and subsequent statements are not executed. Once exception object is thrown JVM checks is there any catch block to handle the exception. If not then the next catch statement till it finds the appropriate handler. If appropriate handler is not found ,then default exception handler halts the program and prints the description and location of exception.

In general we use throw keyword for throwing userdefined or customized exception.

71. Can we write any code after throw statement?

After throw statement jvm stop execution and subsequent statements are not executed. If we try to write any statement

after throw we do get compile time error saying unreachable code.

72. Explain importance of throws keyword in java?

Throws statement is used at the end of method signature to indicate that an exception of a given type may be thrown from the method.

The main purpose of throws keyword is to delegate responsibility of exception handling to the caller methods, in the case of checked exception.

In the case of unchecked exceptions, it is not required to use throws keyword.

We can use throws keyword only for throwable types otherwise compile time error saying incompatible types.

An error is unchecked , it is not required to handle by try catch or by throws.

Syntax : Class Test{

Public static void main(String args[]) throws IE

{

}

}

Note : The method should throw only checked exceptions and subclasses of checked exceptions.

It is not recommended to specify exception superclasses in the throws class when the actual exceptions thrown in the method are instances of their subclass.

73. Explain the importance of finally over return statement?

finally block is more important than return statement when both are present in a program.

For example if there is any return statement present inside try or catch block , and finally block is also present first finally statement will be executed and then return statement will be considered.

74. Explain a situation where finally block will not be executed?

Finally block will not be executed whenever jvm shutdowns. If we use system.exit(0) in try statement finally block if present will not be executed.

75. Can we use catch statement for checked exceptions when there is no chance of raising exception in our code?

If there is no chance of raising an exception in our code then we can't declare catch block for handling checked exceptions .This raises compile time error if we try to handle checked exceptions when there is no possibility of causing exception.

76. What are user defined exceptions?

To create customized error messages we use userdefined exceptions. We can create user defined exceptions as checked or unchecked exceptions.

We can create user defined exceptions that extend Exception class or subclasses of checked exceptions so that userdefined exception becomes checked.

Userdefined exceptions can extend RuntimeException to create userdefined unchecked exceptions.

Note : It is recommended to keep our customized exception class as unchecked,i.e we need to extend Runtime

Exception class but not Excpetion class.

77. Can we rethrow the same exception from catch handler?

Yes we can rethrow the same exception from our catch handler. If we want to rethrow checked exception from a catch block we need to declare that exception.

78. Can we nested try statements in java?

Yes try statements can be nested. We can declare try statements inside the block of another try statement.

79. Explain the importance of throwable class and its methods?

Throwable class is the root class for Exceptions. All exceptions are derived from this throwable class. The two main subclasses of Throwable are Exception and Error. The three methods defined in throwable class are :

1) void printStackTrace() :

This prints the exception information in the following format :

Name of the exception, description followed by stack trace.

2) getMessage()

This method prints only the description of Exception.

3) toString():

It prints the name and description of Exception.

80. Explain when ClassNotFoundException will be raised ?

When JVM tries to load a class by its string name, and couldn't able to find the class classNotFoundException will be thrown. An example for this exception is when class name is misspelled and when we try to load the class by string name hence class cannot be found which raises ClassNotFoundException.

81. Explain when NoClassDefFoundError will be raised ?

This error is thrown when JVM tries to load the class but no definition for that class is found NoClassDefFoundError will occur. The class may exist at compile time but unable to find at runtime. This might be due to misspelled classname at command line, or classpath is not specified properly , or the class file with byte code is no longer available.

83. What is process ?

A process is a program in execution.

Every process have their own memory space.Process are heavy weight and requires their own address space. One or more threads make a process.

Java Interview questions on threads

84. What is thread in java?

Thread is separate path of execution in program.

Threads are

1. Light weight

2) They share the same address space.

3. Creating thread is simple when compared to process because creating thread requires less resources when compared to process

4) Threads exists in process. A process have atleast one thread.

85. Difference between process and thread?

Process Thread

1) Program in execution. 1. Separate path of execution in program. One or more threads is called as process.

2) Processes are heavy weight 2. Threads are light weight.

3) Processes require separate address space. 3. Threads share same address space.

