Best Selling Revised Edition 2013

Cracking The Java Coding Interview.

Easy Standard Beginners To Experts Edition 2013.

For First Time Learner's Approach Guide.

A STEP BY STEP TUTOR GUIDE.

15Th Best Selling Edition

2013 Revised Edition.

- Harry.

Cracking The Java Coding Interview Questions & Answers

‘‘... I am just now beginning to discover the difficulty of expressing one’s ideas on paper. As long as it consists solely of description it is pretty easy; but where reasoning comes into play, to make a proper connection, a clearness & a moderate fluency, is to me, as I have said, a difficulty of which I had no idea ...’’

– Harry (Hariom Choudhary)

Cracking The Java Coding Interview.

Easy Standard Beginners To Experts Edition 2013.

For First Time Learner's Approach Guide.

A STEP BY STEP TUTOR GUIDE.

15Th Best Selling Edition

2013 Revised Edition.

- Harry.

Author's note:

The publisher Amazon Kindle & Create Space & Authors offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact: Hr.EvC.India@gmail.com

About Author:

Harry (Hariom Choudhary), MSC-IT, MBA-IT., is a Head of Computer Science at the NWR Engineering. Harry is the leading authority on C and C++ & C# & Core Java & DSA and a best-selling author whose books have sold more than 1.5 million copies. His acclaimed C and C++ & Java books. He has over 6 years of experience as a software methodologist. His teaching and research interests are in the areas of artificial intelligence, programming languages, practical complexity problems, heuristic search methods, deductive algorithms, and educational and social issues. He originated error analysis in heuristic search methods and deductive algorithms.

Authors Side:

Who Am I?

Yet another guy who dreamt of doing it big somewhere but finally ended up working as a Software Engineer for an IT company. For the time being we can skip the history and geography of my life which I intend to officially release along with my autobiography in the future. I have been writing for more than 4 years now. From what started as a way to kill time turned into a passion soon and now has become an integral part of my life.

After all, as with Java Is a powerful raindrops of technology, Programming isn't inherently good or bad—this determination depends entirely on how it is used. The same hammer can be used to either build a wall or knock it down. –Harry

Preface

Actually Like all other computer languages, the elements of Java do not exist in isolation. Rather, they work together to form the language as a whole. However, this interrelatedness can make it difficult to describe one aspect of Java without involving several others. Often a discussion of one feature implies prior knowledge of another. For this reason, this chapter presents a quick overview of several key features of Java. The material described here will give you a foothold that will allow you to write and understand simple programs. Most of the topics discussed will be examined in greater detail in the remaining chapters of Part 1.

As we know in the past few years document the following fact: The Web has irrevocably recast the face of computing and programmers unwilling to master its environment will be left behind. The preceding is a strong statement. It is also true. More and more, applications must interface to the Web. It no longer matters much what the application is, near universal Web access is dragging, pushing, and coaxing programmers to program for the online world, and Java is the language that many will use to do it. Frankly, fluency in Java is no longer an option for the professional programmer, it is a requirement. This book will help you acquire it.

Aside from being the preeminent language of the Internet, Java is important for another reason: it has altered the course of computer language development. Many of the features first mainstreamed by Java are now finding their way into other languages. For example, the new C# language is strongly influenced by Java. Knowledge of Java opens the door to the latest innovations in programming. Put directly, Java is one of the world's most important computer languages.

A Book for All Programmers To use this book does not require any previous programming experience. However, if you come from a C/C++ background, then you will be able to advance a bit more rapidly. As most readers will know, Java is similar, in form and spirit, to C/C++. Thus, knowledge of those languages helps, but is not necessary. Even if you have never programmed before, you can learn to program in Java using this book.

Contents at a Glance

Cracking the

JAVA^{TM}

PROGRAMMING *Coding Interview.*

BRIEF EDITION 2013

Fifteenth Revised Easy Beginners Edition Harry (Hariom Choudhary)
Dream Catchers Associates.

Story of Beautiful Java Girl (Princess of Technology):

About the time that the details of Java were being worked out, a second, and ultimately more important, factor was emerging that would play a crucial role in the future of Java. This second force was, of course, the World Wide Web. Had the Web not taken shape at about the same time that Java was being implemented, Java might have remained a useful but obscure language for programming consumer electronics. However, with the emergence of the World Wide Web, Java was propelled to the forefront of computer language design, because the Web, too, demanded portable programs. Most programmers learn early in their careers that portable programs are as elusive as they are desirable. While the quest for a way to create efficient, portable (platformindependent) programs is nearly as old as the discipline of programming itself, it had taken a back seat to other, more pressing problems. Further, because much of the computer world had divided itself into the three competing camps of Intel, Macintosh, and UNIX, most programmers stayed within their fortified boundaries, and the urgent need for portable code was reduced. However, with the advent of the Internet and the Web, the old problem of portability returned with a vengeance. After all, the Internet consists of a diverse, distributed universe populated with many types of computers, operating systems, and CPUs.

Even though many types of platforms are attached to the Internet, users would like them all to be able to run the same program. What was once an irritating but low-priority problem had become a high-profile necessity. By 1993, it became obvious to members of the Java design team that the problems of portability frequently encountered when creating code for embedded controllers are also found when attempting to create code for the Internet. In fact, the same problem that Java was initially designed to solve on a small scale could also be applied to the Internet on a large scale. This realization caused the focus of Java to switch from consumer electronics to Internet programming. So, while the desire for an architecture neutral programming language provided the initial spark, the Internet ultimately led to Java's large-scale success.

As mentioned earlier, Java derives much of its character from C and C++. This is by intent. The Java designers knew that using the familiar syntax of C and echoing the objectoriented features of C++ would make their language appealing to the legions of experienced C/C++ programmers. In addition to the surface similarities, Java shares some of the other attributes that helped make C and C++ successful. First, Java was designed, tested, and refined by real, working programmers. It is a language grounded in the needs and experiences of the people who devised it.

Thus, Java is also a programmer's language. Second, Java is cohesive and logically consistent. Third, except for those constraints imposed by the Internet environment, Java gives you, the programmer, and full control. If you program well, your programs reflect it. If you program poorly, your programs reflect that, too. Put differently, Java is not a language with training wheels. It is a language for professional programmers.

Before Interview Session:
Introduction:

Java is an powerful object oriented programming language developed by Sun Microsystems Inc. in 1991. Java was developed for consumer electronic devices but later it was shifted towards Internet. Now Java has become the widely used programming language for the Internet. Java is a platform neutral language (Machine Independent). Program developed by Java can run on any hardware or on any operating system in this world.

When the chronicle of computer languages is written, the following will be said: B led to C, C evolved into C++, and C++ set the stage for Java. To understand Java is to understand the reasons that drove its creation, the forces that shaped it, and the legacy that it inherits. Like the successful computer languages that came before, Java is a blend of the best elements of its rich heritage combined with the innovative concepts required by its unique environment. While the remaining chapters of this book describe the practical aspects of Java—including its syntax, libraries, and applications—in this chapter, you will learn how and why Java came about, and what makes it so important. Although Java has become inseparably linked with the online environment of the Internet, it is important to remember that Java is first and foremost a programming language. Computer language innovation and development occurs for two fundamental reasons:

■ To adapt to changing environments and uses
■ To implement refinements and improvements in the art of programming
As you will see, the creation of Java was driven by both elements in nearly equal measure.

History of Java & The Creation of Java:

Basically Java was Developed by James Gosling, Patrick Naughton, ChrisWarth, Ed Frank, and Mike Sheridan at Sun Microsystems, Inc. in 1991. Initial name of this language was "Oak" but it was renamed in 1995 as "Java". It took 18 months to develop the first working version. This language was initially called "Oak" but was renamed "Java" in 1995. Between the initial implementation of Oak in the fall of 1992 and the public announcement of Java in the spring of 1995, many more people contributed to the design and evolution of the language. Bill Joy, Arthur van Hoff, Jonathan Payne, Frank Yellin, and Tim Lindholm were key contributors to the maturing of the original prototype.

Somewhat surprisingly, the original impetus for Java was not the Internet! Instead, the primary motivation was the need for a platformindependent (that is, architecture-neutral) language that could be used to create software to be embedded in various consumer electronic devices, such as microwave ovens and remote controls. As you can probably guess, many different types of CPUs are used as controllers. The trouble with C and C++ (and most other languages) is that they are designed to be compiled for a specific target. Although it is possible to compile a C++ program for just about any type of CPU, to do so requires a full C++ compiler targeted for that CPU. The problem is that compilers are expensive and time-consuming to create. An easier and more cost-efficient—solution was needed. In an attempt to find such a solution, Gosling and others began work on a

portable, platformindependent language that could be used to produce code that would run on a variety of CPUs under differing environments. This effort ultimately led to the creation of Java.

The Stage Is Set for Java:

By the end of the 1980s and the early 1990s, objectoriented programming using C++ took hold. Indeed, for a brief moment it seemed as if programmers had finally found the perfect language. Because C++ blended the high efficiency and stylistic elements of C with the objectoriented paradigm, it was a language that could be used to create a wide range of programs. However, just as in the past, forces were brewing that would, once again, drive computer language evolution forward. Within a few years, the World Wide Web and the Internet would reach critical mass. This event would precipitate another revolution in programming.

Need for Java:

Java was developed due to the need for a platform neutral language that could be used to create software to be embedded in various consumer electronic devices, such as microwave ovens and remote controls. The program written in C and C++ are compiled for a particular piece of hardware and software and that program will not run on any other hardware or software. So we need C/C++ compilers one for each type of hardware to compile a single program. But compilers are expensive and time-consuming to create. So there is a need for platform neutral language. So that program compiled from that compiler can run on any hardware. This need led to the creation of Java.

Java Class Libraries

Java programs consist of pieces called classes. Classes consist of pieces called methods that perform tasks and return information when they complete their tasks. You can program each piece you may need to form a Java program. However, most Java programmers take advantage of rich collections of existing classes in Java class libraries. The class libraries are also known as the Java APIs (Application Programming Interfaces). Thus, there are really two pieces to learning the Java "world." The first is learning the Java language itself so that you can program your own classes; the second is learning how to use the classes in the extensive Java class libraries. Throughout the book, we discuss many library classes. Class libraries are provided primarily by compiler vendors, but many class libraries are supplied by independent software vendors (ISVs). Also, many class libraries are available from the Internet and World Wide Web as freeware or shareware. You can download free ware products and use them for free—subject to any restrictions specified by the copyright owner. You also can download shareware products for free, so you can try the software. Shareware products often are free of charge for personal use. However, for shareware products that you use regularly or use for commercial purposes, you are expected to pay a fee designated by the copyright owner. Many freeware and shareware products are also open source. The source code for open-source products is freely available on the Internet, which enables you to learn from the source code, validate that the code serves its stated purpose and even modify the code. Often, open-source products require that you publish any enhancements you make so the open-source community can continue to evolve those products. One example of a popular open-source product is the Linux operating system.

Basics of a Typical Java Environment

Java systems generally consist of several parts: An environment, the language, the Java Applications Programming Interface (API) and various class libraries. The following discussion explains a typical Java program development environment, Java programs normally go through five phases to be executed (Fig. 1.1). These are: edit, compile, load, verify and execute. We discuss these concepts in the context of the Java 2 Software Development Kit (J2SDK) that is included on the CD that accompanies this book. Carefully follow the installation instructions for the J2SDK provided on the CD to ensure that you set up your computer properly to compile and execute Java programs. [Note: If you are not using UNIX/Linux, Windows 95/98/ME or Windows NT/2000, refer to the manuals for your system's Java environment or ask your instructor how to accomplish these tasks in your environment (which will probably be similar to the environment, Phase 1 consists of editing a file. This is accomplished with an editor program (normally known as an editor). The programmer types a Java program, using the editor, and makes corrections, if necessary. When the programmer specifies that the file in the editor should be saved, the program is stored on a secondary storage device, such as a disk. Java program file names end with the .java extension. Two editors widely used on UNIX/Linux systems are vi and emacs. On Windows 95/98/ME and Windows NT/2000, simple edit programs like the DOS Edit command and the Windows Notepad will suffice. Java integrated development environments (IDEs), such as Forte for Java Community Edition, NetBeans, Borland's JBuilder, Symantec's Visual Cafe and IBM's Visual Age have built in editors that are integrated into the programming environment. We assume the reader knows how to edit a file.

Languages such as Java are objectoriented—programming in such programming (OOP) and allows designers to implement the object a language is called objectoriented

oriented design as a working system. Languages such as C, on the other hand, are procedural programming languages, so programming tends to be action-oriented. In C, the unit of programming is the function. In Java, the unit of programming is the class from which objects are eventually instantiated (a fancy term for "created"). Java classes contain methods (that

implement class behaviors) and attributes (that implement class data).

C programmers concentrate on writing functions. Groups of actions that perform some common task are formed into functions, and functions are grouped to form programs. Data are certainly important in C, but the view is that data exist primarily in support of the actions that functions perform. The verbs in a system specification help the C programmer determine the set of functions needed to implement that system. Java programmers concentrate on creating their own user-defined types called classes and components. Each class contains data and the set of functions that manipulate that data. The data components of a Java class are called attributes. The function components of a Java class are called methods. Just as an instance of a builtin type such as int is called a variable, an instance of a user-defined type (i.e., a class) is called an object. The programmer uses builtin types as the "building blocks" for constructing user-defined types. The focus in Java is on classes (out of which we make objects) rather than on functions. The nouns in a system specification help the Java programmer determine the set of classes from which objects will be created that will work together to implement the system. Classes are to objects as blueprints are to houses.

We can build many houses from one blueprint, and we can instantiate many objects from one class. Classes can also have relationships with other classes. For example, in an objectoriented design of a bank, the "bank teller" class needs to relate to the "customer" class. These relationships are called associations.

We will see that, when software is packaged as classes, these classes can be reused in future software systems. Groups of related classes are often packaged as reusable components. Just as real-estate brokers tell their clients that the three most important factors affecting the price of real estate are "location, location and location," many people in the software community believe that the three most important factors affecting the future of software development are "reuse, reuse and reuse."

Indeed, with object technology, we can build much of the software we will need by combining "standardized, interchangeable parts" called classes. This book teaches you how to "craft valuable classes" for reuse. Each new class you create will have the potential to become a valuable software asset that you and other programmers can use to speed and enhance the quality of future software-development efforts—an exciting possibility.

Relation of Java with C, C++, & C#:

From C Java derives its syntax and from C++ it derives object oriented features. It is not an enhanced version of C++. Java is neither upwardly nor downwardly compatible with C++. Java was not designed to replace C++. C# another language developed by Microsoft to support the .NET Framework, C# is closely related to Java because both share C++ & C style syntax, support distributed programming, and utilize the same object model.

Primary Objective of Java is to achieve:

1. **Security:** There is no threat of virus infection when we use Java compatible Web Browser. Also there is no threat of malicious programs that can gather private information, such as credit card numbers, bank account balances and passwords from local machine. Java provides a firewall between a networked application and our computer.

2. **Portability:** Java programs are portable from one computer to another computer running different types of operating systems and having different hardware.

Java Bytecode:

The output of a Java compiler is bytecode not the machine code (".class" file). Bytecode is a highly optimized set of instructions designed to be executed by the Java runtime system, which is called as JVM (Java Virtual Machine). JVM is the interpreter which interprets the bytecode. Compiled program runs faster but still Java uses interpreter to achieve portability so Java programs runs a little slower. Now a program compiled through a Java compiler can run in any environment but JVM needs to be implemented for each platform. Java programs are interpreted. This also helps to make it secure because the execution of every Java program is under the control of JVM.

JIT (Just In Time):

JIT is a translator used by JVM to translate bytecode into actual machine code. It does not translate entire bytecodes rather it translates piece by piece on demand basis.

Interview Part 1

Learning Programming Strategies:

A programming course is quite different from other courses. In a programming course, you learn from examples, from practice, and from mistakes. You need to devote a lot of time to writing programs, testing them, and fixing errors.

For first-time programmers, learning Java is like learning any high-level programming language. The fundamental point is to develop the critical skills of formulating programmatic solutions for real problems and translating them into programs using selection statements, loops, methods, and arrays.

Once you acquire the basic skills of writing programs using loops, methods, and arrays, you can begin to learn how to develop large programs and GUI programs using the object oriented approach.

When you know how to program and you understand the concept of objectoriented programming, learning Java becomes a matter of learning the Java API. The Java API establishes a framework for programmers to develop applications using Java. You have to use the classes and interfaces in the API and follow their conventions and rules to create applications. The best way to learn the Java API is to imitate examples and do exercises.

The Java Language Specification, API, JDK, and IDE

Computer languages have strict rules of usage. If you do not follow the rules when writing a program, the computer will be unable to understand it. The Java language specification and Java API define the Java standard.

The Java language specification is a technical definition of the language that includes the syntax and semantics of the Java programming language. The complete Java language specification can be found at java.sun.com/docs/books/jls.

The application program interface (API) contains predefined classes and interfaces for developing Java programs. The Java language specification is stable, but the API is still expanding. At the Sun Java Website (java.sun.com), you can view and download the latest version of the Java

(java.sun.com), you can view and download the latest version of the Java API.

Java is a full-fledged and powerful language that can be used in many ways. It comes in three editions: Java Standard Edition(Java SE),Java Enterprise Edition(Java EE), and Java Micro Edition(Java ME). Java SE can be used to develop client-side standalone applications or applets. Java EE can be used to develop server-side applications, such as Java servlets and Java Server Pages. Java ME can be used to develop applications for mobile devices, such as cell phones. This book uses Java SE to introduce Java programming.

