|  |
| --- |
| How garbage collector knows that the object is not in use and needs to be removed? |
| |  | | --- | | Answer: | | Garbage collector reclaims objects that are no longer being used, clears their memory, and keeps the memory available for future allocations. This is done via bookkeeping the references to the objects. Any unreferenced object is a garbage and will be collected. | |

|  |  |  |
| --- | --- | --- |
| Question: Can Java thread object invoke start method twice?  |  | | --- | | Code: | | package com.java2novice.exmpcode;  public class MyExmpCode extends Thread{  public void run(){  System.out.println("Run");  }    public static void main(String a[]){  Thread t1 = new Thread(new MyExmpCode());  t1.start();  t1.start();  }  } | |
| |  | | --- | | Answer: | | No, it throws IllegalThreadStateException | |

|  |
| --- |
| Question: Give the list of Java Object class methods. |
| |  | | --- | | Answer: | | clone() - Creates and returns a copy of this object.  equals() - Indicates whether some other object is "equal to" this one.  finalize() - Called by the garbage collector on an object when garbage collection  determines that there are no more references to the object.  getClass() - Returns the runtime class of an object.  hashCode() - Returns a hash code value for the object.  notify() - Wakes up a single thread that is waiting on this object's monitor.  notifyAll() - Wakes up all threads that are waiting on this object's monitor.  toString() - Returns a string representation of the object.  wait() - Causes current thread to wait until another thread invokes the notify() method  or the notifyAll() method for this object. | |

|  |
| --- |
| Question: Can we call servlet destory() from service()? |
| |  | | --- | | Answer: | | As you know, destory() is part of servlet life cycle methods, it is used to kill the  servlet instance. Servlet Engine is used to call destory(). In case, if you call destory  method from service(), it just execute the code written in the destory(), but it wont  kill the servlet instance. destroy() will be called before killing the servlet instance  by servlet engine. | |

|  |
| --- |
| Question: Can we override static method? |
| |  | | --- | | Answer: | | We cannot override static methods. Static methods are belogs to class, not belongs  to object. Inheritance will not be applicable for class members | |

|  |
| --- |
| Question: Can you list serialization methods? |
| |  | | --- | | Answer: | | Serialization interface does not have any methods. It is a marker interface.  It just tells that your class can be serializable. | |

|  |
| --- |
| Question: What is the difference between super() and this()? |
| |  | | --- | | Answer: | | super() is used to call super class constructor, whereas this() used to call  constructors in the same class, means to call parameterized constructors. | |

|  |
| --- |
| Question: How to prevent a method from being overridden? |
| |  | | --- | | Answer: | | By specifying final keyword to the method you can avoid overriding  in a subcalss. Similarlly one can use final at class level to  prevent creating subclasses. | |

|  |
| --- |
| Question: Can we create abstract classes without any abstract methods? |
| |  | | --- | | Answer: | | Yes, we can create abstract classes without any abstract methods. | |

|  |
| --- |
| Question: Can we create abstract classes without any abstract methods? |
| |  | | --- | | Answer: | | Yes, we can create abstract classes without any abstract methods. | |

|  |
| --- |
| Question: Can we have static methods in interface? |
| |  | | --- | | Answer: | | By default, all methods in an interface are decleared as public, abstract. It will never be static. But this concept is changed with java 8. Java 8 came with new feature called "default methods" with in interfaces [click here](http://www.java2novice.com/java-8/interface-default-methods/) for more details. | |

|  |
| --- |
| Question: What is transient variable? |
| |  | | --- | | Answer: | | Transient variables cannot be serialized. During serialization process,  transient variable states will not be serialized. State of the value will  be always defaulted after deserialization. | |

|  |
| --- |
| Question: Incase, there is a return at the end of try block, will execute finally block? |
| |  | | --- | | Answer: | | Yes, the finally block will be executed even after writing return statement  at the end fo try block. It returns after executing finally block. | |