4) Interprocess communication is expensive. 4. Interthread communication is less expensive compared to processes.

5. Context switching from one process to another is costly. 5. Context switching between threads is low cost.

86. What is multitasking ?

Multitasking means performing more than one activity at a time on the computer. Example Using spreadsheet and using calculator at same time.

87. What are different types of multitasking?

There are two different types of multitasking :

1) Process based multitasking

2) Thread based multitasking

Process based multitasking : It allows to run two or more programs concurrently. In process based multitasking a process is the smallest part of code .

Example : Running Ms word and Ms powerpoint at a time.

Thread based multitasking : It allows to run parts of a program to run concurrently.

Example : Formatting the text and printing word document at same time .

Java supports thread based multitasking and provides built in support for multithreading.

88. What are the benefits of multithreaded programming?

Multithreading enables to use idle time of cpu to another thread which results in faster execution of program. In single threaded environment each task has to be completed before proceeding to next task making cpu idle.

89. Explain thread in java?

1) Thread is independent path of execution with in a program.

2) A thread consists of three parts Virtual Cpu, Code and data.

3) At run time threads share code and data i.e they use same address space.

4. Every thread in java is an object of java.lang.Thread class.

90. List Java API that supports threads?

java.lang.Thread : This is one of the way to create a thread. By extending Thread class and overriding run(. We can create thread in java.

java.lang.Runnable : Runnable is an interface in java. By implementing runnable interface and overriding run(. We can create thread in java.

java.lang.Object : Object class is the super class for all the classes in java. In object class we have three methods wait(), notify(), notifyAll() that supports threads.

java.util.concurrent : This package has classes and interfaces that supports concurrent programming.

Ex : Executor interface, Future task class etc.

91. What about main thread in java?

Main thread is the first thread that starts immediately after a program is started.

Main thread is important because :

1) All the child threads spawn from main thread.

2) Main method is the last thread to finish execution.

When JVM calls main method() it starts a new thread. main() method is temporarily stopped while the new thread starts running.

92. In how many ways we can create threads in java?

We can create threads in java by any of the two ways :

1) By extending Thread class

2) By Implementing Runnable interface.

93. Explain creating threads by implementing Runnable class?

This is first and foremost way to create threads . By implementing runnable interface and implementing run() method

we can create new thread.

Method signature : public void run()

Run is the starting point for execution for another thread within our program.

Example :

```
public class MyClass implements Runnable {

@Override

public void run() {

// T

}
```

}

94. Explain creating threads by extending Thread class ?

We can create a thread by extending Thread class. The class which extends Thread class must override the run()

method.

Example :

public class MyClass extends Thread {

@Override

public void run() {

// Starting point of Execution

}

}

95. Which is the best approach for creating thread ?

The best way for creating threads is to implement runnable interface.

When we extend Thread class we can't extend any other class.

When we create thread by implementing runnable interface we can implement Runnable interface. In both ways we have to implement run() method.

96. Explain the importance of thread scheduler in java?

Thread scheduler is part of JVM use to determine which thread to run at this moment when there are multiple threads.

Only threads in runnable state are choosen by scheduler.

Thread scheduler first allocates the processor time to the higher priority threads. To allocate microprocessor time in between the threads of the same priority, thread scheduler follows round robin fasion.

97. Explain the life cycle of thread?

A thread can be in any of the five states :

1) New : When the instance of thread is created it will be in New state.

Ex : Thread t= new Thread();

In the above example t is in new state. The thread is created but not in active state to make it active we need to call start() method on it.

2) Runnable state : A thread can be in the runnable state in either of the following two ways :

a. When the start method is invoked or

b) A thread can also be in runnable state after coming back from blocked or sleeping or waiting state.

3) Running state : If thread scheduler allocates cpu time, then the thread will be in running state.

4. Waited /Blocking/Sleeping state:

In this state the thread can be made temporarily inactive for a short period of time.

A thread can be in the above state in any of the following ways:

1) The thread waits to acquire lock of an object.

2) The thread waits for another thread to complete.

3) The thread waits for notification of other thread.

5. Dead State : A thread is in dead state when thread's run method execution is complete. It dies automatically when thread's run method execution is completed and the thread object will be garbage collected.