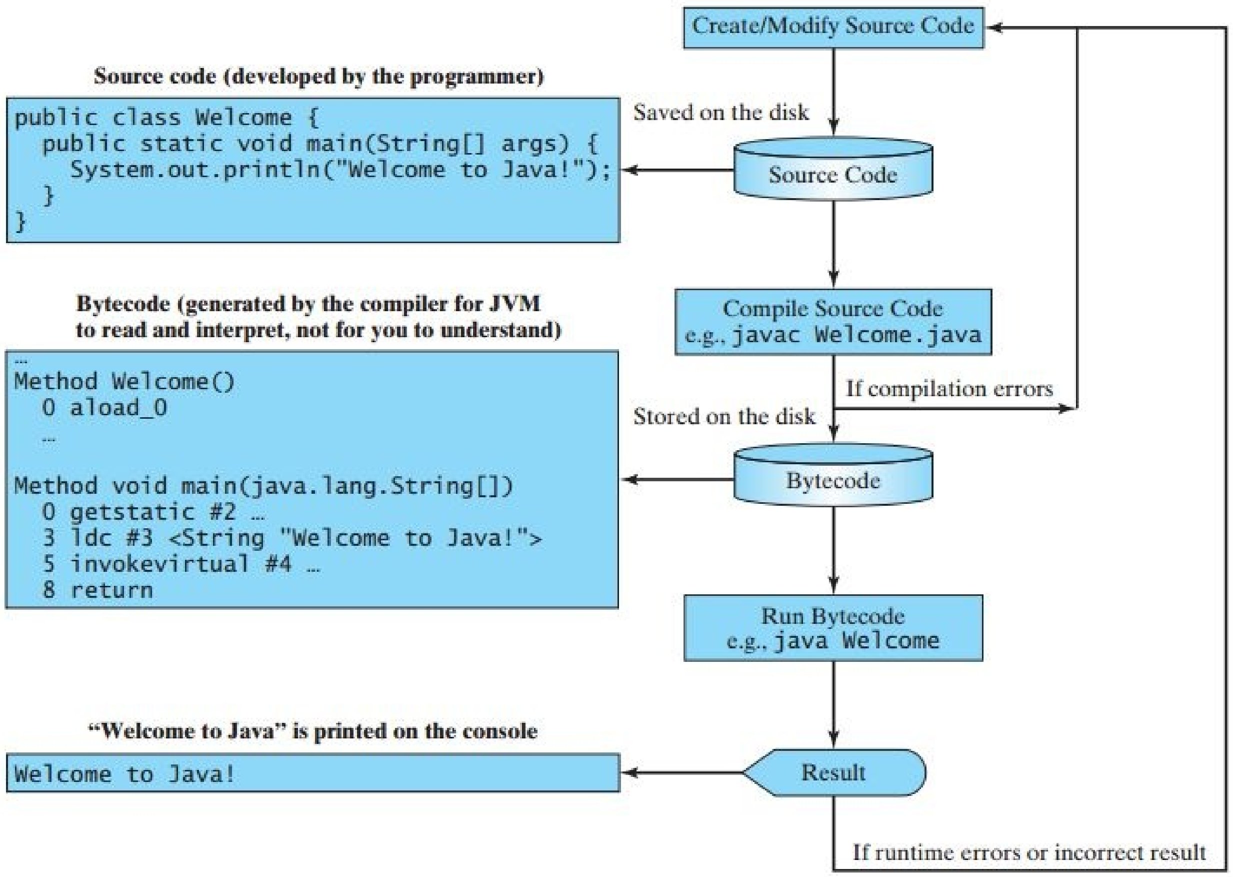
There are many versions of Java SE. The latest, Java SE 6, will be used in this book. Sun releases each version with a Java Development Toolkit(JDK). For Java SE 6, the Java Development Toolkit is called JDK 1.6(also known as Java 6 or JDK 6).

JDK consists of a set of separate programs, each invoked from a command line, for developing and testing Java programs. Besides JDK, you can use a Java development tool (e.g., Net Beans, Eclipse, and TextPad)—software that provides an integrated development environment

(IDE) for rapidly developing Java programs. Editing, compiling, building, debugging, and online help are integrated in one graphical user interface. Just enter source code in one window or open an existing file in a window, then click a button, menu item, or function key to compile and run the program. If you want to learn Java step By Step as you think your Java Programming is weak you can try this book "Core Java Professional" available on Amazon.com authored By Harry.

Creating, Compiling, and Executing a Java Program:

You have to create your program and compile it before it can be executed. This process is repetitive, as shown in Figure 1.9. If your program has compilation errors, you have to modify the program to fix them, and then recompile it. If your program has runtime errors or does not produce the correct result, you have to modify the program, recompile it, and execute it again.


Create/Modify Source Code
Source code (developed by the programmer)
Saved on the disk
Source Code
Bytecode (generated by the compiler for JVM to read and interpret, not for you to understand)
Compile Source Code e.g., javac Welcome.java
If compilation errors
Stored on the disk
Bytecode
Run Bytecode e.g., java Welcome
"Welcome to Java" is printed on the console
Result
If runtime errors or incorrect result

```
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

```
…
Method Welcome()
  0 aload_0
  …

Method void main(java.lang.String[])
  0 getstatic #2 …
  3 ldc #3 <String "Welcome to Java!">
  5 invokevirtual #4 …
  8 return
```