|  |
| --- |
| Question: What is abstract class or abstract method? |
| |  | | --- | | Answer: | | We cannot create instance for an abstract class. We can able to create  instance for its subclass only. By specifying abstract keyword just before  class, we can make a class as abstract class.  public abstract class MyAbstractClass{  }  Abstract class may or may not contains abstract methods. Abstract method is  just method signature, it does not containes any implementation. Its subclass  must provide implementation for abstract methods. Abstract methods are looks  like as given below:  public abstract int getLength(); | |

|  |
| --- |
| Question: What is default value of a boolean? |
| |  | | --- | | Answer: | | Default value of a boolean is false. | |

|  |
| --- |
| Question: When to use LinkedList or ArrayList? |
| |  | | --- | | Answer: | | Accessing elements are faster with ArrayList, because it is index based.  But accessing is difficult with LinkedList. It is slow access. This is  to access any element, you need to navigate through the elements one by  one. But insertion and deletion is much faster with LinkedList, because  if you know the node, just change the pointers before or after nodes.  Insertion and deletion is slow with ArrayList, this is because, during  these operations ArrayList need to adjust the indexes according to  deletion or insetion if you are performing on middle indexes. Means,  an ArrayList having 10 elements, if you are inserting at index 5, then  you need to shift the indexes above 5 to one more. | |

|  |
| --- |
| Question: What is daemon thread? |
| |  | | --- | | Answer: | | Daemon thread is a low priority thread. It runs intermittently  in the back ground, and takes care of the garbage collection  operation for the java runtime system. By calling setDaemon()  method is used to create a daemon thread. | |

|  |
| --- |
| Question: Does each thread in java uses seperate stack? |
| |  | | --- | | Answer: | | In Java every thread maintains its own separate stack. It is  called Runtime Stack but they share the same memory. | |

|  |
| --- |
| Question: Does each thread in java uses seperate stack? |
| |  | | --- | | Answer: | | In Java every thread maintains its own separate stack. It is  called Runtime Stack but they share the same memory. | |

|  |  |  |
| --- | --- | --- |
| Question: Find out below switch statement output.  |  | | --- | | Code: | | public static void main(String a[]){  int price = 6;  switch (price) {  case 2: System.out.println("It is: 2");  default: System.out.println("It is: default");  case 5: System.out.println("It is: 5");  case 9: System.out.println("It is: 9");  }  } | |
| |  | | --- | | Answer: | | It is: default  It is: 5  It is: 9 | |

|  |  |  |
| --- | --- | --- |
| Question: Does system.exit() in try block executes code in finally block?  |  | | --- | | Code: | | try{  System.out.println("I am in try block");  System.exit(1);  } catch(Exception ex){  ex.printStackTrace();  } finally {  System.out.println("I am in finally block!!!");  } | |
| |  | | --- | | Answer: | | It will not execute finally block. The program will be terminated  after System.exit() statement. | |
| Question: What is fail-fast in java? |
| |  | | --- | | Answer: | | A fail-fast system is nothing but immediately report any failure that  is likely to lead to failure. When a problem occurs, a fail-fast system  fails immediately. In Java, we can find this behavior with iterators.  Incase, you have called iterator on a collection object, and another  thread tries to modify the collection object, then concurrent modification  exception will be thrown. This is called fail-fast. | |

|  |
| --- |
| Question: What is final, finally and finalize? |
| |  | | --- | | Answer: | | final:  final is a keyword. The variable decleared as final should be  initialized only once and cannot be changed. Java classes  declared as final cannot be extended. Methods declared as final  cannot be overridden.    finally:  finally is a block. The finally block always executes when the  try block exits. This ensures that the finally block is executed  even if an unexpected exception occurs. But finally is useful for  more than just exception handling - it allows the programmer to  avoid having cleanup code accidentally bypassed by a return,  continue, or break. Putting cleanup code in a finally block is  always a good practice, even when no exceptions are anticipated.    finalize:  finalize is a method. Before an object is garbage collected, the  runtime system calls its finalize() method. You can write system  resources release code in finalize() method before getting garbage  collected. | |

|  |
| --- |
| Question: In java, are true and false keywords? |
| |  | | --- | | Answer: | | true, false, and null might seem like keywords, but they are actually  literals. You cannot use them as identifiers in your programs. | |