98. Can we restart a dead thread in java?

If we try to restart a dead thread by using start method we will get run time exception since the thread is not alive.

99. Can one thread block the other thread?

No one thread cannot block the other thread in java. It can block the current thread that is running.

100. Can we restart a thread already started in java?

A thread can be started in java using start() method in java. If we call start method second time once it is started it will cause RunTimeException(IllegalThreadStateException). A runnable thread cannot be restarted.

101. What happens if we don't override run method ?

If we don't override run method .Then default implementation of Thread class run() method will be executed and hence the thread will never be in runnable state.

102. Can we overload run() method in java?

We can overload run method but Thread class start method will always cal run method with no arguments. But the overloaded method will not be called by start method we have to explicitly call this start() method.

105. What is a lock or purpose of locks in java?

Lock also called monitor is used to prevent access to a shared resource by multiple threads.

A lock is associated to shared resource. Whenever a thread wants to access a shared resource if must first acquire a lock . If already a lock has been acquired by other it can't access that shared resource. At this moment the thread has to wait until another thread releases the lock on shared resource. To lock an object we use synchronization in java.

A lock protects section of code allowing only one thread to execute at at a time.

106. In how many ways we can do synchronization in java?

There are two ways to do synchronization in java:

1. Synchronized methods

2. Synchronized blocks

To do synchronization we use synchronize keyword.

107. What are synchronized methods ?

If we want a method of object to be accessed by single thread at a time we declare that method with synchronized keyword.

Signature :

public synchronized void methodName(){}

To execute synchronized method first lock has to be acquired on that object. Once synchronized method is called lock will be automatically acquired on that method when no other thread has lock on that method. once lock has been acquired then synchronized method gets executed. Once synchronized method execution completes automatically lock will be released. The prerequisite to execute a synchronized method is to acquire lock before method execution. If there is a lock already acquired by any other thread it waits till the other thread completes.

108. When do we use synchronized methods in java?

If multiple threads tries to access a method where method can manipulate the state of object , in such scenario we can declare a method as synchronized.

109. When a thread is executing synchronized methods , then is it possible to execute other synchronized methods simultaneously by other threads?

No it is not possible to execute synchronized methods by other threads when a thread is inside a synchronized method.

110. When a thread is executing a synchronized method , then is it possible for the same thread to access other synchronized methods of an object ?

Yes it is possible for thread executing a synchronized method to execute another synchronized method of an object.

```
public synchronized void methodName()

{

}
```

To execute synchronized method first lock has to be acquired on that object. Once synchronized method is called lock will be automatically acquired on that method when no other thread has lock on that method. once lock has been acquired then synchronized method gets executed. Once synchronized method execution completes automatically lock will be released. The prerequisite to execute a synchronized method is to acquire lock before method execution. If there is a lock already acquired by any other thread it waits till the other thread completes.

111. What are synchronized blocks in java?

Synchronizing few lines of code rather than complete method with the help of synchronized keyword are called

synchronized blocks.

Signature :

Synchronized (object reference){// code}

112. When do we use synchronized blocks and advantages of using synchronized blocks?

If very few lines of code requires synchronization then it is recommended to use synchronized blocks. The main

advantage of synchronized blocks over synchronized methods is it reduces the waiting time of threads and improves performance of the system.

113. What is class level lock ?

Acquiring lock on the class instance rather than object of the class is called class level lock. The difference between class level lock and object level lock is in class level lock lock is acquired on class .class instance and in object level lock ,lock is acquired on object of class.

114. Can we synchronize static methods in java?

Every class in java has a unique lock associated with it. If a thread wants to execute static synchronize method it need to acquire first class level lock. When a thread was executing static synchronized method no other thread can execute static synchronized method of class since lock is acquired on class.

But it can execute the following methods simultaneously :

1) Normal static methods

2) Normal instance methods

3. Synchronize instance methods

Signature :

synchronized(Classname.class){}

115. Can we use synchronized block for primitives?

Synchronized blocks are applicable only for objects if we try to use synchronized blocks for primitives we get compile time error.

116. What are thread priorities and importance of thread priorities in java?

When there are several threads in waiting, thread priorities determine which thread to run. In java programming language every thread has a priority. A thread inherits priority of its parent thread. By default thread has normal priority of 5. Thread scheduler uses thread priorities to decide when each thread is allowed to run. Thread scheduler runs higher priority threads first.