```
Welcome to Java!
```

Question Can we define private and protected modifiers for variables in interfaces? (Core Java)

Answer No.

Question What is the query used to display all tables names in SQL Server (Query analyzer)? (JDBC)

Answer select * from information_schema.tables

Question What is Externalizable? (Core Java)

Answer Externalizable is an Interface that extends Serializable Interface. And sends data into Streams in Compressed Format. It has two methods, writeExternal(ObjectOuput out) and readExternal(ObjectInput in)

Question What modifiers are allowed for methods in an Interface? (Core Java)

Answer Only public and abstract modifiers are allowed for methods in interfaces.

Question What is a local, member and a class variable? (Core Java)

Answer Variables declared within a method are "local" variables. Variables declared within the class i.e not within any methods are "member" variables (global variables). Variables declared within the class i.e not within any methods and are defined as "static" are class variables.

and are defined as "static" are class variables.

Question How many types of JDBC Drivers are present and what are they? (JDBC) Answer There are 4 types of JDBC Drivers Type 1: JDBC-ODBC Bridge Driver Type 2: Native API Partly Java Driver Type 3: Network protocol Driver Type 4: JDBC Net pure Java Driver

Question Can we implement an interface in a JSP? (JSP) AnswerNo
Question What is the difference between ServletContext and PageContext? (JSP) Answer ServletContext: Gives the information about the container PageContext: Gives the information about the Request
Question What is the difference in using request.getRequestDispatcher() and context.getRequestDispatcher()? (JSP)

Answer request.getRequestDispatcher(path): In order to create it we need to give the relative path of the resource context.getRequestDispatcher(path): In order to create it we need to give the absolute path of the resource.

Question How to pass information from JSP to included JSP? (JSP) Answer Using <%jsp:param> tag.
Question What is the difference between directive include and jsp include? (JSP)
Answer <%@ include> : Used to include static resources during translation time.
: Used to include dynamic content or static content during
What are Predefined variables or implicit objects?

To simplify code in JSP expressions and scriptlets, we can use eight automatically defined variables, sometimes called implicit objects. They are request, response, out, session, application, config, pageContext, and page.

What is BDK?
BDK, Bean Development Kit is a tool that enables to create, configure and connect a set of set of Beans and it can be used to test Beans without writing a code.
What is a Jar file?

Jar file allows to efficiently deploying a set of classes and their associated resources. The elements in a jar file are compressed, which makes downloading a Jar file much faster than separately downloading several uncompressed files. The package java. util. zip contains classes that read and write jar files

and write jar files.

Explain the methods, rebind() and lookup() in Naming class?

rebind() of the Naming class(found in java. rmi) is used to update the RMI registry on the server machine. Naming. rebind(”AddSever”, AddServerImpl); lookup() of the Naming class accepts one argument, the rmi URL and returns a reference to an object of type AddServerImpl.

what is UnicastRemoteObject?
All remote objects must extend UnicastRemoteObject, which provides functionality that is needed to make objects available from remote machines.

What is RMI architecture?

RMI architecture consists of four layers and each layer performs specific functions: a) Application layer - contains the actual object definition.
b) Proxy layer - consists of stub and skeleton.
c) Remote Reference layer - gets the stream of bytes from the transport layer and sends it to the proxy layer.
d) Transportation layer - responsible for handling the actual machine-to-machine communication.

What steps are involved in developing an RMI object?

The steps involved in developing an RMI object are:
a) Define the interfaces
b) Implementing these interfaces
c) Compile the interfaces and their implementations with the java compiler
d) Compile the server implementation with RMI compiler
e) Run the RMI registry
f) Run the application
What is RMI?

Remote Method Invocation (RMI) allows java object that executes on one machine and to invoke the method of a Java object to execute on another machine.
How do servlets handle multiple simultaneous requests?

The server has multiple threads that are available to handle requests. When

a request comes in, it is assigned to a thread, which calls a service method (for example: doGet(), doPost() and service()) of the servlet. For this reason, a single servlet object can have its service methods called by many threads at once.

What is Servlet chaining?

Servlet chaining is a technique in which two or more servlets can cooperate in servicing a single request. In servlet chaining, one servlet's output is piped to the next servlet's input. This process continues until the last servlet is reached. Its output is then sent back to the client.

Is it possible to call servlet with parameters in the URL?
Yes. You can call a servlet with parameters in the syntax as (?Param1 = xxx || m2 = yyy).
Why should we go for interservlet communication?

Servlets running together in the same server communicate with each other in several ways. The three major reasons to use interservlet communication are:
a) Direct servlet manipulation - allows to gain access to the other currently loaded servlets and perform certain tasks (through the ServletContext object)
b) Servlet reuse - allows the servlet to reuse the public methods of another servlet. c) Servlet collaboration - requires to communicate with each other by sharing specific information (through method invocation)

What is connection pooling?

With servlets, opening a database connection is a major bottleneck because we are creating and tearing down a new connection for every page request and the time taken to create connection will be more. Creating a connection pool is an ideal approach for a complicated servlet. With a connection pool, we can duplicate only the resources we need to duplicate rather than the entire servlet. A connection pool can also intelligently manage the size of the pool and make sure each connection remains valid. A number of connection pool packages are currently available. Some like DbConnectionBroker are freely available from Java Exchange Works by creating an object that dispenses connections and connection Ids on request. The ConnectionPool class maintains a Hastable, using Connection objects as keys and Boolean

values as stored values. The Boolean value indicates whether a connection is in use or not. A program calls getConnection() method of the ConnectionPool for getting Connection object it can use; it calls returnConnection() to give the connection back to the pool.

Is it possible to communicate from an applet to servlet and how many ways and how?

Yes, there are three ways to communicate from an applet to servlet and they are: a) HTTP Communication(Text-based and object-based)
b) Socket Communication
c) RMI Communication

What are cookies and how will you use them?

Cookies are a mechanism that a servlet uses to have a client hold a small amount of stateinformation associated with the user.
a) Create a cookie with the Cookie constructor: public Cookie(String name, String value) b) A servlet can send a cookie to the client by passing a Cookie object to the addCookie() method of HttpServletResponse: public void HttpServletResponse. addCookie(Cookie cookie) c) A servlet retrieves cookies by calling the getCookies() method of HttpServletRequest: public Cookie[ ] HttpServletRequest. getCookie().

What is Server-Side Includes (SSI)?

Server-Side Includes allows embedding servlets within HTML pages using a special servlet tag. In many servlets that support servlets, a page can be processed by the server to include output from servlets at certain points inside the HTML page. This is accomplished using a special internal SSINCLUDE, which processes the servlet tags. SSINCLUDE servlet will be invoked whenever a file with an. shtml extension is requested. So HTML files that include server-side includes must be stored with an . shtml extension.
What is session tracking and how do you track a user session in servlets?

Session tracking is a mechanism that servlets use to maintain state about a series requests from the same user across some period of time. The methods used for session tracking are: a) User Authentication - occurs when a web server restricts access to some of its resources to only those clients that log

in using a recognized username and password.
b) Hidden form fields - fields are added to an HTML form that are not displayed in the client's browser. When the form containing the fields is submitted, the fields are sent back to the server.
c) URL rewriting - every URL that the user clicks on is dynamically modified or rewritten to include extra information. The extra information can be in the form of extra path information, added parameters or some custom, server-specific URL change. d) Cookies - a bit of information that is sent by a web server to a browser and which can later be read back from that browser.
e) HttpSession-places a limit on the number of sessions that can exist in memory. This limit is set in the session. maxresidents property.

How many ways can we track client and what are they?
The servlet API provides two ways to track client state and they are: a) Using Session tracking and b) Using Cookies.
What is the difference between doPost and doGet methods?
a) doGet() method is used to get information, while doPost() method is used for posting information.
b) doGet() requests can't send large amount of information and is limited to 240-255 characters. However, doPost()requests passes all of its data, of unlimited length.

c) A doGet() request is appended to the request URL in a query string and this allows the exchange is visible to the client, whereas a doPost() request passes directly over the socket connection as part of its HTTP request body and the exchange are invisible to the client. How to create and call stored procedures?

To create stored procedures:
Create procedure procedurename (specify in, out and in out parameters) BEGIN Any multiple SQL statement; END;

To call stored procedures: CallableStatement csmt = con. prepareCall(",call procedure name(?,?)-"); csmt. registerOutParameter(column no. , data type); csmt. setInt(column no. , column name) csmt. execute();

What is stored procedure?

Stored procedure is a group of SQL statements that forms a logical unit and

Stored procedure is a group of SQL statements that forms a logical unit and performs a particular task. Stored Procedures are used to encapsulate a set of operations or queries to execute on database. Stored procedures can be compiled and executed with different parameters and results and may have any combination of input/output parameters.

What are the types of statements in JDBC?

Statement: to be used createStatement() method for executing single SQL statement PreparedStatement — To be used preparedStatement() method for executing same SQL statement over and over. CallableStatement — To be used prepareCall() method for multiple SQL statements over and over.

What are the steps involved for making a connection with a database or how do you connect to a database?

a) Loading the driver : To load the driver, Class. forName() method is used. Class. forName(”sun. jdbc. odbc. JdbcOdbcDriver”); When the driver is loaded, it registers itself with the java. sql. DriverManager class as an available database driver.

b) Making a connection with database: To open a connection to a given database, DriverManager. getConnection() method is used. Connection con = DriverManager. getConnection (”jdbc:odbc:somedb”, “user”, “password”);

c) Executing SQL statements : To execute a SQL query, java. sql. statements class is used. createStatement() method of Connection to obtain a new Statement object. Statement stmt = con. createStatement(); A query that returns data can be executed using the executeQuery() method of Statement. This method executes the statement and returns a java. sql. ResultSet that encapsulates the retrieved data: ResultSet rs = stmt. executeQuery(”SELECT * FROM some table”);

d) Process the results : ResultSet returns one row at a time. Next() method of ResultSet object can be called to move to the next row. The getString() and getObject() methods are used for retrieving column values: while(rs. next()) , String event = rs. getString(”event”); Object count = (Integer) rs. getObject(”count”);

What are the types of JDBC Driver Models and explain them?

There are two types of JDBC Driver Models and they are: a) Two tier model and
b) Three tier model

Two tier model: In this model, Java applications interact directly with the database. A JDBC driver is required to communicate with the particular database management system that is being accessed. SQL statements are sent to the database and the results are given to user. This model is referred to as client/server configuration where user is the client and the machine that has the database is called as the server. Three tier model: A middle tier is introduced in this model. The functions of this model are: a) Collection of SQL statements from the client and handing it over to the database, b) Receiving results from database to the client and c) Maintaining control over accessing and updating of the above.

What is the difference between JDBC and ODBC?

a) OBDC is for Microsoft and JDBC is for Java applications.
b) ODBC can’t be directly used with Java because it uses a C interface.
c) ODBC makes use of pointers which have been removed totally from Java.
d) ODBC mixes simple and advanced features together and has complex options for simple queries. But JDBC is designed to keep things simple while allowing advanced capabilities when required.
e) ODBC requires manual installation of the ODBC driver manager and driver on all client machines. JDBC drivers are written in Java and JDBC code is automatically installable, secure, and portable on all platforms.
f) JDBC API is a natural Java interface and is built on ODBC. JDBC retains some of the basic features of ODBC.
What is JDBC?

JDBC is a set of Java API for executing SQL statements. This API consists of a set of classes and interfaces to enable programs to write pure Java Database applications.
What is serialization and deserialization?
Serialization is the process of writing the state of an object to a byte stream.
Deserialization is the process of restoring these objects.
What is the difference between Reader/Writer and InputStream/Output Stream?
The Reader/Writer class is character-oriented and the

The Reader/Writer class is character-oriented and the InputStream/OutputStream class is byte-oriented.
What is a stream and what are the types of Streams and classes of the Streams?

A Stream is an abstraction that either produces or consumes information. There are two types of Streams and they are:
Byte Streams: Provide a convenient means for handling input and output of bytes. Character Streams: Provide a convenient means for handling input & output of characters.

Byte Streams classes: Are defined by using two abstract classes, namely InputStream and OutputStream.
Character Streams classes: Are defined by using two abstract classes, namely Reader and Writer.

What are Vector, Hashtable, LinkedList and Enumeration?

Vector : The Vector class provides the capability to implement a growable array of objects. Hashtable : The Hashtable class implements a Hashtable data structure. A Hashtable indexes and stores objects in a dictionary using hash codes as the object's keys. Hash codes are integer values that identify objects.
LinkedList: Removing or inserting elements in the middle of an array can be done using LinkedList. A LinkedList stores each object in a separate link whereas an array stores object references in consecutive locations.
Enumeration: An object that implements the Enumeration interface generates a series of elements, one at a time. It has two methods, namely hasMoreElements() and nextElement(). HasMoreElemnts() tests if this enumeration has more elements and nextElement method returns successive elements of the series.
How are the elements of different layouts organized?

FlowLayout: The elements of a FlowLayout are organized in a top to bottom, left to right fashion.
BorderLayout: The elements of a BorderLayout are organized at the borders (North, South, East and West) and the center of a container.
CardLayout: The elements of a CardLayout are stacked, on top of the other, like a deck of cards.
GridLayout: The elements of a GridLayout are of equal size and are laid out using the square of a grid.

out using the square of a grid.
GridBagLayout: The elements of a GridBagLayout are organized according to a grid. However, the elements are of different size and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.

What is a layout manager and what are different types of layout managers available in java AWT?

A layout manager is an object that is used to organize components in a container. The different layouts are available are FlowLayout, BorderLayout, CardLayout, GridLayout and GridBagLayout.

What is the difference between scrollbar and scrollpane?
A Scrollbar is a Component, but not a Container whereas Scrollpane is a Conatiner and handles its own events and perform its own scrolling.
What is source and listener

Source : A source is an object that generates an event. This occurs when the internal state of that object changes in some way.
Listener : A listener is an object that is notified when an event occurs. It has two major requirements. First, it must have been registered with one or more sources to receive notifications about specific types of events. Second, it must implement methods to receive and process these notifications.

How do you set security in applets?
using setSecurityManager() method. What is the lifecycle of an applet?

init() method - Can be called when an applet is first loaded start() method - Can be called each time an applet is started. paint() method - Can be called when the applet is minimized or maximized. stop() method - Can be used when the browser moves off the applet’s page. destroy() method - Can be called when the browser is finished with the applet.

When do you use codebase in applet?
When the applet class file is not in the same directory, codebase is used.
How does applet recognize the height and width?
Using getParameters() method.
Are there any global variables in Java, which can be accessed by other part of your program?
No, it is not the main method in which you define variables. Global

No, it is not the main method in which you define variables. Global variables is not possible because concept of encapsulation is eliminated here.
What is daemon thread and which method is used to create the daemon thread?

Daemon thread is a low priority thread which runs intermittently in the back ground doing the garbage collection operation for the java runtime system. setDaemon method is used to create a daemon thread.

When you will synchronize a piece of your code?
When you expect your code will be accessed by different threads and these threads may change a particular data causing data corruption.
What is synchronization?

Synchronization is the mechanism that ensures that only one thread is accessed the resources at a time.
What is the class and interface in java to create thread and which is the most advantageous method?

Thread class and Runnable interface can be used to create threads and using Runnable interface is the most advantageous method to create threads because we need not extend thread class here.

What are the methods for inter-thread communication and what is the class in which these methods are defined?

wait (), notify () and notifyAll() methods can be used for inter-thread communication and these methods are in Object class. wait() : When a thread executes a call to wait() method, it surrenders the object lock and enters into a waiting state. notify() or notifyAll() : To remove a thread from the waiting state, some other thread must make a call to notify() or notifyAll() method on the same object.

What is multithreading?
Multithreading is the mechanism in which more than one thread run independent of each other within the process.
What is the difference between process and thread?
Process is a program in execution whereas thread is a separate path of execution in a program.
What is the difference between Array and vector?
Array is a set of related data type and static whereas vector is a growable

Array is a set of related data type and static whereas vector is a growable array of objects and dynamic.

What is the difference between String and String Buffer?
a) String objects are constants and immutable whereas StringBuffer objects are not. b) String class supports constant strings whereas StringBuffer class supports growable and modifiable strings.

Can you have an inner class inside a method and what variables can you access?
Yes, we can have an inner class inside a method and final variables can be accessed. What is a cloneable interface and how many methods does it contain?
It is not having any method because it is a TAGGED or MARKER interface.
What is the difference between Integer and int?

a) Integer is a class defined in the java.lang package, whereas int is a primitive data type defined in the Java language itself. Java does not automatically convert from one to the other.
b) Integer can be used as an argument for a method that requires an object, whereas int can be used for calculations.

What are inner class and anonymous class?

Inner class : classes defined in other classes, including those defined in methods are called inner classes. An inner class can have any accessibility including private. Anonymous class : Anonymous class is a class defined inside a method without a name and is instantiated and declared in the same place and cannot have explicit constructors.

What modifiers may be used with toplevel class?
public, abstract and final can be used for toplevel class.
What is meant by Preinitialization of Servlet?

When servlet container is loaded, all the servlets defined in the web.xml file does not initialized by default. But the container receives the request it loads the servlet. But in some cases if you want your servlet to be initialized when context is loaded, you have to use a concept called preinitialization of Servlet. In case of Preinitialization, the servlet is loaded when context is loaded. You can specify 1

loaded. You can specify 1
in between the tag.

What is ServletContext?

ServletContext is an Interface that defines a set of methods that a servlet uses to communicate with its servlet container, for example, to get the MIME type of a file, dispatch requests, or write to a log file. There is one context per “web application” per Java Virtual Machine. (A “web application” is a collection of servlets and content installed under a specific subset of the server’s URL namespace such as /catalog and possibly installed via a .war file.) What is a Servlet?

Java Servlets are server side components that provides a powerful mechanism for developing server side of web application. Earlier CGI was developed to provide server side capabilities to the web applications. Although CGI played a major role in the explosion of the Internet, its performance, scalability and reusability issues make it less than optimal solutions. Java Servlets changes all that. Built from ground up using Sun’s write once run anywhere technology java servlets provide excellent framework for server side processing.

Where the CardLayout is used ?

CardLayout manages two or more components that share the same display space. It lets you use one container (usually a panel) to display one out of many possible component children (like flipping cards on a table). A program can use this layout to show a different child component to different users. For example, the interface shown to an administrator might have additional functionality from the interface shown to a regular user. With card layout, our program can show the appropriate interface depending on the type of user using the program. Another typical use of card layout would be to let end user toggle among different displays and choose the one they prefer. In this case, the program must provide a GUI for the user to make the selection.

What is JFC ?
Java Foundation Classes include:
Standard AWT 1.1
Accessibility interface

Lightweight components: which are user interface components that do not subclass an existing AWT interface element. They do not use native interface elements as provided by the underlying windowing system. This means that they are less limiting than standard AWT components.

Java look and feel
Support for native look and feel
Services such as Java2D and Drag and Drop How do you communicate in between Applets & Servlets ?

We can use the java.net.URLConnection and java.net.URL classes to open a standard HTTP connection and “tunnel” to the web server. The server then passes this information to the servlet in the normal way. Basically, the applet pretends to be a web browser, and the servlet doesn’t know the difference. As far as the servlet is concerned, the applet is just another HTTP client.

How will you communicate between two Applets ?

The simplest method is to use the static variables of a shared class since there’s only one instance of the class and hence only one copy of its static variables. A slightly more reliable method relies on the fact that all the applets on a given page share the same AppletContext. We obtain this applet context as follows:
<font><font size="2" face="Verdana, Arial"> AppletContext ac = getAppletContext(); </font></font>
AppletContext provides applets with methods such as getApplet(name), getApplets(),getAudioClip, getImage, showDocument and showStatus().

When is update method called ?

Whenever a screen needs redrawing (e.g., upon creation, resizing, validating) the update method is called. By default, the update method clears the screen and then calls the paint method, which normally contains all the drawing code.

What is the order of method invocation in an Applet ? ***public void init()*** **: Initialization method called once by browser.**