|  |
| --- |
| Question: What are the different session tracking methods? |
| |  | | --- | | Answer: | | Cookies:  You can use HTTP cookies to store information. Cookies will be  stored at browser side.  URL rewriting:  With this method, the information is carried through url as  request parameters. In general added parameter will be sessionid,  userid.  HttpSession:  Using HttpSession, we can store information at server side. Http  Session provides methods to handle session related information.    Hidden form fields:  By using hidden form fields we can insert information in the webpages  and these information will be sent to the server. These fields are not  visible directly to the user, but can be viewed using view source  option from the browsers. The hidden form fields are as given below:    <input type='hidden' name='siteName' value='java2novice'/> | |

|  |
| --- |
| Question: What is the purpose of garbage collection? |
| |  | | --- | | Answer: | | The garbage collection process is to identify the objects which are  no longer referenced or needed by a program so that their resources can be  reclaimed and reused. These identified objects will be discarded | |

|  |
| --- |
| Question: What are the types of ResultSet? |
| |  | | --- | | Answer: | | The type of a ResultSet object determines the level of its functionality in  two areas: the ways in which the cursor can be manipulated, and how concurrent  changes made to the underlying data source are reflected by the ResultSet object.  The sensitivity of a ResultSet object is determined by one of three different  ResultSet types:  TYPE\_FORWARD\_ONLY:  The result set cannot be scrolled; its cursor moves forward only, from  before the first row to after the last row. The rows contained in the  result set depend on how the underlying database generates the results.  That is, it contains the rows that satisfy the query at either the time  the query is executed or as the rows are retrieved.    TYPE\_SCROLL\_INSENSITIVE:  The result can be scrolled; its cursor can move both forward and backward  relative to the current position, and it can move to an absolute position.  The result set is insensitive to changes made to the underlying data source  while it is open. It contains the rows that satisfy the query at either the  time the query is executed or as the rows are retrieved.    TYPE\_SCROLL\_SENSITIVE:  The result can be scrolled; its cursor can move both forward and backward  relative to the current position, and it can move to an absolute position.  The result set reflects changes made to the underlying data source while  the result set remains open.  The default ResultSet type is TYPE\_FORWARD\_ONLY. | |
| Question: What is difference between wait and sleep methods in java? |
| |  | | --- | | Answer: | | sleep():  It is a static method on Thread class. It makes the current thread into the  "Not Runnable" state for specified amount of time. During this time, the thread  keeps the lock (monitors) it has acquired.    wait():  It is a method on Object class. It makes the current thread into the "Not Runnable"  state. Wait is called on a object, not a thread. Before calling wait() method, the  object should be synchronized, means the object should be inside synchronized block.  The call to wait() releases the acquired lock. | |
| Question: What is servlet context? |
| |  | | --- | | Answer: | | The servlet context is an interface which helps to communicate with  other servlets. It contains information about the Web application and  container. It is kind of application environment. Using the context, a  servlet can obtain URL references to resources, and store attributes that  other servlets in the context can use. | |

|  |
| --- |
| Question: What happens if one of the members in a class does not implement Serializable interface? |
| |  | | --- | | Answer: | | When you try to serialize an object which implements Serializable  interface, incase if the object includes a reference of an non  serializable object then NotSerializableException will be thrown. | |

|  |
| --- |
| Question: What is race condition? |
| |  | | --- | | Answer: | | A race condition is a situation in which two or more threads or  processes are reading or writing some shared data, and the final  result depends on the timing of how the threads are scheduled.  Race conditions can lead to unpredictable results and subtle  program bugs. A thread can prevent this from happening by locking  an object. When an object is locked by one thread and another  thread tries to call a synchronized method on the same object,  the second thread will block until the object is unlocked. | |

|  |
| --- |
| Question: How to get current time in milli seconds? |
| |  | | --- | | Answer: | | System.currentTimeMillis() returns the current time in milliseconds.  It is a static method, returns long type. | |