117. Explain different types of thread priorities ?

Every thread in java has priorities in between 1 to 10. By default priority is 5 (Thread.NORM_PRIORITY). The maximum priority would be 10 and minimum would be 1.Thread class defines the following constants(static final variables) to define properties.

Thread. MIN_PRIORITY = 1;

Thread.NORM_PRIORITY=5;

Thread. MAX_PRIORITY=10;

118. How to change the priority of thread or how to set priority of thread?

Thread class has a set method to set the priority of thread and get method to get the priority of the thread.

Signature : final void setPriority(int value);

The setPriority() method is a request to jvm to set the priority. JVM may or may not oblige the request.

We can get the priority of current thread by using getPriority() method of Thread class.

final int getPriority()

{

}

119. If two threads have same priority which thread will be executed first ?

We are not guaranteed which thread will be executed first when there are threads with equal priorities in the pool. It depends on thread scheduler to which thread to execute. The scheduler can do any of the following things :

1) It can pick any thread from the pool and run it till it completes.

2) It can give equal opportunity for all the threads by time slicing.

120. What all methods are used to prevent thread execution ?

There are three methods in Thread class which prevents execution of thread.

1) yield()

2) join()

3. Sleep()

121. Explain yield() method in thread class ?

yield() method makes the current running thread to move in to runnable state from running state giving chance to remaining threads of equal priority which are in waiting state. yield() makes current thread to sleep for a specified amount of time.There is no guarantee that moving a current running thread from runnable to running state. It all depends on thread scheduler it doesn’t gurantee anything.

Calling yield() method on thread does not have any affect if object has a lock. The thread does'nt lose any lock if it has acquired a lock earlier.

Signature :

public static native void yield()

{

—

}

122. Is it possible for yielded thread to get chance for its execution again ?

yield(. Causes current thread to sleep for specified amount of time giving opportunity for other threads of equal priority to

execute. Thread scheduler decides whether it get chance for execution again or not. It all depends on mercy of thread

scheduler.

123. Explain the importance of join() method in thread class?

A thread can invoke the join() method on other thread to wait for other thread to complete its execution. Assume we have two threads, t1 and t2 threads . A running thread t1 invokes join() on thread t2 then t1 thread will wait in to waiting state until t2 completes. Once t2 completes the execution, t1 will continue.

join() method throws Interrupted Exception so when ever we use join() method we should handle Interrrupted Exception by throws or by using try catch block.

Syntax :

public final void join() throws InterruptedException{

```
}

public final synchronized void join(long millis)throws InterruptedException

{

}

public final synchronized void join(long millis, int nanos)throws
InterruptedException

{

}
```

124. Explain purpose of sleep() method in java?

sleep() method causes current running thread to sleep for specified amount of time . sleep() method is the minimum amount of the time the current thread sleeps but not the exact amount of time.

Signature :

```
public static native void sleep(long millis) throws InterruptedException{

}

public static void sleep(long millis, int nanos)

throws InterruptedException {

}
```

125) Assume a thread has lock on it, calling sleep() method on that thread will release the lock?

Calling sleep() method on thread which has lock does’nt affect. Lock will not be released though the thread sleeps for a specified amount of time.

126. Can sleep() method causes another thread to sleep?

No sleep() method causes current thread to sleep not any other thread.

127. What about interrupt() method of thread class ?

Thread class interrupt() method is used to interrupt current thread or another thread. It doesnot mean the current thread to stop immediately, it is polite way of telling or requesting to continue your present work. That is the reason we may not see the impact of interrupt call immediately.

Initially thread has a boolean property(interrupted status) false. So when we call interrupt() method status would set to true. This causes the current thread to continue its work and does not have impact immediately.

If a thread is in sleeping or waiting status (i.e thread has executed wait () or sleep() method) thread gets interrupted it stops what it is doing and throws an interrupted exception. This is reason we need to handle interrupted exception with throws or try/ catch block.