public void start() **: Method called after init() and contains code to start**

processing. If the user leaves the page and returns without killing the current browser session, the start () method is called without being preceded by init ().

***public void stop()* : Stops all processing started by start (). Done if user moves off page.**
***public void destroy()* : Called if current browser session is being terminated. Frees all resources used by applet.**
What's the difference between notify() and notifyAll()?

notify() is used to unblock one waiting thread; notifyAll() is used to unblock all of them. Using notify() is preferable (for efficiency) when only one blocked thread can benefit from the change (for example, when freeing a buffer back into a pool). notifyAll() is necessary (for correctness) if multiple threads should resume (for example, when releasing a "writer"• lock on a file might permit all "readers"• to resume).

What is meant by time slicing?

Its a task scheduling method. With time slicing, or "Round-Robin Systems", several processes are executed sequentially to completion. Each executable task is assigned a fixed-time quantum called a time slice in which to execute.

How does thread synchronization occurs inside a monitor ?

The JVM uses locks in conjunction with monitors. A monitor is basically a guardian in that it watches over a sequence of code, making sure only one thread at a time executes the code. Each monitor is associated with an object reference. When a thread arrives at the first instruction in a block of code it must obtain a lock on the referenced object. The thread is not allowed to execute the code until it obtains the lock. Once it has obtained the lock, the thread enters the block of protected code. When the thread leaves the block, no matter how it leaves the block, it releases the lock on the associated object.

What is Model 1?

Using JSP technology alone to develop Web page. Such term is used in the earlier JSP specification. Model 1 architecture is suitable for applications that have very simple page flow, have little need for centralized security

control or logging, and change little over time. Model 1 applications can often be refactored to Model 2 when application requirements change.

What is Model 2?
Using JSP and Servlet together to develop Web page. Model 2 applications are easier to maintain and extend, because views do not refer to each other directly.
What will be the default values of all the elements of an array defined as an instance variable?

If the array is an array of primitive types, then all the elements of the array will be initialized to the default value corresponding to that primitive type. *e.g.* All the elements of an array of int will be initialized to 0, while that of boolean type will be initialized to false. Whereas if the array is an array of references (of any type), all the elements will be initialized to null. When a thread blocks on I/O, what state does it enter?

A thread enters the waiting state when it blocks on I/O.
Why do threads block on I/O ?
Threads block on i/o (that is enters the waiting state) so that other threads may execute while the i/o Operation is performed.
What is the difference between JSP and JSF?
JSP simply provides a Page which may contain markup, embedded Java code, and “tags” which encapsulate more complicated logic /html.

JSF may use JSP as its template, but provides much more. This includes validation, rich component model and life cycle, more sophisticated EL, separation of data, navigation handling, different view technologies (instead of JSP), ability to provide more advanced features such as AJAX, *etc.*

What is JSF?

JavaServer Faces (JSF) is a new standard Java framework for building Web applications. It simplifies development by providing a component-centric approach to developing Java Web user interfaces. JavaServer Faces also appeals to a diverse audience of Java/Web developers. “Corporate developers” and Web designers will find that JSF development can be as simple as dragging and dropping user interface (UI) components onto a page, while “systems developers” will find that the rich and robust JSF API offers them unsurpassed power and programming flexibility. JSF also

ensures that applications are well designed with greater maintainability by integrating the well established Model-View-Controller (MVC) design pattern into it's architecture. Finally, since JSF is a Java standard developed through Java Community Process (JCP), development tools vendors are fully empowered to provide easy to use, visual, and productive develop environments for JavaServer Faces.

What is the difference between superclass and subclass?
A super class is a class that is inherited whereas sub class is a class that does the inheriting. What is difference between overloading and overriding?

a) In overloading, there is a relationship between methods available in the same class whereas in overriding, there is relationship between a superclass method and subclass method.

b) Overloading does not block inheritance from the superclass whereas overriding blocks inheritance from the superclass.
c) In overloading, separate methods share the same name whereas in overriding, subclass method replaces the superclass.
d) Overloading must have different method signatures whereas overriding must have same signature.
What is method overloading and method overriding?
Method Overloading: When a method in a class having the same method name with different arguments is said to be method overloading.
Method Overriding: When a method in a class having the same method name with same arguments is said to be method overriding.
What is finalize() method?
finalize() method is used just before an object is destroyed and can be called just prior to garbage collection.
What are Unicode characters?
Unicode is used for internal representation of characters and strings and it uses 16 bits to represent each other.
What is final, finalize() and finally?

final : final keyword can be used for class, method and variables. A final class cannot be subclassed and it prevents other programmers from subclassing a secure class to invoke insecure methods. A final method can't be overridden. A final variable can't change from its initialized value.
finalize() : finalize() method is used just before an object is destroyed and can be called just prior to garbage collection.

can be called just prior to garbage collection.

finally : finally, a key word used in exception handling, creates a block of code that will be executed after a try/catch block has completed and before the code following the try/catch block. The finally block will execute whether or not an exception is thrown. For example, if a method opens a file upon exit, then you will not want the code that closes the file to be bypassed by the exception-handling mechanism. This finally keyword is designed to address this contingency.

What are different types of access modifiers in Java?
Public: Any thing declared as public can be accessed from anywhere.
Private: Any thing declared as private can't be seen outside of its class.
Protected: Any thing declared as protected can be accessed by classes in the same package and subclasses in the other packages.
Default Modifier: Can be accessed only to classes in the same package.
How many ways can an argument be passed to a subroutine?
An argument can be passed in two ways. They are Pass by Value and Passing by Reference.
Passing by value: This method copies the value of an argument into the formal parameter of the subroutine.
Passing by reference: In this method, a reference to an argument (not the value of the argument) is passed to the parameter.
What is the use of "bin" and "lib" in JDK?
"bin" contains all tools such as javac, appletviewer, awt tool, *etc.* whereas "lib" contains API and all packages.
What are different types of Transaction Isolation Levels?

The isolation level describes the degree to which the data being updated is visible to other transactions. This is important when two transactions are trying to read the same row of a table. Imagine two transactions: A and B. Here three types of inconsistencies can occur: * Dirty-read: A has changed a row, but has not committed the changes. B reads the uncommitted data but his view of the data may be wrong if A rolls back his changes and updates his own changes to the database.
*** Non-repeatable read: B performs a read, but A modifies or deletes that data later. If B reads the same row again, he will get different data.**
*** Phantoms: A does a query on a set of rows to perform an operation. B modifies the table such that a query of A would have given a different result. The table may be inconsistent.**

What do you mean by multiple inheritance in C++ ?
Multiple inheritance is a feature in C++ by which one class can be of different types. Say class teachingAssistant is inherited from two classes say teacher and Student.
What do you mean by virtual methods?

Virtual Methods are used to use the polymorphisms feature in C++. Say class A is inherited from class B. If we declare say function f() as virtual in class B and override the same function in class A then at runtime appropriate method of the class will be called depending upon the type of the object.

What do you mean by static methods?

By using the static method there is no need creating an object of that class to use that method. We can directly call that method on that class. For example, say class A has static function f(), then we can call f() function as A.f(). There is no need of creating an object of class A.

What are the advantages of OOPL?
Object oriented programming languages directly represent the real life objects. The features of OOPL as inhreitance, polymorphism, encapsulation makes it powerful.
Are there any predefined file filters for a JFileChooser?

Apart from the accept all filter, until Java 6, there were no predefined filters. Java 6 introduces the FileNameExtensionFilter, allowing you to define one or more types of files for the user to select.
FileFilter filter = new FileNameExtensionFilter(”JPEG file”, “jpg”, “jpeg”);
JFileChooser fileChooser = …;
fileChooser.addChoosableFileFilter(filter);

Why can’t my applet read or write to files?

Applets execute under the control of a web browser. Netscape and Internet Explorer impose a security restriction, that prohibits access to the local filesystem by applets. While this may cause frustration for developers, this is an important security feature for the end-user. Without it, applets would be free to modify the contents of a user’s hard-drive, or to read its contents and send this information back over a network.

Digitally signed applets can request permission to access the local file system, but the easiest way around the problem is to read and write to remote files located on a network drive. For example, in conjunction with a CGI script or servlet, you could send HTTP requests to store and retrieve data.

How can I call a Java method from Javascript?
Direct Web Remoting is an open source Java library which allows for easy integration of AJAX into your web site. It allows you to call Java methods directly from Javascript.
How do I find whether a parameter exists in the request object?
boolean hasFoo = !(request.getParameter(”foo”) == null || request.getParameter(”foo”).equals(”"));
or
boolean hasParameter = request.getParameterMap().contains(theParameter); //(which works in Servlet 2.3+)
What is the difference between RequestDispatcher and sendRedirect?
RequestDispatcher: server-side redirect with request and response objects. sendRedirect : Client-side redirect with new request and response objects.
What is the difference between directive include and jsp include?
<%@ include>: Used to include static resources during translation time. JSP include: Used to include dynamic content or static content during runtime.
What is the difference in using request.getRequestDispatcher() and context.getRequestDispatcher()?

request.getRequestDispatcher(path): In order to create it we need to give the relative path of the resource, context.getRequestDispatcher(path): In order to create it we need to give the absolute path of the resource.

What is the difference between ServletContext and PageContext?
ServletContext: Gives the information about the container. PageContext: Gives the information about the Request
How many types of JDBC Drivers are present and what are they?
There are 4 types of JDBC Drivers

JDBC-ODBC Bridge Driver Native API Partly Java Driver Network protocol Driver JDBC Net pure Java Driver

What is the query used to display all tables names in SQL Server (Query analyzer)?
select * from information_schema.tables
How do you call a stored procedure from JDBC?

The first step is to create a CallableStatement object. As with Statement an and PreparedStatement objects, this is done with an open onnection object. A CallableStatement object contains a call to a stored procedure.

CallableStatement cs = con.prepareCall(",call SHOW_SUPPLIERS-");
ResultSet rs = cs.executeQuery();
How can you use PreparedStatement?

This special type of statement is derived from class Statement.If you need a Statement object to execute many times, it will normally make sense to use a PreparedStatement object instead. The advantage to this is that in most cases, this SQL statement will be sent to the DBMS right away, where it will be compiled. As a result, the PreparedStatement object contains not just an SQL statement, but an SQL statement that has been precompiled. This means that when the PreparedStatement is executed, the DBMS can just run the PreparedStatement's SQL statement without having to compile it first.

PreparedStatement updateSales = con.prepareStatement("UPDATE COFFEES SET SALES = ? WHERE COF_NAME =
What are the different types of Statements?
Regular statement (use createStatement method), prepared statement (use prepareStatement method) and callable statement (use prepareCall)
How can you retrieve data from the ResultSet?

JDBC returns results in a ResultSet object, so we need to declare an instance of the class ResultSet to hold our results. The following code demonstrates declaring the ResultSet object rs.
ResultSet rs = stmt.executeQuery("SELECT COF_NAME, PRICE FROM COFFEES"); String s = rs.getString("COF_NAME");

The method getString is invoked on the ResultSet object rs, so getString() will retrieve (get) the value stored in the column COF_NAME in the current row of rs.
How can you create JDBC statements and what are they?

A Statement object is what sends your SQL statement to the DBMS. You simply create a Statement object and then execute it, supplying the appropriate execute method with the SQL statement you want to send. For a SELECT statement, the method to use is executeQuery. For statements that create or modify tables, the method to use is executeUpdate. It takes an instance of an active connection to create a Statement object. In the following example, we use our Connection object con to create the Statement object

Statement stmt = con.createStatement(); What will Class.forName do while loading drivers?
It is used to create an instance of a driver and register it with the DriverManager. When you have loaded a driver, it is available for making a connection with a DBMS.
How can you load the drivers?
Loading the driver or drivers you want to use is very simple and involves just one line of code. If, for example, you want to use the JDBC-ODBC Bridge driver, the following code will load it:
Class.forName(”sun.jdbc.odbc.JdbcOdbcDriver”);
Your driver documentation will give you the class name to use. For instance, if the class name is jdbc.DriverXYZ, you would load the driver with the following line of code:
Class.forName(”jdbc.DriverXYZ”);
What are the steps involved in establishing a JDBC connection?
This action involves two steps: loading the JDBC driver and making the connection.
What is a data source?

A DataSource class brings another level of abstraction than directly using a connection object. Data source can be referenced by JNDI. Data Source may point to RDBMS, file System , any DBMS *etc*.

What is 2 phase commit?
A 2-phase commit is an algorithm used to ensure the integrity of a committing transaction. In Phase 1, the transaction coordinator contacts potential participants in the transaction. The participants all agree to make the results of the transaction permanent but do not do so immediately. The participants log information to disk to ensure they can complete In phase 2 f
all the participants agree to commit, the coordinator logs that agreement

all the participants agree to commit, the coordinator logs that agreement and the outcome is decided. The recording of this agreement in the log ends in Phase 2, the coordinator informs each participant of the decision, and they permanently update their resources. What is Metadata and why should I use it?

Metadata ('data about data') is information about one of two things: Database information (java.sql.DatabaseMetaData), or Information about a specific ResultSet
(java.sql.ResultSetMetaData). Use DatabaseMetaData to find information about your database, such as its capabilities and structure. Use ResultSetMetaData to find information about the results of an SQL query, such as size and types of columns

What is a "dirty read"?

Quite often in database processing, we come across the situation wherein one transaction can change a value, and a second transaction can read this value before the original change has been committed or rolled back. This is known as a dirty read scenario because there is always the possibility that the first transaction may rollback the change, resulting in the second transaction having read an invalid value. While you can easily command a database to disallow dirty reads, this usually degrades the performance of your application due to the increased locking overhead. Disallowing dirty reads also leads to decreased system concurrency.

What is the advantage of using PreparedStatement?

If we are using PreparedStatement the execution time will be less. The PreparedStatement object contains not just an SQL statement, but the SQL statement that has been precompiled. This means that when the PreparedStatement is executed,the RDBMS can just run the PreparedStatement's Sql statement without having to compile it first.

When we will Denormalize data?

Data denormalization is reverse procedure, carried out purely for reasons of improving performance. It maybe efficient for a high-throughput system to replicate data for certain data.

What is cold backup, hot backup, warm backup recovery?

What is cold backup, hot backup, warm backup recovery?

Cold backup (All these files must be backed up at the same time, before the databaseis restarted). Hot backup (official name is ‘online backup’) is a backup taken of each tablespace while the database is running and is being accessed by the users.

Is the JDBC-ODBC Bridge multithreaded?
No. The JDBC-ODBC Bridge does not support concurrent access from different threads. The JDBC-ODBC Bridge uses synchronized methods to serialize all of the calls that it makes to ODBC. Multithreaded Java programs may use the Bridge, but they won’t get the advantages of multithreading.

What is an Applet? Should applets have constructors?

Applets are small programs transferred through Internet, automatically installed and run as part of web-browser. Applets implements functionality of a client. Applet is a dynamic and interactive program that runs inside a Web page displayed by a Java-capable browser. We don’t have the concept of Constructors in Applets. Applets can be invoked either through browser or through Appletviewer utility provided by JDK.

Can main method be declared final?
Yes, the main method can be declared final, in addition to being public static.
Can a public class MyClass be defined in a source file named YourClass.java?
No the source file name, if it contains a public class, must be the same as the public class name itself with a .java extension.
What is the default value of the local variables?

The local variables are not initialized to any default value, neither primitives nor object references. If you try to use these variables without initializing them explicitly, the java compiler will not compile the code. It will complain abt the local varaible not being initilized.

What are the different scopes for Java variables?

The scope of a Java variable is determined by the context in which the variable is declared. Thus a java variable can have one of the three scopes

at any given point in time. 1. Instance : - These are typical object level variables, they are initialized to default values at the time of creation of object, and remain accessible as long as the object accessible. 2. Local : - These are the variables that are defined within a method. They remain accessbile only during the course of method excecution. When the method finishes execution, these variables fall out of scope.
3. Static: - These are the class level variables. They are initialized when the class is loaded in JVM for the first time and remain there as long as the class remains loaded. They are not tied to any particular object instance.
What will be the initial value of an object reference which is defined as an instance variable?

The object references are all initialized to null in Java. However in order to do anything useful with these references, you must set them to a valid object, else you will get NullPointerExceptions everywhere you try to use such default initialized references.

What happens if you dont initialize an instance variable of any of the primitive types in Java?
Java by default initializes it to the default value for that primitive type. Thus an int will be initialized to 0, a boolean will be initialized to false.
Are main, next delete, exit keywords in Java?
No, they are not keywords in Java.

delete is not a keyword in Java. Java does not make use of explicit destructors the way C++ does.
To exit a program explicitly you use exit method in System object.

Is String a primitive data type in Java?
No String is not a primitive data type in Java, even though it is one of the most extensively used object. Strings in Java are instances of String class defined in java.lang package.
Can a .java file contain more than one java classes?
Yes, a .java file contain more than one java classes, provided at the most one of them is a public class.
Is Empty.java file a valid source file?
Yes, an empty .java file is a perfectly valid source file.
What does it mean that a method or field is “static”?

Static variables and methods are instantiated only once per class. In other

Static variables and methods are instantiated only once per class. In other words they are class variables, not instance variables. If you change the value of a static variable in a particular object, the value of that variable changes for all instances of that class. Static methods can be referenced with the name of the class rather than the name of a particular object of the class (though that works too). That’s how library methods like System.out.println() work out is a static field in the java.lang.System class.

What modifiers are allowed for methods in an Interface?
Only public and abstract modifiers are allowed for methods in interfaces.
What is Externalizable?

Externalizable is an Interface that extends Serializable Interface. And sends data into Streams in Compressed Format. It has two methods, writeExternal(ObjectOuput out) and readExternal(ObjectInput in)

What method must be implemented by all threads?
All tasks must implement the run() method, whether they are a subclass of Thread or implement the Runnable interface.
What are synchronized methods and synchronized statements?

Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method’s object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

Can an unreachable object become reachable again?

An unreachable object may become reachable again. This can happen when the object’s finalize() method is invoked and the object performs an operation which causes it to become accessible to reachable objects.

How do I serialize an object to a file?

The class whose instances are to be serialized should implement an interface Serializable. Then you pass the instance to the ObjectOutputStream which is connected to a fileoutputstream. This will save the object to a file.
Which methods of Serializable interface should I implement?

The serializable interface is an empty interface, it does not contain any methods. So we do not implement any methods.
How can I customize the seralization process? *i.e.* how can one have a control over the serialization process?

Yes it is possible to have control over serialization process. The class should implement Externalizable interface. This interface contains two methods namely readExternal and writeExternal. You should implement these methods and write the logic for customizing the serialization process.

What is the common usage of serialization?
Whenever an object is to be sent over the network, objects need to be serialized. Moreover if the state of an object is to be saved, objects need to be serilazed.
When you serialize an object, what happens to the object references included in the object?

The serialization mechanism generates an object graph for serialization. Thus it determines whether the included object references are serializable or not. This is a recursive process. Thus when an object is serialized, all the included objects are also serialized alongwith the original obect.

What one should take care of while serializing the object?
One should make sure that all the included objects are also serializable. If any of the objects is not serializable then it throws a NotSerializableException.
What happens to the static fields of a class during serialization?

There are three exceptions in which serialization doesnot necessarily read and write to the stream. These are
1. Serialization ignores static fields, because they are not part of ay particular state state. 2. Base class fields are only hendled if the base class itself is serializable.
3. Transient fields.
Does importing a package imports the subpackages as well?

No you will have to import the subpackages explicitly.
For eg: Importing com.MyTest.* will import classes in the package MyTest only. It will not import any class in any of it's subpackage.

What is the sequence for calling the methods by AWT for applets?