|  |
| --- |
| Question: How can you convert Map to List? |
| |  | | --- | | Answer: | | We know that Map contains key-value pairs, whereas a list contains  only objects. Since Entry class contains both key-value pair,  Entry class will helps us to convert from Map (HashMap) to  List (ArrayList). By using Map.entrySet() you will get Set  object, which intern you can use it to convert to list object. |  |  | | --- | | Code: | | public static void main(String a[]){  Map<String, String> wordMap = new HashMap<String, String>();  Set<Entry<String, String>> set = wordMap.entrySet();  List<Entry<String, String>> list = new ArrayList<Entry<String, String>>(set);  } | |

|  |
| --- |
| Question: What is strictfp keyword? |
| |  | | --- | | Answer: | | By using strictfp keyword, we can ensure that floating point operations  take place precisely. | |
| Question: What is System.out in Java? |
| Question: What is System.out in Java?  |  | | --- | | Answer: | | Here out is an instance of PrintStream. It is a static member variable in  System class. This is called standard output stream, connected to console. | |
| |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Question: What is difference between ServletOuptputStream and PrintWriter? | | | | | | | | | |  | | --- | | Answer: | | ServletOutputStream: ServletResponse.getOutputStream() returns a ServletOutputStream  suitable for writing binary data in the response. The servlet  container does not encode the binary data, it sends the raw data  as it is.    PrintWriter: ServletResponse.getWriter() returns PrintWriter object which sends  character text to the client. The PrintWriter uses the character  encoding returned by getCharacterEncoding(). If the response's  character encoding has not been specified then it does default  character encoding. | | | | | | | | | | Question: What is java static import? | | | | | | | | | | |  | | --- | | Answer: | | By using static imports, we can import the static members from a class  rather than the classes from a given package. For example, Thread class has  static sleep method, below example gives an idea:  import static java.lang.Thread;  public class MyStaticImportTest {  public static void main(String[] a) {  try{  sleep(100);  } catch(Exception ex){    }  }  } | | | | | | | | | | | Question: When to use String and StringBuffer? | | | | | | | | | | | |  | | --- | | Answer: | | We know that String is immutable object. We can not change the value  of a String object once it is initiated. If we try to change the value of  the existing String object then it creates new object rather than changing  the value of the existing object. So incase, we are going to do more  modificatios on String, then use StringBuffer. StringBuffer updates the  existing objects value, rather creating new object. | | | | | | | | | | | | Question: What is difference between StringBuffer and StringBuilder? | | | | | | | | | | | | |  | | --- | | Answer: | | The only difference between StringBuffer and StringBuilder is StringBuffer  is thread-safe, that is StringBuffer is synchronized. | | | | | | | | | | | | | Question: What is wrapper class in java? | | | | | | | | | | | | | |  | | --- | | Answer: | | Everything in java is an object, except primitives. Primitives are  int, short, long, boolean, etc. Since they are not objects, they cannot  return as objects, and collection of objects. To support this, java provides  wrapper classes to move primitives to objects. Some of the wrapper classes  are Integer, Long, Boolean, etc. | | | | | | | | | | | | | | Question: Is Iterator a Class? | | | | | | | | |  | | --- | | Answer: | | Iterator is an interface. It is not a class. It is used to  iterate through each and every element in a list. Iterator is  implemented Iterator design pattern. | | | | | | | | | Question: What is java classpath? | | | | | | | |  | | --- | | Answer: | | The classpath is an environment variable. It is used to  let the compiler know where the class files are available  for import. | | | | | | | | Question: Can a class in java be private? | | | | | | |  | | --- | | Answer: | | We can not declare top level class as private. Java allows  only public and default modifier for top level classes in java.  Inner classes can be private. | | | | | | | Question: Is null a keyword in java? | | | | | |  | | --- | | Answer: | | The null value is not a keyword in java. true and flase are also not  keywords in java. They are reserved words in java language. | | | | | | Question: What is the initial state of a thread when it is started? | | | | | | | | | | | | | | |  | | --- | | Answer: | | When the thread is createdn and started, initially it will be  in the ready state. | | | | | | | | | | | | | | | Question: What is the super class for Exception and Error? | | | | | | | | | | | | | | | |  | | --- | | Answer: | | The super class or base class for Exception and Error is Throwable. | | | | | | | | | | | | | | | | Question: What is Class.forName()? | | | | |  | | --- | | Answer: | | Class.forName() loads the class into the ClassLoader. | | | | | Question: Can interface be final? | | | |  | | --- | | Answer: | | No. We can not instantiate interfaces, so in order to make interfaces  useful we must create subclasses. The final keyword makes a class unable  to be extended. | | | | Question: What is the difference between exception and error? | | | | | | | | | | | | | | | | |  | | --- | | Answer: | | An error is an irrecoverable condition occurring at runtime like out of  memory error. These kind of jvm errors cannot be handled at runtime.  Exceptions are because of condition failures, which can be handled  easily at runtime. | | | | | | | | | | | | | | | | | Question: What is default value of a local variables? | | | | | | | | | | | | | | | | | |  | | --- | | Answer: | | The local variables are not initialized to any default values. We should  not use local variables with out initialization. Even the java compiler  throws error. | | | | | | | | | | | | | | | | | | Question: What is local class in java? | | |  | | --- | | Answer: | | In java, local classes can be defined in a block as in a  method body or local block. | | | Question: Can we initialise uninitialized final variable? | | | | | | | | | | | | | | | | | | |  | | --- | | Answer: | | Yes. We can initialise blank final variable in constructor, only in construtor.  The condition here is the final variable should be non-static. | | | | | | | | | | | | | | | | | | | Question: Can we declare abstract method as final? | | | | | | | | | | | | | | | | | | | |  | | --- | | Answer: | | No, we can not declare abstract method as final. We have to  proved implementation to abstract methods in subclasses. | | | | | | | | | | | | | | | | | | | | Question: Can we have finally block without catch block? | | | | | | | | | | | | | | | | | | | | |  | | --- | | Answer: | | Yes, we can have finally block without catch block. | | | | | | | | | | | | | | | | | | | | |