128. What about interthread communication and how it takes place in java?

Usually threads are created to perform different unrelated tasks but there may be situations where they may perform

related tasks. Interthread communication in java is done with the help of following three methods :

1. wait()
2. notify()
3. notifyAll()

129. Explain wait(), notify() and notifyAll() methods of object class ?

wait() : wait() method() makes the thread current thread sleeps and releases the lock until some other thread acquires the lock and calls notify().

notify() :notify() method wakes up the thread that called wait on the same object.

notfiyAll() :notifyAll() method wakes up all the threads that are called wait() on the same object. The highest priority threads will run first.

All the above three methods are in object class and are called only in synchronized context.

All the above three methods must handle InterruptedException by using throws clause or by using try catch clause.

130. Explain why wait() , notify() and notifyAll() methods are in Object class rather than in thread class?

First to know why they are in object class we should know what wait(), notify(), notifyAll() methods do. wait() , notify(), notifyAll() methods are object level methods they are called on same object.wait(), notify(), notifyAll() are called on an shared object so to they are kept in object class rather than thread class.

Interview questions on Nested classses and inner classes

131. Is there a way to increase the size of an array after its declaration?

Arrays are static and once we have specified its size, we can't change it. If we want to use such collections where we may require a change of size (no of items), we should prefer vector over array.

132. If an application has multiple classes in it, is it okay to have a main method in more than one class?

If there is main method in more than one classes in a java application, it won't cause any issue as entry point for any application will be a specific class and code will start from the main method of that particular class only.

133. I want to persist data of objects for later use. What's the best approach to do so?

The best way to persist data for future use is to use the concept of serialization.

134. What is a Local class in Java?

In Java, if we define a new class inside a particular block, it's called a local class. Such a class has local scope and isn't usable outside the block where its defined.

135. String and StringBuffer both represent String objects. Can we compare String and StringBuffer in Java?

Although String and StringBuffer both represent String objects, we can't compare them with each other and if we try to compare them, we get an error.

136. Which API is provided by Java for operations on set of objects?

Java provides a Collection API which provides many useful methods which can be applied on a set of objects. Some of the important classes provided by Collection API include ArrayList, HashMap, TreeSet and TreeMap.

137. Can we cast any other type to Boolean Type with type casting?

No, we can neither cast any other primitive type to Boolean data type nor can cast Boolean data type to any other primitive data type.

138. Can we use different return types for methods when overridden?

The basic requirement of method overriding in Java is that the overridden method should have same name, and parameters.But a method can be

overridden with a different return type as long as the new return type extends the original.

For example , method is returning a reference type.

```
Class B extends A{

A method(int x){

//original method

}

B method(int x){

//overridden method

}

}
```

139. What are nested classes in java?

Class declared with in another class is defined as nested class.

There are two types of nested classes in java.

1. Static nested class

2) Non static nested class

A static nested class has static keyword declared before class definition.

140. What are inner classes or non static nested classes in java?

Nested classes without any static keyword declaration in class definition are defined as non static nested classes.

Generally non static nested classes are referred as inner classes.

There are three types of inner classes in java :

1. Local inner class

2) Member inner class

3) Anonymous inner class

141. Why to use nested classes in java?

(or)

What is the purpose of nested class in java?

1) Grouping of related classes

Classes which are not reusable can be defined as inner class instead of creating inner class.

For example : We have a submit button upon click of submit button we need to execute some code. This code is related only to that class and cannot be reused for other class . Instead of creating a new class we can create inner class

2) To increase encapsulation :

Inner class can access private members of outer class.so by creating getter and setter methods for private variables ,outside world can access these variables. But by creating inner class private variables can be accesed only by inner class.

3. Code readable and maintainable :

Rather than creating a new class we can create inner class so that it is easy to maintain.

4. Hiding implementation :

Inner class helps us to hide implementation of class.

142. What about static nested classes in java?

When a static class is defined inside a enclosing class we define that as nested class. Static nested classes are not

inner classes. Static nested classes can be instantiated without instance of outer class.

A static nested doesnot have access to instance variables and non static methods of outer class.

143. How to instantiate static nested classes in java?

We can access static members and static methods of outer class without creating any instance of outer class.

Syntax for instantiating Static nested class :

OuterclassName.StaticNestedClassName ref=new OuterclassName.StaticNestedClassName();

144. What about method local inner classes or local inner classes in java?

Nested classes defined inside a method are local inner classes. We can create objects of local inner class only inside

method where class is defined. A local inner classes exist only when method is invoked and goes out of scope when method returns.