When an applet begins, the AWT calls the following methods, in this sequence: init()
start()
paint()
When an applet is terminated, the following sequence of method calls takes place
stop()
destroy()
What are the Applet’s Life Cycle methods? Explain them?
Following are methods in the life cycle of an Applet:

init() method - called when an applet is first loaded. This method is called only once in the entire cycle of an applet. This method usually intialize the variables to be used in the applet

start() method - called each time an applet is started

paint() method - called when the applet is minimized or refreshed. This method is used for drawing different strings, figures, and images on the applet
window

stop() method - called when the browser moves off the applet’s page
destroy() method - called when the browser is finished with the applet
What are some alternatives to inheritance?

Delegation is an alternative to inheritance. Delegation means that you include an instance of another class as an instance variable, and forward messages to the instance. It is often safer than inheritance because it forces you to think about each message you forward, because the instance is of a known class, rather than a new class, and because it doesn’t force you to accept all the methods of the super class: you can provide only the methods that really make sense. On the other hand, it makes you write more code, and it is harder to re-use (because it is not a subclass).

If I write return at the end of the try block, will the finally block still execute?
Yes even if you write return as the last statement in the try block and no exception occurs, the finally block will execute. The finally block will

execute and then the control return.
Is it necessary that each try block must be followed by a catch block?

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

What are the different ways to handle exceptions?

There are two ways to handle exceptions,
1. By wrapping the desired code in a try block followed by a catch block to catch the exceptions. and
2. List the desired exceptions in the throws clause of the method and let the caller of the method hadle those exceptions.

How does an exception permeate through the code?

An unhandled exception moves up the method stack in search of a matching When an exception is thrown from a code which is wrapped in a try block followed by one or more catch blocks, a search is made for matching catch block. If a matching type is found then that block will be invoked. If a matching type is not found then the exception moves up the method stack and reaches the caller method. Same procedure is repeated if the caller method is included in a try catch block. This process continues until a catch block handling the appropriate type of exception is found. If it does not find such a block then finally the program terminates.
What are runtime exceptions?

Runtime exceptions are those exceptions that are thrown at runtime because of either wrong input data or because of wrong business logic *etc.* These are not checked by the compiler at compile time.

What are checked exceptions?
Checked exception are those which the Java compiler forces you to catch. *e.g.* IOException are checked Exceptions.
Does Java provide any construct to find out the size of an object?
No there is not sizeof operator in Java. So there is not direct way to determine the size of an object directly in Java.
What is Externalizable interface?

Externalizable is an interface which contains two methods readExternal and writeExternal. These methods give you a control over the serialization mechanism. Thus if your class implements this interface, you can customize the serialization process by implementing these methods.

What type of parameter passing does Java support?
In Java the arguments are always passed by value .
Can a top level class be private or protected?

No, a top level class can not be private or protected. It can have either “public” or no modifier. If it does not have a modifier it is supposed to have a default access.If a top level class is declared as private the compiler will complain that the “modifier private is not allowed here”. This means that a top level class can not be private. Same is the case with protected.

What is the difference between declaring a variable and defining a variable?

In declaration we just mention the type of the variable and it’s name. We do not initialize it. But defining means declaration + initialization.
e.g String s; is just a declaration while String s = new String (”abcd”); Or String s = “abcd”; are both definitions.
What are different types of inner classes?

Nested toplevel classes, Member classes, Local classes, Anonymous classes

Nested toplevel classes-If you declare a class within a class and specify the static modifier, the compiler treats the class just like any other toplevel class.
Any class outside the declaring class accesses the nested class with the declaring class name acting similarly to a package. eg, outer.inner. Toplevel inner classes implicitly have access only to static variables.There can also be inner interfaces. All of these are of the nested toplevel variety.

Member classes - Member inner classes are just like other member methods and member variables and access to the member class is restricted, just like methods and variables. This means a public member class acts similarly to a nested toplevel class. The primary difference between member classes and nested toplevel classes is that member classes have access to the specific

instance of the enclosing class.

Local classes - Local classes are like local variables, specific to a block of code. Their visibility is only within the block of their declaration. In order for the class to be useful beyond the declaration block, it would need to implement a
more publicly available interface.Because local classes are not members, the modifiers public, protected, private, and static are not usable.

Anonymous classes - Anonymous inner classes extend local inner classes one level further. As anonymous classes have no name, you cannot provide a constructor.
What is Overriding?

When a class defines a method using the same name, return type, and arguments as a method in its superclass, the method in the class overrides the method in the superclass. When the method is invoked for an object of the class, it is the new definition of the method that is called, and not the method definition from superclass. Methods may be overridden to be more public, not more private.

What are Checked and UnChecked Exception?

A checked exception is some subclass of Exception (or Exception itself), excluding class RuntimeException and its subclasses.
Making an exception checked forces client programmers to deal with the possibility that the exception will be thrown. eg, IOException thrown by java.io.FileInputStream's read() method· Unchecked exceptions are RuntimeException and any of its subclasses. Class Error and its subclasses also are unchecked. With an unchecked exception, however, the compiler doesn't force client programmers either to catch the
exception or declare it in a throws clause. In fact, client programmers may not even know that the exception could be thrown. eg, StringIndexOutOfBoundsException thrown by String's charAt() method· Checked exceptions must be caught at compile time. Runtime exceptions do not need to be. Errors often cannot be.

How can one prove that the array is not null but empty using one line of code?
Print args.length. It will print 0. That means it is empty. But if it would

have been null then it would have thrown a NullPointerException on attempting to print args.length.
What is the first argument of the String array in main method?
The String array is empty. It does not have any element. This is unlike C/C++ where the first element by default is the program name.
What is an Iterator?

Some of the collection classes provide traversal of their contents via a java.util.Iterator interface. This interface allows you to walk through a collection of objects, operating on each object in turn. Remember when using Iterators that they contain a snapshot of the collection at the time the Iterator was obtained; generally it is not advisable to modify the collection itself while traversing an Iterator.

Difference between Vector and ArrayList?
Vector is synchronized whereas arraylist is not.
Difference between HashMap and HashTable?

The HashMap class is roughly equivalent to Hashtable, except that it is unsynchronized and permits nulls. (HashMap allows null values as key and value whereas Hashtable doesnt allow). HashMap does not guarantee that the order of the map will remain constant over time. HashMap is unsynchronized and Hashtable is synchronized.

What is HashMap and Map?
Map is Interface and Hashmap is class that implements that.
What are pass by reference and passby value?
Pass By Reference means the passing the address itself rather than passing the value. Passby Value means passing a copy of the value to be passed.
Describe synchronization in respect to multithreading.

With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchonization, it is possible for one thread to modify a shared variable while another thread is in the process of using or updating same shared variable. This usually leads to significant errors.

What is the purpose of garbage collection in Java, and when is it used?

The purpose of garbage collection is to identify and discard objects that are

no longer needed by a program so that their resources can be reclaimed and reused. A Java object is subject to garbage collection when it becomes unreachable to the program in which it is used.

What is the difference between a constructor and a method?
A constructor is a member function of a class that is used to create objects of that class. It has the same name as the class itself, has no return type, and is invoked using the new operator. A method is an ordinary member function of a class. It has its own name, a return type (which may be void), and is invoked using the dot operator.

What is final?

A final class can't be extended ie., final class may not be subclassed. A final method can't be overridden when its class is inherited. You can't change value of a final variable (is a constant).

What is static in java?

Static means one per class, not one for each object no matter how many instance of a class might exist. This means that you can use them without creating an instance of a class.Static methods are implicitly final, because overriding is done based on the type of the object, and static methods are attached to a class, not an object. A static method in a superclass can be shadowed by another static method in a subclass, as long as the original method was not declared final. However, you can't override a static method with a nonstatic method. In other words, you can't change a static method into an instance method in a subclass.

What if the main method is declared as private?
The program compiles properly but at runtime it will give "Main method not public." message.
What is the difference between HttpServlet and GenericServlet?

A GenericServlet has a service() method aimed to handle requests. HttpServlet extends GenericServlet and adds support for doGet(), doPost(), doHead() methods (HTTP 1.0) plus doPut(), doOptions(), doDelete(), doTrace() methods (HTTP 1.1).
Both these classes are abstract

What is an initialization of servlet?

What is preinitialization of a servlet?

A container doesnot initialize the servlets ass soon as it starts up, it initializes a servlet when it receives a request for that servlet first time. This is called lazy loading. The servlet specification defines the element, which can be specified in the deployment descriptor to make the servlet container load and initialize the servlet as soon as it starts up. The process of loading a servlet before any request comes in is called preloading or preinitializing a servlet.

What is the difference between ServletContext and ServletConfig?

ServletContext: Defines a set of methods that a servlet uses to communicate with its servlet container, for example, to get the MIME type of a file, dispatch requests, or write to a log file.The ServletContext object is contained within the ServletConfig object, which the Web server provides the servlet when the servlet is initialized

ServletConfig: The object created after a servlet is instantiated and its default constructor is read. It is created to pass initialization information to the servlet.
Explain ServletContext.

ServletContext interface is a window for a servlet to view it’s environment. A servlet can use this interface to get information such as initialization parameters for the web applicationor servlet container’s version. Every web application has one and only one ServletContext and is accessible to all active resource of that application.

What is the difference between the getRequestDispatcher(String path) method of javax.servlet.ServletRequest interface and javax.servlet.ServletContext interface?

The getRequestDispatcher(String path) method of javax.servlet.ServletRequest interface accepts parameter the path to the resource to be included or forwarded to, which can be relative to the request of the calling servlet. If the path begins with a “/” it is interpreted as relative to the current context root.
The getRequestDispatcher(String path) method of javax.servlet.ServletContext interface cannot accepts relative paths. All
[illegible]

path must sart with a "/" and are interpreted as relative to curent context root.

Explain the life cycle methods of a Servlet.

The javax.servlet.Servlet interface defines the three methods known as lifecycle method. public void init(ServletConfig config) throws ServletException
public void service(ServletRequest req, ServletResponse res) throws ServletException, IOException
public void destroy()
First the servlet is constructed, then initialized wih the init() method.
Any request from client are handled initially by the service() method before delegating to the doXxx() methods in the case of HttpServlet.

The servlet is removed from service, destroyed with the destroy() methid, then garbaged collected and finalized.
What is an Iterator interface?
The Iterator interface is used to step through the elements of a Collection .
Describe the Garbage Collection process in Java ?

The JVM spec mandates automatic garbage collection outside of the programmers control. The System.gc() or Runtime.gc() is merely a suggestion to the JVM to run the GC process but is NOT guaranteed.

How many static init can you have ?

As many as you want, but the static initializers and class variable initializers are executed in textual order and may not refer to class variables declared in the class whose declarations appear textually after the use, even though these class variables are in scope.

What is constructor chaining and how is it achieved in Java ?

A child object constructor always first needs to construct its parent (which in turn calls its parent constructor.). In Java it is done via an implicit call to the no-args constructor as the first statement.
What methods can be overridden in Java?

In C++ terminology, all public methods in Java are virtual. Therefore, all Java methods can be overwritten in subclasses except those that are

Java methods can be overwritten in subclasses except those that are declared final, static, and private.
Describe java's security model.

Java's security model is one of the most interesting and unique aspects of the language. For the most part it's broken into two pieces: the user adjustable security manager that checks various API operations like file access, and the byte code verifier that asserts the validity of compiled byte code. public abstract class SecurityManager java.lang.SecurityManager is an abstract class which different applications subclass to implement a particular security policy. It allows an application to determine whether or not a particular operation will generate a security exception.

What is daemon thread and which method is used to create the daemon thread?

Daemon thread is a low priority thread which runs intermittently in the back ground doing the garbage collection operation for the java runtime system. setDaemon method is used to create a daemon thread.

What are Transient and Volatile Modifiers?

Transient: The transient modifier applies to variables only and it is not stored as part of its object's Persistent state. Transient variables are not serialized.
Volatile: Volatile modifier applies to variables only and it tells the compiler that the variable modified by volatile can be changed unexpectedly by other parts of the program.

What is preemptive and Non-preemptive Time Scheduling?

Preemptive: Running tasks are given small portions of time to execute by using time-slicing. Non-Preemptive: One task doesn't give another task a chance to run until its finished or has normally yielded its time.

What is runnable?
Its an Interface through which Java implements Threads.The class can extend from any class but if it implements Runnable,Threads can be used in that particular application. What is the difference between set and list?
Set stores elements in an unordered way but does not contain duplicate elements, whereas list stores elements in an ordered way but may contain

duplicate elements.
What is a layout manager?
A layout manager is an object that is used to organize components in a container.
What is the difference between a public and a non-public class?
A public class may be accessed outside of its package. A non-public class may not be accessed outside of its package.
What is the difference between instanceof and isInstance?

instanceof is used to check to see if an object can be cast into a specified type without throwing a cast classexception. isInstance()
Determines if the specified Object is assignment-compatible with the object represented by this Class. This method is the dynamic equivalent of the Java language instanceof operator. The method returns true if the specified Object argument is non-null and can be cast to the reference type represented by this Class object without raising a ClassCastException. It returns false otherwise.

What does the “abstract” keyword mean in front of a method? A class?

Abstract keyword declares either a method or a class. If a method has a abstract keyword in front of it,it is called abstract method.Abstract method hs no body.It has only arguments and return type.Abstract methods act as placeholder methods that are implemented in the subclasses.
Abstract classes can’t be instantiated.If a class is declared as abstract,no objects of that class can be created.If a class contains any abstract method it must be declared as abstract

What is JDBC? Describe the steps needed to execute a SQL query using JDBC.

The JDBC is a pure Java API used to execute SQL statements. It provides a set of classes and interfaces that can be used by developers to write database applications. The steps needed to execute a SQL query using JDBC:
1. Open a connection to the database.
2. Execute a SQL statement.
3. Process th results.
4. Close the connection to the database.
What is RMI?

RMI stands for Remote Method Invocation. Traditional approaches to executing code on other machines across a network have been confusing as well as tedious and error-prone to implement. The nicest way to think about this problem is that some object happens to live on another machine, and that you can send a message to the remote object and get a result as if the object lived on your local machine. This simplification is exactly what Java Remote Method Invocation (RMI) allows you to do.

What are native methods? How do you use them?
Native methods are methods that are defined as public static methods within a java class, but whose implementation is provided in another programming language such as C.
What does the keyword "synchronize" mean in java. When do you use it? What are the

Synchronize is used when u want to make ur methods thread safe. The disadvantage of synchronise is it will end up in slowing down the program. Also if not handled properly it will end up in dead lock.

1. Only use (and minimize it's use)synchronization when writing multithreaded code as there is a speed (up to five to six time slower, depending on the execution time of the synchronized/non-synchronized method) cost associated with its use.

2. In case of syncronized method modifier, the byte code generated is the exact same as nonsyncronized method. The only difference is that a flag called ACC_SYNCRONIZED property flag in method's method_info structure is set if the syncronized method modifier is present.

3. Also, syncronized keyword can make the code larger in size if used in the body of the method as bytecode for monitorenter/monitorexit is generated in addition to any exception handling.

What is the difference between a Vector and an Array. Discuss the advantages and disadvantages

The vector container class generalizes the concept of an ordinary C array. Like an array, a vector is an indexed data structure, with index values that range from 0 to one less than the number of elements contained in the structure. Also like an array, values are most commonly assigned to and

extracted from the vector using the subscript operator. However, the vector differs from an array in the following important
The size of the vector can change dynamically. New elements can be inserted on to the end of a vector, or into the middle. It is important to note, however, that while these abilities are provided, insertion into the middle of a vector is not as efficient as insertion into the middle of a list. A vector has more “self-knowledge” than an ordinary array. In particular, a vector can be queried about its size, about the number of elements it can potentially hold (which may be different from its current size), and so on.
A vector can only hold references to objects and not primitive types. Vector Implementaions are usually slower then array because of all the functionality that comes with them. As implemented in Java, vector is a thread-safe class and hence all methods are synchronous methods, which makes them considerably slow.

Java says “write once, run anywhere”. What are some ways this isn’t quite true?

Any time you use system calls specific to one operating system and do not create alternative calls for another operating system, your program will not function correctly. Solaris systems and Intel systems order the bits of an integer differently. (You may have heard of little endian vs. big endian)
If your code uses bit shifting, or other binary operators, they will not work on systems that have opposide endianism.

What is the difference between an Applet and an Application?

1. Applets can be embedded in HTML pages and downloaded over the Internet whereas Applications have no special support in HTML for embedding or downloading.
2. Applets can only be executed inside a java compatible container, such as a browser or appletviewer
whereas Applications are executed at command line by java.exe or jview.exe.
3. Applets execute under strict security limitations that disallow certain operations(sandbox model security) whereas Applications have no inherent security restrictions.
4. Applets don’t have the main() method as in applications. Instead they operate on an entirely different mechanism where they are initialized by init(),started by start(),stopped by stop() or destroyed by destroy().

init(),started by start(),stopped by stop() or destroyed by destroy().

How can you force all derived classes to implement a method present in the base class?

Creating and implementing an interface would be the best way for this situation. Just create an interface with empty methods which forces a programmer to implement all the methods present under it.
Another way of achieving this task is to declare a class as abstract with all its methods abstract.
What are abstract classes, abstract methods?

Simply speaking a class or a method qualified with "abstract" keyword is an abstract class or abstract method.
You create an abstract class when you want to manipulate a set of classes through a common interface. All derived-class methods that match the signature of the base-class declaration will be called using the dynamic binding mechanism.
An abstract method is an incomplete method. It has only a declaration and no method body. Here is the syntax for an abstract method declaration:
abstract void f();

What's the difference between == and equals method?

The equals method can be considered to perform a deep comparison of the value of an object, whereas the == operator performs a shallow comparison. The equals() method compares the characters inside a string object. == operator compares two object references to check whether they refer to the same instances or not.

Describe, in general, how java's garbage collector works?

The Java runtime environment deletes objects when it determines that they are no longer being used. This process is known as garbage collection. The Java runtime environment supports a garbage collector that periodically frees the memory used by objects that are no longer needed. The Java garbage collector is a mark-sweep garbage collector that scans Java's dynamic memory areas for objects, marking those that are referenced. After all possible paths to objects are investigated, those objects that are not marked (i.e. are not referenced) are known to be garbage and are collected.

What is the difference between StringBuffer and String class?

A string buffer implements a mutable sequence of characters. A string buffer is like a String, but can be modified. At any point in time it contains some particular sequence of characters, but the length and content of the sequence can be changed through certain method calls. The String class represents character strings. All string literals in Java programs, such as “abc” are constant and implemented as instances of this class; their values cannot be changed after they are created.

How can you achieve Multiple Inheritance in Java?

Java’s interface mechanism can be used to implement multiple inheritance, with one important difference from c++ way of doing MI: the inherited interfaces must be abstract. This obviates the need to choose between different implementations, as with interfaces there are no implementations.
What are interfaces?