|  |
| --- |
| Question: What is pass by value and pass by reference? |
| |  | | --- | | Answer: | | Pass by value: Passing a copy of the value, not the original  reference.  Pass by reference: Passsing the address of the object, so that  you can access the original object. | |
| Question: Can we declare main method as private? |
| |  | | --- | | Answer: | | Yes, we can declare main method as private. It compiles without  any errors, but in runtime, it says main method is not public. | |

|  |
| --- |
| Question: Can we declare main method as private? |
| |  | | --- | | Answer: | | Yes, we can declare main method as private. It compiles without  any errors, but in runtime, it says main method is not public. | |

|  |
| --- |
| Question: Can non-static member classes (Local classes) have static members? |
| |  | | --- | | Answer: | | No, non-static member classes cannot have static members. Because,  an instance of a non-static member class or local class must be  created in the context of an instance of the enclosing class. You  can declare constants, means static final variables. | |

|  |
| --- |
| Question: What are the environment variables do we neet to set to run Java? |
| |  | | --- | | Answer: | | We need to set two environment variables those are PATH and CLASSPATH. | |

|  |
| --- |
| Question: Can you serialize static fields of a class? |
| |  | | --- | | Answer: | | Since static fields are not part of object state, they are part of class, serialization ignores the static fields. | |

|  |
| --- |
| Question: What is the difference between declaring a variable and defining a variable? |
| |  | | --- | | Answer: | | When variable declaration we just mention the type of the variable and it's name, it does not have any reference to live object. But defining means combination of declaration and initialization. The examples are as given below:   **Declaration:** List list;  **Defining:** List list = new ArrayList(); | |

|  |
| --- |
| Question: Where can we use serialization? |
| |  | | --- | | Answer: | | Whenever an object has to sent over the network, those objects should be serialized. Also if the state of an object is to be saved, objects need to be serilazed. | |

|  |
| --- |
| Question: What modifiers are allowed for methods in an Interface? |
| |  | | --- | | Answer: | | Only public and abstract modifiers are allowed for methods in an interfaces. | |

|  |
| --- |
| Question: What is the purpose of Runtime and System class? |
| |  | | --- | | Answer: | | The purpose of the Runtime class is to provide access to the Java runtime system. The runtime information like memory availability, invoking the garbage collector, etc.   The purpose of the System class is to provide access to system resources. It contains accessibility to standard input, standart output, error output streams, current time in millis, terminating the application, etc. | |