145. What about features of local inner class?

1. Local inner class does not have any access specifier.
2. We cannot use access modifiers static for local inner class. But we can use abstract and final for local inner class.
3. We cannot declare static members inside local inner classes.
4. We can create objects of local inner class only inside method where class is defined.

5. Method local inner classes can only access final variables declared inside a method.
6. Method local inner classes can be defined inside loops(for,while) and blocks such as if etc.

146. What about anonymous inner classes in java?

Inner class defined without any class name is called anonymous inner class. Inner class is declared and instantiated using new keyword.The main purpose of anonymous inner classes in java are to provide interface implementation. We use anonymous classes when we need only one instance for a class. We can use all members of enclosing class and final local variables.

When we compile anonymous inner classes compiler creates two files

1. EnclosingName.class

2. EnclsoingName$1.class

147. Explain restrictions for using anonymous inner classes?

1) An anonymous inner class cannot have any constructor because there is no name for class.

2) An anonymous inner class cannot define static methods, fields or classes.

3. We cannot define an interface anonymously.

4) Anonymous inner class can be instantiated only once.

148. Is this valid in java ? can we instantiate interface in java?

Runnable r = new Runnable() {

@Override

public void run() {

```
}

};
```

Runnable is an interface.If we see the above code it looks like we are instantiating Runnable interface. But we are not

instantiating interface we are instantiating anonymous inner class which is implementation of Runnable interface.

149. What about member inner classes?

Non static class defined with in enclosing class are called member inner class. A member inner class is defined at member level of class. A member inner class can access the members of outer class including private members.

Features of member inner classes :

1) A member inner class can be declared abstract or final.

2) A member inner class can extend class or implement interface.

3) An inner class cannot declare static fields or methods.

4) A member inner class can be declared with public, private, protected or default access.

150. How to instantiate member inner class?

OuterClassName.InnerclassName inner=new OuterClassReference.new InnerClassName();

We cannot instantiate inner class without outer class reference.

151. How to do encapsulation in Java?

- Make instance variables private.
- Define getter and setter methods to access instance variables .

152. What are reference variables in java?

Variables which are used to access objects in java are called reference variables.

Ex : Employee emp=new Employee();

In the above example emp is reference variable.

Reference variable can be of only one type.

A reference variable can point to any number of objects. But if a reference variable is declared final it can't point to other objects.

A reference variable can be declared either to a class type or interface type. If a reference variable is declared with interface type it points to the class that implements the interface.

153. Will the compiler creates a default constructor if I have a parameterized constructor in the class?

No compiler won't create default constructor if there is parameterized constructor in the class. For example if I have a class with no constructors, then compiler will create default constructor.

For Example :

public classCar {}

In the above Car class there are no constructors so compiler creates a default constructor.

public classCar {Car(String name) {

}

}

In this example compiler won't create any default constructor because already there is one constructor in the Car class.

154. Can we have a method name same as class name in java?

Yes we can have method name same as class name it won't throw any compilation error but it shows a warning message that method name is same as class name.

155. Can we override constructors in java?

Only methods can be overridden in java. Constructors can't be inherited in java. So there is no point of verriding constructors in java.

156. Can Static methods access instance variables in java?

No.Instance variables can't be accessed in static methods. When we try to access instance variable in static method we get compilation error. The error is as follows:

Cannot make a static reference to the non static field name

157. How do we access static members in java?

Instance variables and instance methods can be accessed using reference variable . But to access static variables or static methods we use Class name in java.

158. Can we override static methods in java?

Static methods can't be overridden. If we have a static method in superclass and subclass with same signature then we don't say that as overriding. We call that as

159. Difference between object and reference?

Reference and object are both different. Objects are instances of class that resides in heap memory. Objects does'nt have any name so to access objects

we use references. There is no alternative way to access objects except through references.

Object cannot be assigned to other object and object cannot be passed as an argument to a method.

Reference is a variable which is used to access contents of an object. A reference can be assigned to other reference ,passed to a method.

160. Objects or references which of them gets garbage collected?

Objects get garbage collected not its references.

Java interview Questions

CORE JAVA Questions with Answers Pdf Download