Interfaces provide more sophisticated ways to organize and control the objects in your system. The interface keyword takes the abstract concept one step further. You could think of it as a “pure” abstract class. It allows the creator to establish the form for a class: method names, argument lists, and return types, but no method bodies. An interface can also contain fields, but The interface keyword takes the abstract concept one step further. You could think of it as a “pure” abstract class. It allows the creator to establish the form for a class: method names, argument lists, and return types, but no method bodies. An interface can also contain fields, but An interface says: “This is what all classes that implement this particular interface will look like.” Thus, any code that uses a particular interface knows what methods might be called for that interface, and that’s all. So the interface is used to establish a “protocol” between classes.

How to make application thread-safe ?
You should use the word synchronized to mark the critical section of code. You may also use other methods of thread synchronization (see wait(), notify(), notifyAll() ***etc.***
What do you know about networking support in Java ?

Java supports “low-level” and “high-level” classes. “Low-level” classes provide support for socket programming: Socket, DatagramSocket, and

ServerSocket classes. “High-level” classes provide “Web programming”: URL, URLEncoder, and URLConnection classes. Networking programming classes ease the programming of network applications, but do not substitute your knowledge of networking. Java networking like anything else in Java is platformindependent.

What are the problems faced by Java programmers who don’t use layout managers?

Without layout managers, Java programmers are faced with determining how their GUI will be displayed across multiple windowing systems and finding a common sizing and positioning that will work within the constraints imposed by each windowing system.

What are the two basic ways in which classes that can be run as threads may be defined?
A thread class may be declared as a subclass of Thread, or it may implement the Runnable interface.
What is the purpose of a statement block?
A statement block is used to organize a sequence of statements as a single statement group. What is the purpose of garbage collection?
The purpose of garbage collection is to identify and discard objects that are no longer needed by a program so that their resources may be reclaimed and reused.
How are this and super used?
This is used to refer to the current object instance. super is used to refer to the variables and methods of the superclass of the current object instance.
What an I/O filter?
An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.
How are the elements of a GridLayout organized?
The elements of a GridBad layout are of equal size and are laid out using the squares of a grid.
How are this() and super() used with constructors?
this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.
How can the Checkbox class be used to create a radio button? By associating Checkbox objects with a CheckboxGroup
If a method is declared as protected, where may the method be accessed?

If a method is declared as protected, where may the method be accessed?
A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.
When does the compiler supply a default constructor for a class?
The compiler supplies a default constructor for a class if no other constructors are provided.
What is the highest-level event class of the event-delegation model?
The java.util.EventObject class is the highest-level class in the event-delegation class hierarchy.
What restrictions are placed on the values of each case of a switch statement?
During compilation, the values of each case of a switch statement must evaluate to a value that can be promoted to an int value.
How are the elements of a CardLayout organized?
The elements of a CardLayout are stacked, one on top of the other, like a deck of cards.
For which statements does it make sense to use a label?
The only statements for which it makes sense to use a label are those statements that can enclose a break or continue statement.
How is rounding performed under integer division?
The fractional part of the result is truncated. This is known as rounding toward zero.
Can an object be garbage collected while it is still reachable?
A reachable object cannot be garbage collected. Only unreachable objects may be garbage collected.
What is the difference between a Window and a Frame?
The Frame class extends Window to define a main application window that can have a menu bar.
When can an object reference be cast to an interface reference?
An object reference be cast to an interface reference when the object implements the referenced interface.
What is the Dictionary class?
The Dictionary class provides the capability to store key-value pairs.
What are the high-level thread states?
The high-level thread states are ready, running, waiting, and dead. What is the argument type of a program’s main() method?
A program’s main() method takes an argument of the String*+ type.
What invokes a thread’s run() method?
After a thread is started, via its start() method or that of the Thread class, the JVM invokes the thread’s run() method when the thread is initially

the JVM invokes the thread's run() method when the thread is initially executed.
What are order of precedence and associativity, and how are they used?
Order of precedence determines the order in which operators are evaluated in expressions. Associatity determines whether an expression is evaluated left-to-right or right-to-left
What is clipping?
Clipping is the process of confining paint operations to a limited area or shape.
Which characters may be used as the second character of an identifier, but not as the first
The digits 0 through 9 may not be used as the first character of an identifier but they may be used after the first character of an identifier
What is CTS (Common Type System)?
It defines about how Objects should be declard, defined and used within .NET. CLS is the subset of CTS.
What is a policy?.

It's an abstract class for representing the system security policy for a Java application environment(specifying which permissions are available for code from various sources). Java security properties file resides in *lib*security/java.security directory. Value of "policy.provider" should be changed.
What are the restrictions imposed by a Security Manager on Applets?.

i) Cannot read or write files on the host that's executing it.
ii) Cannot load libraries or define native methods.
iii) Cannot make network connections except to the host that it came from
iv) Cannot start any program on the host that's executing it.
v) Cannot read certain system properties.
vi) Windows that an applet brings up look different than windows that an application

brings up.
Scope and lifetime of variables ?
Scope of variables is only to that particular blocklifetime will be till the block ends.variables declared above the block within the class are valid to that inner block also.
What is Object class and java.lang ?

Object class is the superclass of all the classes and means that reference variable of type object can refer to an object of any other class. and also defines methods like finalise,wait. java.lang contains all the basic language functions and is imported in all the programs implicitly.

What r packages and why ? how to execute a program in a package ?
Package is a set of classes, which can be accessed by themselves and cannot be accessed outside the package. and can be defined as package

.Package name and the directory name must be the same.And the execution of programs in package is done by : java mypack.account where mypack is directory name and account is program name.

View the Answer
What is a StringTokenizer ?

String Tokenizer provide parsing process in which it identifies the delimiters provided by the user , by default delimiters are spaces, tab, newline *etc.* and separates them from the tokens. Tokens are those which are separated by delimiters.

What are nested classes ?

There are two types : static and nonstatic.
static class means the members in its enclosing class (class within class) can be accessed by creating an object and cannot be accessed directly without creating the object. nonstatic class means inner class and can be accessed directly with the object created for the outer class no need to create again an object like static class.

What are methods and how are they defined?
Methods are functions that operate on instances of classes in which they are defined. Objects can communicate with each other using methods and can call methods in other classes.Method definition has four parts. They are name of the method, type of object or primitive type the method returns, a list of parameters and the body of the method. A method’s signature is a combination of the first three parts mentioned above.

What is Garbage Collection and how to call it explicitly?
When an object is no longer referred to by any variable, java automatically reclaims memory used by that object. This is known as garbage

reclaims memory used by that object. This is known as garbage collection.System.gc() method may be used to call it explicitly.

What is interface and its use?

Interface is similar to a class which may contain method's signature only but not bodies and it is a formal set of method and constant declarations that must be defined by the class that implements it. Interfaces are useful for:

a) Declaring methods that one or more classes are expected to implement b) Capturing similarities between unrelated classes without forcing a class relationship. c) Determining an object's programming interface without revealing the actual body of the class.

What is the difference between abstract class and interface?
a) All the methods declared inside an interface are abstract whereas abstract class must have at least one abstract method and others may be concrete or abstract.

b) In abstract class, key word abstract must be used for the methodsWhereas interface we need not use that keyword for the methods.
c)Abstract class must have subclasses whereas interface can't have subclasses.

What is the difference between exception and error?

The exception class defines mild error conditions that your program encounters. Ex: Arithmetic Exception, FilenotFound exception Exceptions can occur when try to open the file, which does not exist,the network connection is disrup,operands being manipulated are out of prescribed range,the class file you are interested in loading is missingThe error class defines serious error conditions that you should not attempt to recover from. In most cases it is advisable to let the program terminate when such an error is encountered.Ex: Running out of memory error, Stack overflow error.

What is the difference between applications and applets?

a) Application must be run on local machine whereas applet needs no explicit installation on local machine.

explicit installation on local machine.
b) Application must be run explicitly within a java-compatible virtual machine whereas applet loads and runs itself automatically in a java-enabled browser.
c) Application starts execution with its main method whereas applet starts execution with its init method.
d) Application can run with or without graphical user interface whereas applet must run within a graphical user interface.

What is an event and what are the models available for event handling?

An event is an event object that describes a state of change in a source. In other words, event occurs when an action is generated, like pressing button, clicking mouse, selecting a list, *etc.* There are two types of models for handling events and they are: a) event-inheritance model and b) event-delegation model

What is adapter class?
An adapter class provides an empty implementation of all methods in an event listener interface. Adapter classes are useful when you want to receive and process only some of the events that are handled by a particular event listener interface. You can define a new class to act listener by extending one of the adapter classes and implementing only those events in which you are interested.

What is meant by controls and what are different types of controls in AWT?

Controls are components that allow a user to interact with your application and the AWT supports the following types of controls:Labels, Push Buttons, Check Boxes, Choice Lists, Lists, Scrollbars, Text Components.

These controls are subclasses of Component.
What is the difference between choice and list?

A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices and only one item may be selected from a choice.A List may be displayed in such a way that several list items are visible and it supports the selection of one or more list items.

What is a Java package and how is it used?

A Java package is a naming context for classes and interfaces. A package is used to create a separate name space for groups of classes and interfaces. Packages are also used to organize related classes and interfaces into a single API unit and to control accessibility to these classes and interfaces.

What is the difference between the prefix and postfix forms of the ++ operator? The prefix form performs the increment operation and returns the value of the increment operation. The postfix form returns the current value all of the expression and then performs the increment operation on that value.

What is numeric promotion?
Numeric promotion is the conversion of a smaller numeric type to a larger numeric type, so that integer and floating-point operations may take place. In numerical promotion, byte, char, and short values are converted to int values. The int values are also converted to long values, if necessary. The long and float values are converted to double values, as required.

What are three ways in which a thread can enter the waiting state?

A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully attempting to acquire an object's lock, or by invoking an object's wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.

What happens if an exception is not caught?
An uncaught exception results in the uncaughtException() method of the thread's ThreadGroup being invoked, which eventually results in the termination of the program in which it is thrown.

What restrictions are placed on method overriding?

Overridden methods must have the same name, argument list, and return type. The overriding method may not limit the access of the method it overrides. The overriding method may not throw any exceptions that may not be thrown by the overridden method.

What is your platform's default character encoding?
If you are running Java on English Windows platforms, it is probably Cp1252. If you are running Java on English Solaris platforms, it is most

likely 8859_1..
What is the difference between the File and RandomAccessFile classes?

The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.

What is the purpose of the enableEvents() method?

The enableEvents() method is used to enable an event for a particular object. Normally, an event is enabled when a listener is added to an object for a particular event. The enableEvents() method is used by objects that handle events by overriding their eventdispatch methods.

What is the difference between the JDK 1.02 event model and the event-delegation

The JDK 1.02 event model uses an event inheritance or bubbling approach. In this model, components are required to handle their own events. If they do not handle a particular event, the event is inherited by (or bubbled up to) the component's container. The container then either handles the event or it is bubbled up to its container and so on, until the highest-level container has been tried.

In the event-delegation model, specific objects are designated as event handlers for GUI components. These objects implement event-listener interfaces. The event-delegation model is more efficient than the event-inheritance model because it eliminates the processing required to support the bubbling of unhandled events.

What is the difference between a Choice and a List?

A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices. Only one item may be selected from a Choice. A List may be displayed in such a way that several List items are visible. A List supports the selection of one or more List items.
What is casting?

There are two types of casting, casting between primitive numeric types and casting between object references. Casting between numeric types is used to convert larger values, such as double values, to smaller values, such as byte values. Casting between object references is used to refer to an object by a compatible class, interface, or array type reference.

When is the finally clause of a try-catch-finally statement executed?
The finally clause of the try-catch-finally statement is always executed unless the thread of execution terminates or an exception occurs within the execution of the finally clause.
How does multithreading take place on a computer with a single CPU?
The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.
What is the difference between static and nonstatic variables?
A static variable is associated with the class as a whole rather than with specific instances of a class. Nonstatic variables take on unique values with each object instance.
What advantage do Java's layout managers provide over traditional windowing systems?

Java uses layout managers to lay out components in a consistent manner across all windowing platforms. Since Java's layout managers aren't tied to absolute sizing and positioning, they are able to accomodate platform-specific differences among windowing systems.

How are the elements of a GridBagLayout organized?
The elements of a GridBagLayout are organized according to a grid. However, the elements are of different sizes and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.

What is the difference between a while statement and a do statement?

A while statement checks at the beginning of a loop to see whether the next loop iteration should occur. A do statement checks at the end of a loop to see whether the next iteration of a loop should occur. The do statement will always execute the body of a loop at least once. What is the relationship between an event-listener interface and an
An event-listener interface defines the methods that must be implemented by an event handler for a particular kind of event. An event adapter provides a default implementation of an event-listener interface.

If a class is declared without any access modifiers, where may the class be accessed?

A class that is declared without any access modifiers is said to have package access. This means that the class can only be accessed by other classes and interfaces that are defined within the same package.

What classes of exceptions may be caught by a catch clause?
A catch clause can catch any exception that may be assigned to the Throwable type. This includes the Error and Exception types.
What happens when a thread cannot acquire a lock on an object?
If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object's lock, it enters the waiting state until the lock becomes available
What is the difference between the Font and FontMetrics classes?
The FontMetrics class is used to define implementation-specific properties, such as ascent and descent, of a Font object.
What is the difference between a Window and a Frame?
The Frame class extends Window to define a main application window that can have a menu bar.
What is the % operator?
It is referred to as the modulo or remainder operator. It returns the remainder of dividing the first operand by the second operand.

What is an object's lock and which object's have locks?
An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the object. A thread may execute a synchronized method of an object only after it has acquired the object's lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.

What is the difference between a static and a nonstatic inner class?
A nonstatic inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.
How are Java source code files named?

A Java source code file takes the name of a public class or interface that is defined within the file. A source code file may contain at most one public class or interface. If a public class or interface is defined within a source code file, then the source code file must take the name of the public class or

interface. If no public class or interface is defined within a source code file, then the file must take on a name that is different than its classes and interfaces. Source code files use the .java extension.

What is the purpose of the wait(), notify(), and notifyAll() methods?

The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to wait for a shared resource. When a thread executes an object's wait() method, it enters the waiting state. It only enters the ready state after another thread invokes the object's notify() or notifyAll() methods..

What is the advantage of the event-delegation model over the earlier event-inheritance model?
The event-delegation model has two advantages over the event-inheritance model. First, it enables event handling to be handled by objects other than the ones that generate the events (or their containers). This allows a clean separation between a component's design and its use. The other advantage of the event-delegation model is that it performs much better in applications where many events are generated. This performance improvement is due to the fact that the event-delegation model does not have to repeatedly process unhandled events, as is the case of the event-inheritance model.

What must a class do to implement an interface?
It must provide all of the methods in the interface and identify the interface in its implements clause.
What is the difference between a break statement and a continue statement?

A break statement results in the termination of the statement to which it applies (switch, for, do, or while). A continue statement is used to end the current loop iteration and return control to the loop statement.

What is the Locale class?
The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region.
What is the purpose of the finally clause of a try-catch-finally statement?
The finally clause is used to provide the capability to execute code no matter whether or not an exception is thrown or caught.
What is the purpose of the Runtime class?

What is the purpose of the Runtime class?
The purpose of the Runtime class is to provide access to the Java runtime system.

What is the difference between the Boolean & operator and the && operator? If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the & operator is applied to the operand. When an expression involving the && operator is evaluated, the first operand is evaluated. If the first operand returns a value of true then the second operand is evaluated. The && operator is then applied to the first and second operands. If the first operand evaluates to false, the evaluation of the second operand is skipped.

What is the purpose of finalization?
The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.
In which package are most of the AWT events that support the event-delegation
Most of the AWT-related events of the event-delegation model are defined in the java.awt.event package. The AWTEvent class is defined in the java.awt package. Can an anonymous class be declared as implementing an interface and extending a class?
An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.
What class is the top of the AWT event hierarchy?
The java.awt.AWTEvent class is the highest-level class in the AWT event-class hierarchy.
What is a task's priority and how is it used in scheduling?

A task's priority is an integer value that identifies the relative order in which it should be executed with respect to other tasks. The scheduler attempts to schedule higher priority tasks before lower priority tasks.

What is the catch or declare rule for method declarations?
If a checked exception may be thrown within the body of a method, the method must either catch the exception or declare it in its throws clause
What are order of precedence and associativity, and how are they used?
Order of precedence determines the order in which operators are evaluated in expressions. Associatity determines whether an expression is evaluated left-to-right or right-to-left
What is the difference between preemptive scheduling and time slicing?

What is the difference between preemptive scheduling and time slicing?

**Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and
other factors.**

Does garbage collection guarantee that a program will not run out of memory?

Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection Which java.util classes and interfaces support event handling?

**The EventObject class and the EventListener interface support event processing.
What is the difference between yielding and sleeping?
When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.
How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?**

Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses 16-bit and larger bit patterns

**What is the Vector class?
The Vector class provides the capability to implement a growable array of objects
What is the List interface?
The List interface provides support for ordered collections of objects.
What is the Collections API?
The Collections API is a set of classes and interfaces that support operations on collections of objects.
What is the preferred size of a component?**

[illegible]

The preferred size of a component is the minimum component size that will allow the component to display normally.
Can a lock be acquired on a class?
Yes, a lock can be acquired on a class. This lock is acquired on the class's Class object.. What is synchronization and why is it important?

With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often leads to significant errors.

How are Observer and Observable used?

Objects that subclass the Observable class maintain a list of observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.

what is a transient variable? A transient variable is a variable that may not be serialized.

Interview Part 2

Question Can there be an abstract class with no abstract methods in it? (Core Java)

Answer: Yes

Question Can an Interface be final? (Core Java)

Answer: No

Question Can an Interface have an inner class? (Core Java)

Answer: Yes.