|  |
| --- |
| Question: Which one is faster? ArrayList or Vector? Why? |
| |  | | --- | | Answer: | | ArrayList is faster than Vector. The reason is synchronization. Vector is synchronized. As we know synchronization reduces the performance. | |

|  |
| --- |
| Question: What is the difference between static synchronized and synchronized methods? |
| |  | | --- | | Answer: | | Static synchronized methods synchronize on the class object. If one thread is executing a static synchronized method, all other threads trying to execute any static synchronized methods will be blocked.  Non-static synchronized methods synchronize on this ie the instance of the class. If one thread is executing a synchronized method, all other threads trying to execute any synchronized methods will be blocked. | |

|  |
| --- |
| Question: What is the order of catch blocks when catching more than one exception? |
| |  | | --- | | Answer: | | When you are handling multiple catch blocks, make sure that you are specifing exception sub classes first, then followed by exception super classes. Otherwise we will get compile time error. | |

|  |
| --- |
| Question: What is the difference between the prefix and postfix forms of the increment(++) operator? |
| |  | | --- | | Answer: | | The prefix form first performs the increment operation and then returns the value of the increment operation. The postfix form first returns the current value of the expression and then performs the increment operation on that value. For example:   int count=1; System.out.println(++count);   displays 2. And   int count=1; System.out.println(count++);   displays 1. | |
| Question: What is hashCode? |
| |  | | --- | | Answer: | | The hashcode of a Java Object is simply a number, it is 32-bit signed int, that allows an object to be managed by a hash-based data structure. We know that hash code is an unique id number allocated to an object by JVM. But actually speaking, Hash code is not an unique number for an object. If two objects are equals then these two objects should return same hash code. So we have to implement hashcode() method of a class in such way that if two objects are equals, ie compared by equal() method of that class, then those two objects must return same hash code. If you are overriding hashCode you need to override equals method also.   |  | | --- | | Question: What is the difference between Hashtable and HashMap? | | |  | | --- | | Answer: | | The basic differences are Hashtable is synchronized and HashMap is not synchronized. Hashtable does not allow null values, and HashMap allows null values. | |  |  | | --- | | Question: What are the restrictions when overriding a method? | | |  | | --- | | Answer: | | Overriding methods must have the same name, parameter list, and same return type. i.e., they must have the exact signature of the method we are going to override, including return type. The overriding method cannot be less visible than the method it overrides. i.e., a public method cannot be override to private. The overriding method may not throw any exceptions that may not be thrown by the overridden method. | | | Question: What is the use of assert keyword? | | | |  | | --- | | Answer: | | Java assertion feature allows developer to put assert statements in Java source code to help unit testing and debugging. Assert keyword validates certain expressions. It replaces the if block effectively and throws an AssertionError on failure. | | |  |  | | --- | | Question: What is adapter class? | | |  | | --- | | Answer: | | An adapter class provides the default implementation of all methods in an event listener interface. Adapter classes are very useful when you want to process only few of the events that are handled by a particular event listener interface. You can define a new class by extending one of the adapter classes and implement only those events relevant to you. | | | Question: What is difference between break, continue and return statements? | | | |  | | --- | | Answer: | | The break statement results in the termination of the loop, it will come out of the loop and stops further iterations. The continue statement stops the current execution of the iteration and proceeds to the next iteration. The return statement takes you out of the method. It stops executing the method and returns from the method execution. | | |  |  | | --- | | Question: What is the difference between while and do-while statements? | | |  | | --- | | Answer: | | The while statement verifies the condition before entering into the loop to see whether the next loop iteration should occur or not. The do-while statement executes the first iteration without checking the condition, it verifies the condition after finishing each iteration. The do-while statement will always execute the body of a loop at least once. | | | Question: When does the compiler provides the default constructor? | | | |  | | --- | | Answer: | | The compiler provides a default constructor if no other constructors are available in the class. In case the class contains parametarized constructors, compiler doesnot provide the default constructor. | | | | Question: What are the differences between C++ and Java. | | | | |  | | --- | | Answer: | | Java doesnot support pointers. Pointers are tricky to use and troublesome.  