```
public interface abc {
static int i=0;
void dd();
class a1 {
a1() {
int j;
System.out.println("in interfia");
};
public static void main(String a1[]) {
System.out.println("in interfia"); } } }
```

Question Can we define private and protected modifiers for variables in interfaces? (Core Java)

Answer No.

Question What is the query used to display all tables names in SQL Server (Query analyzer)? (JDBC)

Answer select * from information_schema.tables

Question What is Externalizable? (Core Java)

Answer Externalizable is an Interface that extends Serializable Interface. And sends data into Streams in Compressed Format. It has two methods, writeExternal(ObjectOuput out) and readExternal(ObjectInput in)

Question What modifiers are allowed for methods in an Interface? (Core Java)

Answer Only public and abstract modifiers are allowed for methods in interfaces.

Question What is a local, member and a class variable? (Core Java)

Answer Variables declared within a method are "local" variables. Variables declared within the class i.e not within any methods are "member" variables (global variables). Variables declared within the class i.e not within any methods and are defined as "static" are class variables.

and are defined as "static" are class variables.

Question How many types of JDBC Drivers are present and what are they? (JDBC) Answer There are 4 types of JDBC Drivers Type 1: JDBC-ODBC Bridge Driver Type 2: Native API Partly Java Driver Type 3: Network protocol Driver Type 4: JDBC Net pure Java Driver

Question Can we implement an interface in a JSP? (JSP) AnswerNo
Question What is the difference between ServletContext and PageContext? (JSP) Answer ServletContext: Gives the information about the container PageContext: Gives the information about the Request
Question What is the difference in using request.getRequestDispatcher() and context.getRequestDispatcher()? (JSP)

Answer request.getRequestDispatcher(path): In order to create it we need to give the relative path of the resource context.getRequestDispatcher(path): In order to create it we need to give the absolute path of the resource.

Question How to pass information from JSP to included JSP? (JSP) Answer Using <%jsp:param> tag.
Question What is the difference between directive include and jsp include? (JSP)
Answer <%@ include> : Used to include static resources during translation time. : Used to include dynamic content or static content during runtime.
Question What is the difference between RequestDispatcher and sendRedirect? (JSP)
Answer RequestDispatcher: server-side redirect with request and response objects. sendRedirect : Client-side redirect with new request and response objects.
Question How does JSP handle runtime exceptions? (JSP)
Answer Using errorPage attribute of page directive and also we need to specify isErrorPage=true if the current page is intended to URL redirecting of a JSP.
Question How do you delete a Cookie within a JSP? (JSP)
Answer
Cookie mycook = new Cookie("name","value");
response.addCookie(mycook);
Cookie killmycook = new Cookie("mycook","value");
killmycook.setMaxAge(0);
killmycook.setPath("/");
killmycook.addCookie(killmycook);

Question How do I mix JSP and SSI #include? (JSP)
Answer If you're just including raw HTML, use the #include
directive as usual inside your .jsp file.
<!--#include file="data.inc"-->
But it's a little trickier if you want the server to evaluate any JSP code that's inside the included file. Ronel Sumibcay
(ronel@LIVESOFTWARE.COM) says: If your data.inc file contains
jsp code you will have to use
<%@ vinclude="data.inc" %> The <!--#include file="data.inc"--> is used for including non-JSP files.
Question I made my class Cloneable but I still get 'Can't access protected method clone. Why? (Core Java)

Answer Yeah, some of the Java books, in particular "The Java Programming Language", imply that all you have to do in order to have your class support clone() is implement the Cloneable interface.

Note: Not so. Perhaps that was the intent at some point, but that's not the way it works currently. As it stands, you have to implement your own public clone() method, even if it doesn't do anything special and just calls super.clone().

Question why is XML such an important development? (XML)
Answer It removes two constraints which were holding back Web developments:
1. Dependence on a single, inflexible document type (HTML) which
Was being much abused for tasks it was never designed for;
2. The complexity of full SGML, whose syntax allows many powerful

But hard-to-program options. XML allows the flexible development of user-defined document types. It provides a robust, non-proprietary, persistent, and verifiable file format for the storage and transmission of text and data both on and off the Web; and it removes the more complex options of SGML, making it easier to program for.

Question Are enterprise beans allowed to use Thread.sleep()? (EJB)

Answer Enterprise beans make use of the services provided by the EJB container, such as lifecycle management. To avoid conflicts with these services, enterprise beans are restricted from performing certain operations: Managing or synchronizing threads

Question Is it possible to write two EJB's that share the same Remote and Home interfaces, and have different bean classes? if so, what are the advantages/disadvantages? (EJB)

Answer It's certainly possible. In fact, there's an example that ships with the Inprise Application Server of an Account interface with separate implementations for CheckingAccount and SavingsAccount, one of which was CMP and one of which was BMP.

Question Is it possible to specify multiple JNDI names when deploying an EJB? (EJB)

Answer No. To achieve this you have to deploy your EJB multiple times each specifying a different JNDI name.

Question Is there any way to force an Entity Bean to store itself to the db? I don't wanna wait for the container to update the db, I want to do it NOW! Is it possible? (EJB)

Answer Specify the transaction attribute of the bean as RequiresNew. Then as per section 11.6.2.4 of the EJB v 1.1 spec EJB container automatically starts a new transaction before the method call. The container also performs the commit protocol before the method result is sent to the client.

Question I am developing a BMP Entity bean. I have noticed that whenever the create method is invoked, the ejbLoad() and the ejbStore() methods are also invoked. I feel that once my database insert is done, having to do a select and update SQL queries is major overhead. is this behavior typical of all EJB containers? Is there any way to suppress these invocations? (EJB)

Answer This is the default behaviour for EJB. The specification states that ejbLoad() will be called before every transaction and ejbStore() after every transaction. Each Vendor has optimizations, which are proprietary for this scenario.

Question Can an EJB send asynchronous notifications to its clients? (EJB)

Answer Asynchronous notification is a known hole in the first versions of the EJB spec. The recommended solution to this is to use JMS, which is becoming available in J2EE-compliant servers. The other option, of course, is to use client-side threads and polling. This is not an ideal solution, but it's workable for many scenarios.

Question How can I access EJB from ASP? (EJB)
Answer You can use the Java 2 Platform, Enterprise Edition Client Access Services (J2EETM CAS) COM Bridge 1.0, currently downloadable from **http://developer.java.sun.com/developer/earlyAccess/j2eecas/**
Question Is there a guarantee of uniqueness for entity beans? (EJB)

Answer There is no such guarantee. The server (or servers) can instantiate as many instances of the same underlying Entity Bean (with the same PK) as it wants. However, each instance is guaranteed to have up-to-date data values, and be transactionally consistent, so uniqueness is not required. This allows the server to scale the system to support multiple threads, multiple concurrent requests, and multiple hosts.

Question How do the six transaction attributes map to isolation levels like "dirty read"? Will an attribute like "Required" lock out other readers until I'm finished updating? (EJB)

Answer The Transaction Attributes in EJB do not map to the Transaction Isolation levels used in JDBC. This is a common misconception. Transaction Attributes specify to the container when a Transaction should be started, suspended(paused) and committed between method invocations on Enterprise JavaBeans. For more details and a summary of Transaction Attributes refer to section 11.6 of the EJB 1.1 specification.

Question I have created a remote reference to an EJB in FirstServlet. Can I put the reference in a servlet session and use that in SecondServlet? (EJB)

Answer Yes. The EJB client (in this case your servlet) acquires a remote reference to an EJB from the Home Interface; that reference is serializable and can be passed from servlet to servlet. If it is a session bean, then the EJB server will consider your web client's servlet session to correspond to a single EJB session, which is usually (but not always) what you want.

Question Can the primary key in the entity bean be a Java primitive type such as int? (EJB)

Answer The primary key can't be a primitive type--use the primitive wrapper classes, instead. For example, you can use java.lang.Integer as the primary key class, but not int (it has to be a class, not a primitive)

Question What's new in the EJB 2.0 specification? (EJB)

Answer Following are the main features supported in EJB 2.0 * Integration of EJB with JMS *Message Driven Beans* Implement additional Business methods in Home interface which are not specific for bean instance. * EJB QL.

Question How many types of protocol implementations does RMI have? (RMI)

Answer RMI has at least three protocol implementations: Java Remote Method Protocol(JRMP), Internet Inter ORB Protocol(IIOP), and Jini Extensible Remote Invocation(JERI). These are alternatives, not part of the same thing, All three are indeed layer 6 protocols for those who are still speaking OSI reference model.

Question What are the different identifier states of a Thread? (Core Java)
Answer
The different identifiers of a Thread are:
R - Running or runnable thread
S - Suspended thread
CW - Thread waiting on a condition variable MW - Thread waiting on a monitor lock
MS - Thread suspended waiting on a monitor lock
Question What is the need of Remote and Home interface. Why cant it be in one? (EJB)

Answer In a few words, I would say that the main reason is because there is a clear division of roles and responsibilities between the two interfaces. The home interface is your way to communicate with the container, that is who is responsible of creating, locating even removing one or more beans. The remote interface is your link to the bean, that will allow you to remotely access to all its methods and members. As you can see there are two distinct elements (the container and the beans) and you need two different interfaces for accessing to both of them.

Question What is the difference between Java Beans and EJB?s? (EJB)
Answer Java Beans are client-side objects and EJBs are server side object, and they have completely different development, lifecycle, purpose.

Question QuestionWith regard to Entity Beans, what happens if both my EJB Server and Database crash, what will happen to unsaved changes? Is

there any transactional log file used? (EJB)

Answer Actually, if your EJB server crashes, you will not even be able to make a connection to the server to perform a bean lookup, as the server will no longer be listening on the port for incoming JNDI lookup requests. You will lose any data that wasn't committed prior to the crash. This is where you should start looking into clustering your EJB server.

Another Answer

Hi, Any unsaved and uncommitted changes are lost the moment your EJB Server crashes. If your database also crashes, then all the saved changes are also lost unless you have some backup or some recovery mechanism to retrieve the data. So consider database replication and EJB Clustering for such scenarios, though the occurrence of such a thing is very very rare. All database have the concept of log files(for example oracle have redo log files concept). So if data bases crashes then on starting up they fill look up the log files to perform all pending jobs. But is EJB crashes, It depend upon the container how frequently it passivates or how frequently it refreshes the data with Database.

Question Can you control when passivation occurs? (EJB)

Answer The developer, according to the specification, cannot directly control when passivation occurs. Although for Stateful Session Beans, the container cannot passivate an instance that is inside a transaction. So using transactions can be a a strategy to control passivation. The ejbPassivate() method is called during passivation, so the developer has control over what to do during this exercise and can implement the require optimized logic. Some EJB containers, such as BEA WebLogic, provide the ability to tune the container to minimize passivation calls. Taken from the WebLogic 6.0 DTD

"The passivation-strategy can be either "default" or "transaction". With the default setting the container will attempt to keep a working set of beans in the cache. With the "transaction" setting, the container will passivate the bean after every transaction (or method call for a nontransactional invocation)."

Question Does RMI-IIOP support dynamic downloading of classes? (RMI)

Answer No, RMI-IIOP doesn't support dynamic downloading of the classes as it is done with CORBA in DII (Dynamic Interface Invocation).Actually RMI-IIOP combines the usability of Java Remote Method Invocation (RMI)with the interoperability of the Internet InterORB Protocol (IIOP).So in order to attain this interoperability between RMI and CORBA,some of the features that are supported by RMI but not CORBA and vice versa are eliminated from the RMI-IIOP specification.

Question Does EJB 1.1 support mandate the support for RMI-IIOP ? What is the meaning of "the client API must support the Java RMI-IIOP programming model for portability, but the underlying protocol can be anything”? (EJB)

Answer EJB1.1 does mandate the support of RMI-IIOP. OK, to Answer the second Question:

There are 2 types of implementations that an EJB Server might provide: CORBA-based EJB Servers and Proprietry EJB Servers. Both support the RMI-IIOP API but how that API is implemented is a different story. (NB: By API we mean the interface provided to the client by the stub or proxy). A CORBA-based EJB Server actually implements its EJB Objects as CORBA Objects (it therefore encorporates an ORB and this means that EJB's can be contacted by CORBA clients (as well as RMI-IIOP clients)

A proprietry EJB still implements the RMI-IIOP API (in the client's stub) but the underlying protocol can be anything. Therefore your EJB's CANNOT be contacted by CORBA clients. The difference is that in both cases, your clients see the same API (hence, your client portability) BUT how the stubs communicate with the server is different.

Question The EJB specification says that we cannot use Bean Managed Transaction in Entity Beans. Why? (EJB)

Answer The short, practical Answeris... because it makes your entity beans useless as a reusable component. Also, transaction management is best left to the application server - that's what they're there for. It's all about atomic operations on your data. If an operation updates more than one entity then you want the whole thing to succeed or the whole thing to fail, nothing in between. If you put commits in the entity beans then it's very difficult to rollback if an error occurs at some point late in the operation.

Question Can I invoke Runtime.gc() in an EJB? (EJB)

Answer You shouldn't. What will happen depends on the implementation, but the call will most likely be ignored. You should leave system level management like garbage collection for the container to deal with. After all, that's part of the benefit of using EJBs, you don't have to manage resources yourself.

Question What is clustering? What are the different algorithms used for clustering? (EJB)

Answer Clustering is grouping machines together to transparantly provide enterprise services.The client does not now the difference between approaching one server or approaching a cluster of servers.Clusters provide two benefits: scalability and high availability. Further information can be found in the JavaWorld article J2EE Clustering.

Question What is the advantage of using Entity bean for database operations, over directly using JDBC API to do database operations? When would I use one over the other? (EJB)

Answer Entity Beans actually represents the data in a database. It is not that Entity Beans replaces JDBC API. There are two types of Entity Beans Container Managed and Bean Mananged. In Container Managed Entity Bean - Whenever the instance of the bean is created the container automatically retrieves the data from the DB/Persistance storage and assigns to the object variables in bean for user to manipulate or use them. For this the developer needs to map the fields in the database to the variables in deployment descriptor files (which varies for each vendor). In the Bean Managed Entity Bean - The developer has to specifically make connection, retrive values, assign them to the objects in the ejbLoad() which will be called by the container when it instatiates a bean object. Similarly in the ejbStore() the container saves the object values back the the persistance storage. ejbLoad and ejbStore are callback methods and can be only invoked by the container. Apart from this, when you use Entity beans you dont need to worry about database transaction handling, database connection pooling *etc.* which are taken care by the ejb container. But in case of JDBC you have to explicitly do the above features. what suresh told is exactly perfect. ofcourse, this comes under the database transations, but i want to add this. the great thing about the entity beans of container managed, whenever the connection is failed during the transaction processing, the database consistancy is mantained automatically. the container writes the data stored at persistant storage of the entity beans to the database again to provide the database consistancy. where as in jdbc api, we, developers has to do manually.

Question What is the role of serialization in EJB? (EJB)

Answer A big part of EJB is that it is a framework for underlying RMI: remote method invocation. You're invoking methods remotely from JVM space 'A' on objects which are in JVM space 'B' -- possibly running on another machine on the network. To make this happen, all arguments of each method call must have their current state plucked out of JVM 'A' memory, flattened into a byte stream which can be sent over a TCP/IP network connection, and then deserialized for reincarnation on the other end in JVM 'B' where the actual method call takes place. If the method has a return value, it is serialized up for streaming back to JVM A. Thus the requirement that all EJB methods arguments and return values must be serializable. The easiest way to do this is to make sure all your classes implement java.io.Serializable.

Question What is EJB QL? (EJB)

Answer EJB QL is a Query Language provided for navigation across a network of enterprise beans and dependent objects defined by means of container managed persistence. EJB QL is introduced in the EJB 2.0 specification. The EJB QL query language defines finder methods for entity beans with container managed persistenceand is portable across containers and persistence managers. EJB QL is used for queries of two types of finder methods: Finder methods that are defined in the home interface of an entity bean and which return entity objects. Select methods, which are not exposed to the client, but which are used by the Bean Provider to select persistent values that are maintained by the Persistence Manager or to select entity objects that are related to the entity bean on which the query is defined.

Question What is the fastest type of JDBC driver? (JDBC) Answer JDBC driver performance will depend on a number of issues: (a) the quality of the driver code, (b) the size of the driver code, (c) the database server and its load, (d) network topology,

(e) the number of times your request is translated to a different API. In general, all things being equal, you can assume that the more your request and response change hands, the slower it will be. This means that Type 1 and Type 3 drivers will be slower than Type 2 drivers (the database calls are make at least three translations versus two), and Type 4 drivers are the fastest (only one translation).

Question Request parameter How to find whether a parameter exists in the request object? (Servlets)

Answer 1.boolean hasFoo = !(request.getParameter("foo") == null || request.getParameter("foo").equals("")); 2. boolean hasParameter = request.getParameterMap().contains(theParameter); (which works in Servlet 2.3+)

Question How can I send user authentication information while makingURLConnection? (Servlets)
Answer You'll want to use HttpURLConnection.setRequestProperty and set all the appropriate headers to HTTP authorization.
Question What are some alternatives to inheritance? (Core Java)

Answer Delegation is an alternative to inheritance. Delegation means that you include an instance of another class as an instance variable, and forward messages to the instance. It is often safer than inheritance because it forces you to think about each message you forward, because the instance is of a known class, rather than a new class, and because it doesn't force you to accept all the methods of the super class: you can provide only the methods that really make sense. On the other hand, it makes you write more code, and it is harder to re-use (because it is not a subclass).

Question Why isn't there operator overloading? (Core Java)

Answer Because C++ has proven by example that operator overloading makes code almost impossible to maintain. In fact there very nearly wasn't even method overloading in Java, but it was thought that this was too useful for some very basic methods like print(). Note that some of the classes like DataOutputStream have unoverloaded methods like writeInt() and writeByte().

Question What does it mean that a method or field is "static"? (Core Java)

Answer Static variables and methods are instantiated only once per class. In other words they are class variables, not instance variables. If you change the value of a static variable in a particular object, the value of that variable changes for all instances of that class. Static methods can be referenced with the name of the class rather than the name of a particular object of the class (though that works too). That's how library methods like System.out.println() work. out is a static field in the java.lang.System class.

Question How do I convert a numeric IP address like 192.18.97.39 into a hostname like java.sun.com? (Networking)
Answer String hostname =
InetAddress.getByName("192.18.97.39").getHostName();
Question Difference between JRE And JVM AND JDK (Core Java) Answer: Read Above in Introduction at Beginning of this chapter.
Question Why do threads block on I/O? (Core Java)

Answer Threads block on i/o (that is enters the waiting state) so that other threads may execute while the i/o Operation is performed. Question What is synchronization and why is it important? (Core Java) Answer With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often leads to significant errors.

Question Is null a keyword? (Core Java) Answer The null value is not a keyword.
Question Which characters may be used as the second character of an identifier,but not as the first character of an identifier? (Core Java)
Answer The digits 0 through 9 may not be used as the first character of an identifier but they may be used after the first character of an identifier
Question What modifiers may be used with an inner class that is a member of an outer class? (Core Java)
Answer A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.
Question How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters? (Core Java)

Answer Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses 16-bit and larger bit patterns.

Question What are wrapped classes? (Core Java) Answer Wrapped classes are classes that allow primitive types to be accessed as objects.
Question What restrictions are placed on the location of a package

statement within a source code file? (Core Java)
Answer A package statement must appear as the first line in a source code file (excluding blank lines and comments).
Question What is the difference between preemptive scheduling and time slicing? (Core Java)

Answer Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.

Question What is a native method? (Core Java) Answer A native method is a method that is implemented in a language other than Java.
Question What are order of precedence and associativity, and how are they used? (Core Java)

Answer Order of precedence determines the order in which operators are evaluated in expressions. Associatity determines whether an expression is evaluated left-to-right or right-toleft Question What is the catch or declare rule for method declarations? (Core Java)Answer If a checked exception may be thrown within the body of a method, the method must either catch the exception or declare it in its throws clause.

Question Can an anonymous class be declared as implementing an interface and extending a class? (Core Java)
Answer An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.
Question What is the range of the char type? (Core Java) Answer The range of the char type is 0 to 2^16 - 1.
Question What is the purpose of finalization? (Core Java) Answer The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.
Question What is the difference between the Boolean & operator and the && operator? (Core Java)

Answer If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the & operator is applied to the operand. When an expression involving the && operator is evaluated, the first operand is evaluated. If the first operand returns a value of true then the second operand is evaluated. The && operator is then applied to the first and second operands. If the first operand evaluates to false, the evaluation of the second operand is skipped.

Question How many times may an object's finalize() method be invoked by the garbage collector? (Core Java)
Answer An object's finalize() method may only be invoked once by the garbage collector.
Question What is the purpose of the finally clause of a try-catchfinally statement? (Core Java) Answer The finally clause is used to provide the capability to execute code no matter whether or not an exception is thrown or caught.
Question What is the argument type of a program's main() method? (Core Java) Answer A program's main() method takes an argument of the String[] type.
Question Which Java operator is right associative? (Core Java) Answer The = operator is right associative.
Question Can a double value be cast to a byte? (Core Java) Answer Yes, a double value can be cast to a byte.
Question What is the difference between a break statement and a continue statement? (Core Java)

Answer A break statement results in the termination of the statement to which it applies (switch, for, do, or while). A continue statement is used to end the current loop iteration and return control to the loop statement.

Question What must a class do to implement an interface? (Core Java)
Answer It must provide all of the methods in the interface and identify the interface in its implements clause.
Question What is the advantage of the eventdelegation model over the earlier eventinheritance model? (Core Java)

Answer The eventdelegation model has two advantages over the eventinheritance model. First, it enables event handling to be handled by objects other than the ones that generate the events (or their containers). This allows a clean separation between a component's design and its use. The other advantage of the eventdelegation model is that it performs much better in applications where many events are generated. This performance improvement is due to the fact that the eventdelegation model does not have to repeatedly process unhandled events, as is the case of the eventinheritance model.

Question How are commas used in the intialization and iteration parts of a

for statement? (Core Java)
Answer Commas are used to separate multiple statements within the initialization and iteration parts of a for statement.
Question What is an abstract method? (Core Java) Answer An abstract method is a method whose implementation is deferred to a subclass.
Question What value does read() return when it has reached the end of a file? (Core Java) Answer The read() method returns -1 when it has reached the end of a file.
Question Can a Byte object be cast to a double value? (Core Java) Answer No, an object cannot be cast to a primitive value.
Question What is the difference between a static and a nonstatic inner class? (Core Java)
Answer A nonstatic inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.
Question If a variable is declared as private, where may the variable be accessed? (Core Java)
Answer A private variable may only be accessed within the class in which it is declared.
Question What is an object's lock and which object's have locks? (Core Java)

Answer An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the object. A thread may execute a synchronized method of an object only after it has acquired the object's lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.

Question What is the % operator? (Core Java) Answer It is referred to as the modulo or remainder operator. It returns the remainder of dividing the first operand by the second operand.
Question When can an object reference be cast to an interface reference? (Core Java) Answer An object reference be cast to an interface reference when the object implements the referenced interface.
Question Which class is extended by all other classes? (Core Java) Answer The Object class is extended by all other classes.
Question Can an object be garbage collected while it is still reachable? (Core Java)
Answer A reachable object cannot be garbage collected. Only unreachable objects may be garbage collected.

Question Is the ternary operator written x : y ? z or x ? y : z ? (Core Java)
Answer It is written x ? y : z.
Question How is rounding performed under integer division? (Core Java)
Answer The fractional part of the result is truncated. This is known as rounding toward zero.
Question What is the difference between the Reader/Writer class hierarchy and the InputStream/OutputStream class hierarchy? (Core Java)
Answer The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class hierarchy is byte-oriented.
Question What classes of exceptions may be caught by a catch clause? (Core Java) Answer A catch clause can catch any exception that may be assigned to the Throwable type. This includes the Error and Exception types.
Question If a class is declared without any access modifiers, where may the class be accessed? (Core Java)

Answer A class that is declared without any access modifiers is said to have package access. This means that the class can only be accessed by other classes and interfaces that are defined within the same package.

Question Does a class inherit the constructors of its superclass? (Core Java)
Answer A class does not inherit constructors from any of its superclasses.
Question What is the purpose of the System class? (Core Java) Answer The purpose of the System class is to provide access to system resources.
Question Name the eight primitive Java types. (Core Java) Answer The eight primitive types are byte, char, short, int, long, float, double, and boolean.
Question Which class should you use to obtain design information about an object? (Core Java)
Answer The Class class is used to obtain information about an object's design.
Question Is "abc" a primitive value? (Core Java) Answer The String literal "abc" is not a primitive value. It is a String object.
Question What restrictions are placed on the values of each case of a switch statement? (Core Java)
Answer During compilation, the values of each case of a switch statement must evaluate to a value that can be promoted to an int value.
Question What modifiers may be used with an interface declaration? (Core Java) Answer An interface may be declared as public or abstract.

Question Is a class a subclass of itself? (Core Java) Answer A class is a subclass of itself. **Question What is the difference between a while statement**

and a do statement? (Core Java)

Answer A while statement checks at the beginning of a loop to see whether the next loop iteration should occur. A do statement checks at the end of a loop to see whether the next iteration of a loop should occur. The do statement will always execute the body of a loop at least once.

Question What modifiers can be used with a local inner class? (Core Java)
Answer A local inner class may be final or abstract.
Question What is the purpose of the File class? (Core Java) Answer The File class is used to create objects that provide access to the files and directories of a local file system.
Question Can an exception be rethrown? (Core Java) Answer Yes, an exception can be rethrown.
Question When does the compiler supply a default constructor for a class? (Core Java) Answer The compiler supplies a default constructor for a class if no other constructors are provided.
Question If a method is declared as protected, where may the method be accessed? (Core Java)
Answer A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.
Question Which non-Unicode letter characters may be used as the first character of an identifier? (Core Java)
Answer The non-Unicode letter characters $ and _ may appear as the first character of an identifier.
Question What restrictions are placed on method overloading? (Core Java)
Answer Two methods may not have the same name and argument list but different return types.
Question What is casting? (Core Java)

Answer There are two types of casting, casting between primitive numeric types and casting between object references. Casting between numeric types is used to convert larger values, such as double values, to smaller values, such as byte values. Casting between object references is used to refer to an object by a compatible class, interface, or array type reference.

Question What is the return type of a program's main() method? (Core Java) Answer A program's main() method has a void return type.
Question What class of exceptions are generated by the Java runtime

system? (Core Java) Answer The Java runtime system generates RuntimeException and Error exceptions.