Java does not support multiple inheritances because it causes more problems than it solves. Instead Java supports multiple interface inheritance, which allows an object to inherit many method signatures from different interfaces with the condition that the inheriting object must implement those inherited methods. The multiple interface inheritance also allows an object to behave polymorphically on those methods.  Java does not include structures or unions.  Java does not support destructors but adds a finalize() method. Finalize methods are invoked by the garbage collector prior to reclaiming the memory occupied by the object, which has the finalize() method. This means you do not know when the objects are going to be finalized. Avoid using finalize() method to release non-memory resources like file handles, sockets, database connections etc because Java has only a finite number of these resources and you do not know when the garbage collection is going to kick in to release these resources through the finalize() method.  All the code in Java program is encapsulated within classes therefore Java does not have global variables or functions.  C++ requires explicit memory management, while Java includes automatic garbage collection. | | | | | Question: What are the advantages of java package. | | | | | |  | | --- | | Answer: | | Java packages helps to resolve naming conflicts when different packages have classes with the same names. This also helps you organize files within your project. For example, java.io package do something related to I/O and java.net package do something to do with network and so on. If we tend to put all .java files into a single package, as the project gets bigger, then it would become a nightmare to manage all your files. | | | | |  |  | | --- | | Question: What is dynamic class loading? | | |  | | --- | | Answer: | | Dynamic loading is a technique for programmatically invoking the functions of a class loader at run time. Let us look at how to load classes dynamically by using ***Class.forName (String className);*** method, it is a static method.  The above static method returns the class object associated with the class name. The string className can be supplied dynamically at run time. Once the class is dynamically loaded the ***class.newInstance ()*** method returns an instance of the loaded class. It is just like creating a class object with no arguments.  A ***ClassNotFoundException*** is thrown when an application tries to load in a class through its class name, but no definition for the class with the specified name could be found. | | | Question: What happens if you do not provide a constructor? | | | |  | | --- | | Answer: | | Java does not actually require an explicit constructor in the class description. If you do not include a constructor, the Java compiler will create a default constructor in the byte code with an empty argument | | | | Question: Difference between shallow cloning and deep cloning of objects? | | | | |  | | --- | | Answer: | | The default behavior of an object’s clone() method automatically yields a shallow copy. So to achieve a deep copy the classes must be edited or adjusted.  ***Shallow copy:***Generally clone method of an object, creates a new instance of the same class and copies all the fields to the new instance and returns it. This is called shallow copy. Object class provides a clone method and provides support for the shallow copy. It returns ‘Object’ as type and you need to explicitly cast back to your original object. Since the Object class has the clone method, you cannot use it in all your classes. The class which you want to be cloned should implement clone method and overwrite it. It should provide its own meaning for copy or to the least it should invoke the super.clone(). Also you have to implement Cloneable marker interface or else you will get CloneNotSupportedException. When you invoke the super.clone() then you are dependent on the Object class’s implementation and what you get is a shallow copy.  ***Deep copy:***When you need a deep copy then you need to implement it yourself. When the copied object contains some other object its references are copied recursively in deep copy. When you implement deep copy be careful as you might fall for cyclic dependencies. If you don’t want to implement deep copy yourselves then you can go for serialization. It does implements deep copy implicitly and gracefully handling cyclic dependencies. | | | | | Question: Can we have interfaces with no defined methods in java? | | | | | |  | | --- | | Answer: | | The interfaces with no defined methods act like markers. They just tell the compiler that the objects of the classes implementing the interfaces with no defined methods need to be treated differently. Marker interfaces are also known as “tag” interfaces. | | | | | | Question: What is the difference between “==” and equals() method? | | | | | | |  | | --- | | Answer: | | The == (double equals) returns true, if the variable reference points to the same object in memory. This is called “shallow comparison”.  The equals() method calls the user implemented equals() method, which compares the object attribute values. The equals() method provides “deep comparison” by checking if two objects are logically equal as opposed to the shallow comparison provided by the operator ==.  If equals() method does not exist in a user supplied class then the inherited Object class's equals() method will be called which evaluates if the references point to the same object in memory. In this case, the object.equals() works just like the "==" operator. | | | | | |  |  | | --- | | Question: How can you create an immutable class in java? | | |  | | --- | | Answer: | | Here are the steps to create immutable class:  Declare the class as final, we can not extend the final class.  *public final class MyTestImmutable { ... }*  Declare all fields as final. Final fields can not be changed once its assigned.  *private final int salary;*  Do not provide any method which can change the state of the object, for example the setter methods which changes the values of the instance variables.  The “this” reference is not allowed to escape during construction from the immutable class and the immutable class should have exclusive access to fields that contain references to mutable objects like arrays, collections and mutable classes like Date etc by:  Declaring the mutable references as private.  Not returning or exposing the mutable references to the caller. | |  |  | | --- | | Question: What are access modifiers in java? | | |  | | --- | | Answer: | | ***public:*** A class or interface may be accessed from outside the package. Constructors, inner classes, methods and field variables may be accessed wherever their class is accessed.  ***protected:*** Accessed by other classes in the same package or any subclasses of same package or different package.  ***private:*** Accessed only within the class in which they are declared.  ***no modifier (default modifier):*** Accessed only with in the class. | | | Question: Can we have private constructor in java? | | | |  | | --- | | Answer: | | Private constructor is used if you do not want other classes to instantiate the object.  Private constructors are used in singleton design pattern, factory method design pattern. | | | | Question: Why do we need generics in java? | | | | |  | | --- | | Answer: | | Code that uses generics has many benefits over non-generic code:  **1) Stronger type checks at compile time:** A Java compiler applies strong type checking to generic code and issues errors if the code violates type safety. Fixing compile-time errors is easier than fixing runtime errors, which can be difficult to find.  **2) Elimination of casts:** If you use generics, then explicit type casting is not required.  **3) Enabling programmers to implement generic algorithms:** By using generics, programmers can implement generic algorithms that work on collections of different types, can be customized, and are type safe and easier to read. | | | | | Question: What is the difference between a product and a project? | | | | | |  | | --- | | Answer: | | A project is an endeavor with a clear definition of what needs to be delivered and the date when it needs to be delivered. Actually, it may seem that a product is a project, but it is not. Because, there is no clear definition of what needs to be delivered and there is no clear definition of the the date when it needs to be delivered.  The product backlog is a collection of all possible ideas and additions to a product. The stories in the product backlog have no particular priority or schedule, although you can sort and organize them in hierarchies in the breakdown view by drag & drop. Thus, if you wish to prioritize your stories but don’t want to create projects or iterations, just use the breakdown view!  Products are developed in projects. | | | | | | Question: How does substring() method works on a string? | | | | | | |  | | --- | | Answer: | | String in java is a sequence of characters. String is more like a utility class which works on that character sequence. This character sequence is maintained as a array called value[], for example  *private final char value[];*  String internally defines two private variables called offset and count to manage the char array. The declarations can be as shown below:  */\*\* The offset is the first index of the storage that is used. \*/ private final int offset;  /\*\* The count is the number of characters in the String. \*/ private final int count;*  Everytime we create a substring from any string object, substring() method assigns the new values of offset and count variables. The internal char array is unchanged. This is a possible source of memory leak if substring() method is used without care. | | | | | | | Question: What is the difference between a Java Library and a framework? | | | | | | | |  | | --- | | Answer: | | A library is a collection of class definitions and its implementations. The main benifits of creating library is simply code reuse. A simple example is one of the other developer written code for sending emails. If you think it is a generic module. Most of the places this code can be reusable. If we can make it a library (jar), we can include this library in our code, and call those methods. The classes and methods normally define specific operations in a domain specific area.  In framework, all the control flow is already defined, and there is a bunch of predefined places that you should fill out with your code. We use framework to develope applications. A framework defines a skeleton where the application defines its own features to fill out the skeleton. In this way, your code will be called by the framework when appropriately. The benefit is that developers do not need to worry about if a design is good or not, but just about implementing domain specific functions. | | | | | | | | |