Question What class allows you to read objects directly from a stream? (Core Java) Answer The ObjectInputStream class supports the reading of objects from input streams.
Question What is the difference between a field variable and a local variable? (Core Java)
Answer A field variable is a variable that is declared as a member of a class. A local variable is a variable that is declared local to a method.
Question How are this() and super() used with constructors? (Core Java)
Answer this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.
Question What is the relationship between a method's throws clause and the exceptions that can be thrown during the method's execution? (Core Java)
Answer A method's throws clause must declare any checked exceptions that are not caught within the body of the method.
Question Why are the methods of the Math class static? (Core Java) Answer So they can be invoked as if they are a mathematical code library.
Question What are the legal operands of the instanceof operator? (Core Java)
Answer The left operand is an object reference or null value and the right operand is a class, interface, or array type.
Question What an I/O filter? (Core Java) Answer An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.
Question If an object is garbage collected, can it become reachable again? (Core Java) Answer Once an object is garbage collected, it ceases to exist. It can no longer become reachable again.
Question What are E and PI? (Core Java) Answer E is the base of the natural logarithm and PI is mathematical value pi.
Question Are true and false keywords? (Core Java) Answer The values true and false are not keywords.
Question What is the difference between the File and RandomAccessFile classes? (Core Java)

Answer The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.

Question What happens when you add a double value to a String? (Core Java) Answer The result is a String object.
Question What is your platform's default character encoding? (Core Java)
Answer If you are running Java on English Windows platforms, it is probably Cp1252. If you are running Java on English Solaris platforms, it is most likely 8859_1.
Question Which package is always imported by default? (Core Java)
Answer The java.lang package is always imported by default.
Question What interface must an object implement before it can be written to a stream as an object? (Core Java)
Answer An object must implement the Serializable or Externalizable interface before it can be written to a stream as an object.
Question How can my application get to know when a HttpSession is removed? (JSP)

Answer Define a Class HttpSessionNotifier which implements HttpSessionBindingListener and implement the functionality what you need in valueUnbound() method. Create an instance of that class and put that instance in HttpSession.

Question Whats the difference between notify() and notifyAll()? (Core Java)

Answer notify() is used to unblock one waiting thread; notifyAll() is used to unblock all of them. Using notify() is preferable (for efficiency) when only one blocked thread can benefit from the change (for example, when freeing a buffer back into a pool). notifyAll() is necessary (for correctness) if multiple threads should resume (for example, when releasing a "writer" lock on a file might permit all "readers" to resume).

Question Why can't I say just abs() or sin() instead of Math.abs() and Math.sin()? (Core Java)

Answer The import statement does not bring methods into your local name space. It lets you abbreviate class names, but not get rid of them altogether. That's just the way it works, you'll get used to it. It's really a lot safer this way. <br> However, there is actually a little trick you can use in some cases that gets you what you want. If your top-level class doesn't need to inherit from anything else, make it inherit from java.lang.Math. That *does* bring all the methods into your local name space. But you can't use this trick in an applet, because you have to inherit from java.awt.Applet. And actually, you can't use it on java.lang.Math at all, because Math is a "final" class which means it can't be extended.

Question Is is possible for an EJB client to marshall an object of class java.lang.Class to an EJB? (EJB)

Answer Technically yes, spec. compliant NO! - The enterprise bean must not attempt to query a class to obtain information about the declared members that are not otherwise accessible to the enterprise bean because of the security rules of the Java language.

Question Is it legal to have static initializer blocks in EJB? (EJB)

Answer Although technically it is legal, static initializer blocks are used to execute some piece of code before executing any constructor or method while instantiating a class. Static initializer blocks are also typically used to initialize static fields - which may be illegal in EJB if they are read/write - In EJB this can be achieved by including the code in either the ejbCreate(), setSessionContext() or setEntityContext() methods.

Question How can I implement a thread-safe JSP page? (JSP)

Answer You can make your JSPs thread-safe by having them implement the SingleThreadModel interface. This is done by adding the directive <%@ page isThreadSafe="false" % > within your JSP page.

Question Is it possible to stop the execution of a method before completion in a SessionBean? (EJB)

Answer Stopping the execution of a method inside a Session Bean is not possible without writing code inside the Session Bean. This is because you are not allowed to access Threads inside an EJB.

Question What is the default transaction attribute for an EJB? (EJB)

Answer There is no default transaction attribute for an EJB. Section 11.5 of EJB v1.1 spec says that the deployer must specify a value for the transaction attribute for those methods having container managed transaction. In weblogic, the default transaction attribute for EJB is SUPPORTS.

Question What is the difference between session and entity beans? When should I use one or the other? (EJB)

Answer An entity bean represents persistent global data from the database; a session bean represents transient user-specific data that will die when the user disconnects (ends his session). Generally, the session beans implement business methods (e.g. Bank.transferFunds) that call entity beans (e.g. Account.deposit, Account.withdraw)

Question Is there any default cache management system with Entity beans ? In other words whether a cache of the data in database will be maintained in EJB ? (EJB)

Answer Caching data from a database inside the Application Server are what Entity EJB's are used for.The ejbLoad() and ejbStore() methods are used to synchronize the Entity Bean state with the persistent storage(database). Transactions also play an important role in this scenario. If data is removed from the database, via an external application - your Entity Bean can still be "alive" the EJB container. When the transaction commits, ejbStore() is called and the row will not be found, and the transcation rolled back.

Question Why is ejbFindByPrimaryKey mandatory? (EJB)

Answer An Entity Bean represents persistent data that is stored outside of the EJB Container/Server. The ejbFindByPrimaryKey is a method used to locate and load an Entity Bean into the container, similar to a SELECT statement in SQL. By making this method mandatory, the client programmer can be assured that if they have the primary key of the Entity Bean, then they can retrieve the bean without having to create a new bean each time - which would mean creating duplications of persistent data and break the integrity of EJB.

Question Why do we have a remove method in both EJBHome and EJBObject? (EJB)

Answer With the EJBHome version of the remove, you are able to delete an entity bean without first instantiating it (you can provide a PrimaryKey object as a parameter to the remove method). The home version only works for entity beans. On the other hand, the Remote interface version works on an entity bean that you have alreadyinstantiated. In addition, the remote version also works on session beans (stateless and statefull) to inform the container of your loss of interest in this bean.

Question How can I call one EJB from inside of another EJB? (EJB)
Answer EJBs can be clients of other EJBs. It just works. Use JNDI to locate the Home Interface of the other bean, then acquire an instance reference, and so forth.
Question What is the difference between a Server, a Container, and a Connector? (EJB)

Answer An EJB server is an application, usually a product such as BEA WebLogic, that provides (or should provide) for concurrent client connections and manages system resources such as threads, processes, memory, database connections, network connections, *etc.* An EJB container runs inside (or within) an EJB server, and provides deployed EJB beans with transaction and security management, *etc.* The EJB container insulates an EJB bean from the specifics of an underlying EJB server by providing a simple, standard API between the EJB bean and its container.A Connector provides the ability for any Enterprise Information System (EIS) to plug into any EJB server which supports the Connector architecture. See http://java.sun.com/j2ee/connector/ for more indepth information on Connectors.

Question How is persistence implemented in enterprise beans? (EJB)

Answer Persistence in EJB is taken care of in two ways, depending on how you implement your beans: container managed persistence (CMP) or bean managed persistence (BMP) For CMP, the EJB container which your beans run under takes care of the persistence of the fields you have declared to be persisted with the database - this declaration is in the deployment descriptor. So, anytime you modify a field in a CMP bean, as soon as the method you have executed is finished, the new data is persisted to the database by the container. For BMP, the EJB bean developer is responsible for defining the persistence routines in the proper places in the bean, for instance, the ejbCreate(), ejbStore(), ejbRemove() methods would be developed by the bean developer to make calls to the database. The container is responsible, in BMP, to call the appropriate method on the bean. So, if the bean is being looked up, when the create() method is called on the Home interface, then the container is responsible for calling the ejbCreate() method in the bean, which should have functionality inside for going to the database and looking up the data.

Question What is an EJB Context? (EJB)

Answer EJBContext is an interface that is implemented by the container, and it is also a part of the bean-container contract. Entity beans use a subclass of EJBContext called EntityContext. Session beans use a subclass called SessionContext. These EJBContext objects provide the bean class with information about its container, the client using the bean and the bean itself. They also provide other functions. See the API docs and the spec for more details.

Question Is method overloading allowed in EJB? (EJB) Answer Yes you can overload methods

Question Should synchronization primitives be used on bean methods? (EJB)

Answer No. The EJB specification specifically states that the enterprise bean is not allowed to use thread primitives. The container is responsible for managing concurrent access to beans at runtime.

Question Are we allowed to change the transaction isolation property in middle of a transaction? (EJB)

Answer No. You cannot change the transaction isolation level in the middle of transaction.

Question For Entity Beans, What happens to an instance field not mapped to any persistent storage,when the bean is passivated? (EJB)

Answer The specification infers that the container never serializes an instance of an Entity bean (unlike stateful session beans). Thus passivation simply involves moving the bean from the "ready" to the "pooled" bin. So what happens to the contents of an instance variable is controlled by the programmer. Remember that when an entity bean is passivated the instance gets logically disassociated from it's remote object. Be careful here, as the functionality of passivation/activation for Stateless Session, Stateful Session and Entity beans is completely different. For entity beans the ejbPassivate method notifies the entity bean that it is being disassociated with a particular entity prior to reuse or for dereferenc.

Question What is a Message Driven Bean, What functions does a message driven bean have and how do they work in collaboration with JMS? (EJB)

Answer: Message driven beans are the latest addition to the family of component bean types defined by the EJB specification. The original bean types include session beans, which contain business logic and maintain a state associated with client sessions, and entity beans, which map objects to persistent data. Message driven beans will provide asynchrony to EJB based applications by acting as JMS message consumers. A message bean is associated with a JMS topic or queue and receives JMS messages sent by EJB clients or other beans. Unlike entity beans and session beans, message beans do not have home or remote interfaces. Instead, message driven beans are instantiated by the container as required. Like stateless session beans, message beans maintain no client-specific state, allowing the container to optimally manage a pool of message-bean instances. Clients send JMS messages to message beans in exactly the same manner as they would send messages to any other JMS destination. This similarity is a fundamental design goal of the JMS capabilities of the new specification. To receive JMS messages, message driven beans implement the javax.jms.MessageListener interface, which defines a single "onMessage()" method. When a message arrives, the container ensures that a message bean corresponding to the message topic/queue exists (instantiating it if necessary), and calls its onMessage method passing the client's message as the single argument. The message bean's implementation of this method contains the business logic required to process the message.

Note that session beans and entity beans are not allowed to function as message beans.
Question Does RMI-IIOP support code downloading for Java objects sent by value across an IIOP connection in the same way as RMI does across a JRMP connection? (RMI) Answer Yes. The JDK 1.2 supports the dynamic class loading.

Question The EJB container implements the EJBHome and EJBObject classes. For every request from a unique client, does the container create a separate instance of the generated EJBHome and EJBObject classes? (EJB)

Answer The EJB container maintains an instance pool. The container uses these instances for the EJB Home reference irrespective of the client request. while refering the EJB Object classes the container creates a separate instance for each client request.

Another Answer

The instance pool maintainence is up to the implementation of the container. If the container provides one, it is available otherwise it is not mandatory for the provider to implement it. Having said that, yes most of the container providers implement the pooling functionality to increase the performance of the app server. How it is implemented, it is again up to the implementer.

Question What is the advantage of puttting an Entity Bean instance from the "Ready State" to "Pooled state"? (EJB)

Answer The idea of the "Pooled State" is to allow a container to maintain a pool of entity beans that has been created, but has not been yet "synchronized" or assigned to an EJBObject. This mean thatthe instances do represent entity beans, but they can be used only for serving Home methods (create or findBy), since those methods do not relay on the specific values of the bean. All these instances are, in fact, exactly the same, so, they do not have meaningful state. Jon Thorarinsson has also added: It can be looked at it this way: If no client is using an entity bean of a particular type there is no need for cachig it (the data is persisted in the database). Therefore, in such cases, the container will, after some time, move the entity bean from the "Ready State" to the "Pooled state" to save memory. Then, to save additional memory, the container may begin moving entity beans from the "Pooled State" to the "Does Not Exist State", because even though the bean's cache has been cleared, the bean still takes up some memory just being in the "Pooled State".

Question Can a Session Bean be defined without ejbCreate() method? (EJB)

Answer The ejbCreate() methods is part of the bean's lifecycle, so, the compiler will not return an error because there is no ejbCreate() method. However, the J2EE spec is explicit: the home interface of a Stateless Session Bean must have a single create() method with no arguments, while the session bean class must contain exactly one ejbCreate() method, also without arguments. Stateful Session Beans can have arguments (more than one create method) stateful beans can contain multiple ejbCreate() as long as they match with the home interface definition You need a reference to your EJBObject to startwith. For that Sun insists on putting a method for creating that reference (create methodin the home interface). The EJBObject does matter here. Not the actual bean.

Question Is it possible to share an HttpSession between a JSP and EJB? What happens when I change a value in the HttpSession from inside an EJB? (EJB)

Answer You can pass the HttpSession as parameter to an EJB method, only if all objects in session are serializable.This has to be consider as "passed-by-value", that means that it's readonly in the EJB. If anything is altered from inside the EJB, it won't be reflected back to the HttpSession of the Servlet Container.The "pass-byreference" can be used between EJBs Remote Interfaces, as they are remote references. While it IS possible to pass an HttpSession as a parameter to an EJB object, it is considered to be "bad practice (1)" in terms of object oriented design. This is because you are creating an unnecessary coupling between back-end objects (ejbs) and frontend objects (HttpSession). Create a higher-level of abstraction for your ejb's api. Rather than passing the whole, fat, HttpSession (which carries with it a bunch of http semantics), create a class that acts as a value object (or structure) that holds all the data you need to pass back and forth between frontend/back-end. Consider the case where your ejb needs to support a non-http-based client. This higher level of abstraction will be flexible enough to support it. (1) Core J2EE design patterns (2001)

Question Is there any way to read values from an entity bean without locking it for the rest of the transaction (e.g. readonly transactions)? We have a key-value map bean which deadlocks during some concurrent reads. Isolation levels seem to affect the database only, and we need to work within a transaction. (EJB)

Answer The only thing that comes to (my) mind is that you could write a 'group accessor' - a method that returns a single object containing all of your entity bean's attributes (or all interesting attributes). This method could then be placed in a 'Requires New' transaction. This way, the current transaction would be suspended for the duration of the call to the entity bean and the entity bean's fetch/operate/commit cycle will be in a separate transaction and any locks should be released immediately. Depending on the granularity of what you need to pull out of the map, the group accessor might be overkill.

Question What is the difference between a "Coarse Grained" Entity Bean and a "Fine Grained" Entity Bean? (EJB)

Answer A 'fine grained' entity bean is pretty much directly mapped to one relational table, in third normal form. A 'coarse grained' entity bean is larger and more complex, either because its attributes include values or lists from other tables, or because it 'owns' one or more sets of dependent objects.

Note that the coarse grained bean might be mapped to a single table or flat file, but that single table is going to be pretty ugly, with data copied from other tables, repeated field groups, columns that are dependent on non-key fields, *etc.* Fine grained entities are generally considered a liability in large systems because they will tend to increase the load on several of the EJB server's subsystems (there will be more objects exported through the distribution layer, more objects participating in transactions, more skeletons in memory, more EJB Objects in memory, etc.) The other side of the coin is that the 1.1 spec doesn't mandate CMP Error! No index entries found.support for dependent objects (or even indicate how they should be supported), which makes it more difficult to do coarse grained objects with CMP. The EJB 2.0 specification improves this in a huge way.

Question What is EJBDoclet? (EJB)
Answer EJBDoclet is an open source JavaDoc doclet that generates a lot of the EJB related source files from custom JavaDoc comments tags embedded in the EJB source file.
Question Wha is the output from System.out.println("Hello"+null); (Core Java) Answer Hellonull
Question Can we use the constructor, instead of init(), to initialize servlet? (Servlets)

Answer Yes , of course you can use the constructor instead of init(). There's nothing to stop you. But you shouldn't. The original reason for init() was that ancient versions of Java couldn't dynamically invoke constructors with arguments, so there was no way to give the constructur a ServletConfig. That no longer applies, but servlet containers still will only call your no-arg constructor. So you won't have access to a ServletConfig or ServletContext.

Question How can a servlet refresh automatically if some new data has entered the database? (Servlets)
Answer You can use a client-side Refresh or Server Push.
Question The code in a finally clause will never fail to execute, right? (Servlets) Answer Using System.exit(1); in try block will not allow finally code to execute.
Question Why are there no global variables in Java? (Core Java) Answer Global variables are considered bad form for a variety of reasons:

Adding state variables breaks referential transparency (you no longer can understand a statement or expression on its own: you need to understand it in the understand a statement or expression on its own: you need to understand it in the context of the settings of the global variables). State variables lessen the cohesion of a program: you need to know more to understand how something works. A major point of ObjectOriented programming is to break up global state into more easily understood collections of local state. When you add one variable, you limit the use of your program to one instance. What you thought was global, someone else might think of as local: they may want to run two copies of your program at once. For these reasons, Java decided to ban global variables.

Question What does it mean that a class or member is final? (Core Java)

Answer A final class can no longer be subclassed. Mostly this is done for security reasons with basic classes like String and Integer. It also allows the compiler to make some optimizations, and makes thread safety a little easier to achieve. Methods may be declared final as well. This means they may not be overridden in a subclass. Fields can be declared final, too. However, this has a completely different meaning. A final field cannot be changed after it's initialized, and it must include an initializer statement where it's declared. For example,

public final double c = 2.998;
It's also possible to make a static field final to get the effect of C++'s const statement or some uses of C's #define, *e.g.* public static final double c = 2.998;
Question What does it mean that a method or class is abstract? (Core Java)

Answer An abstract class cannot be instantiated. Only its subclasses can be instantiated. You indicate that a class is abstract with the abstract keyword like this: public abstract class Container extends Component { Abstract classes may contain abstract methods. A method declared abstract is not actually implemented in the current class. It exists only to be overridden in subclasses. It has no body. For example,

public abstract float price();
Abstract methods may only be included in abstract classes. However, an abstract class is not required to have any abstract methods, though most of them do.Each subclass of an abstract class must override the abstract methods of its superclasses or itself be declared abstract.

Question what is a transient variable? (Core Java) Answer transient variable

is a variable that may not be serialized.
Question How are Observer and Observable used? (Core Java)

Answer Objects that subclass the Observable class maintain a list of observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.

Question Can a lock be acquired on a class? (Core Java)
Answer Yes, a lock can be acquired on a class. This lock is acquired on the class's Class object.
Question What state does a thread enter when it terminates its processing? (Core Java) Answer When a thread terminates its processing, it enters the dead state.
Question How does Java handle integer overflows and underflows? (Core Java) Answer It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.
Question What is the difference between the >> and >>> operators? (Core Java)
Answer The >> operator carries the sign bit when shifting right. The >>> zero-fills bits that have been shifted out.
Question Is sizeof a keyword? (Core Java) Answer The sizeof operator is not a keyword.
Question Does garbage collection guarantee that a program will not run out of memory? (Core Java)

Answer Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection.

Question Can an object's finalize() method be invoked while it is reachable? (Core Java)

Answer An object's finalize() method cannot be invoked by the garbage collector while the object is still reachable. However, an object's finalize() method may be invoked by other objects.

Question What value does readLine() return when it has reached the end of

a file? (Core Java)

Answer The readLine() method returns null when it has reached the end of a file.
Question Can a for statement loop indefinitely? (Core Java) Answer Yes, a for statement can loop indefinitely. For example, consider the following: for(;;) ;

Question To what value is a variable of the String type automatically initialized? (Core Java) Answer The default value of an String type is null.
Question What is a task's priority and how is it used in scheduling? (Core Java)

Answer A task's priority is an integer value that identifies the relative order in which it should be executed with respect to other tasks. The scheduler attempts to schedule higher priority tasks before lower priority tasks.

Question What is the range of the short type? (Core Java) Answer The range of the short type is -(2^15) to 2^15 - 1.
Question What is the purpose of garbage collection? (Core Java) Answer The purpose of garbage collection is to identify and discard objects that are no longer needed by a program so that their resources may be reclaimed and reused.
Question How may messaging models do JMS provide for and what are they? (JMS) Answer JMS provide for two messaging models, publish-andsubscribe and point-to-point queuing.
Question What information is needed to create a TCP Socket? (Networking) Answer The Local System?s IP Address and Port Number. And the Remote System's IPAddress and Port Number.
Question What Class.forName will do while loading drivers? (JDBC) Answer It is used to create an instance of a driver and register it with the DriverManager. When you have loaded a driver, it is available for making a connection with a DBMS.
Question How to Retrieve Warnings? (JDBC)

Answer SQLWarning objects are a subclass of SQLException that deal with database access warnings. Warnings do not stop the execution of an application, as exceptions do; they simply alert the user that something did not happen as planned. A warning can be reported on a Connection object, a Statement object (including PreparedStatement and CallableStatement objects), or a ResultSet object. Each of these classes has a getWarnings method, which you must invoke in order to see the first warning reported on the calling object

E.g. **SQLWarning warning = stmt.getWarnings(); if (warning != null) { while (warning != null) { System.out.println("Message: " + warning.getMessage()); System.out.println("SQLState: " + warning.getSQLState()); System.out.print("Vendor error code: "); System.out.println(warning.getErrorCode()); warning = warning.getNextWarning(); } } Question How many JSP scripting elements are there and what are they? (JSP) Answer There are three scripting language elements: declarations scriptlets expressions**

Question In the Servlet 2.4 specification SingleThreadModel has been deprecates, why? (JSP)

Answer Because it is not practical to have such model. Whether you set isThreadSafe to true or false, you should take care of concurrent client requests to the JSP page by synchronizing access to any shared objects defined at the page level.

Question what are stored procedures? How is it useful? (JDBC)

Answer A stored procedure is a set of statements/commands which reside in the database. The stored procedure is precompiled and saves the database the effort of parsing and compiling sql statements everytime a query is run. Each Database has it's own stored procedure language, usually a variant of C with a SQL preproceesor. Newer versions of db's support writing stored procs in Java and Perl too.

Before the advent of 3-tier/n-tier architecture it was pretty common for stored procs to implement the business logic(A lot of systems still do it). The biggest advantage is of course speed. Also certain kind of data manipulations are not achieved in SQL. Stored procs provide a mechanism to do these manipulations. Stored procs are also useful when you want to do Batch updates/exports/houseKeeping kind of stuff on the db. The overhead of a JDBC Connection may be significant in these cases.

Question What do you understand by private, protected and public? (Core Java)

Answer These are accessibility modifiers. Private is the most restrictive, while public is the least restrictive. There is no real difference between protected and the default type (also known as package protected) within the context of the same package, however the protected keyword allows visibility to a derived class in a different package.

Question What is Downcasting ? (Core Java) Answer Downcasting is the casting from a general to a more specific type, *i.e.* casting down the hierarchy
Question Can a method be overloaded based on different return type but same argument type? (Core Java)
Answer No, because the methods can be called without using their return type in which case there is ambiquity for the compiler.
Question What happens to a static var that is defined within a method of a class ? (Core Java) Answer Can't do it. You'll get a compilation error.
Question How many static init can you have? (Core Java)

Answer As many as you want, but the static initializers and class variable initializers are executed in textual order and may not refer to class variables declared in the class whose declarations appear textually after the use, even though these class variables are in scope.

Question What is the difference amongst JVM Spec, JVM Implementation, JVM Runtime ? (Core Java)

Answer The JVM spec is the blueprint for the JVM generated and owned by Sun. The JVM implementation is the actual implementation of the spec by a vendor and the JVM runtime is the actual running instance of a JVM implementation

Question Describe what happens when an object is created in Java? (Core Java) Answer Several things happen in a particular order to ensure the object is constructed properly:

1. Memory is allocated from heap to hold all instance variables and implementationspecific data of the object and its superclasses. Implemenation-specific data includes pointers to class and method data.

2. The instance variables of the objects are initialized to their default values.

3. The constructor for the most derived class is invoked. The first thing a constructor does is call the consctructor for its superclasses. This process continues until the constrcutor for java.lang.Object is called, as java.lang.Object is the base class for all objects in java.

3. Before the body of the constructor is executed, all instance variable initializers and initialization blocks are executed. Then the body of the constructor is executed. Thus, the constructor for the base class completes first and constructor for the most derived class completes last.

Question What does the "final" keyword mean in front of a variable? A method? A class? (Core Java)

Answer FINAL for a variable : value is constant. FINAL for a method : cannot be overridden. FINAL for a class : cannot be derived .

Question What is the difference between instanceof and isInstance? (Core Java)

Answer instanceof is used to check to see if an object can be cast into a specified type without throwing a cast class exception. isInstance() Determines if the specified Object is assignmentcompatible with the object represented by this Class. This method is the dynamic equivalent of the Java language instanceof operator. The method returns true if the specified Object argument is non-null and can be cast to the reference type represented by this Class object without raising a ClassCastException. It returns false otherwise.

Question Why does it take so much time to access an Applet having Swing Components the first time? (Swing)

Answer Because behind every swing component are many Java objects and resources. This takes time to create them in memory. JDK 1.3 from Sun has some improvements which may lead to faster execution of Swing applications.

Question How do I include static files within a JSP page? (JSP)

Answer Static resources should always be included using the JSP include directive. This way, the inclusion is performed just once during the translation phase. The following example shows the syntax: Do note that you should always supply a relative URL for the file attribute. Although you can also include static resources using the action, this is not advisable as the inclusion is then performed for each and every request.

Question Why does JComponent have add() and remove() methods but Component does not? (Swing)

Answer because JComponent is a subclass of Container, and can contain other components and jcomponents.
Question How would you create a button with rounded edges? (Swing)

Answer There are 2 ways. The first thing is to know that a JButton?s edges are drawn by a Border. so you can override the Button?s paintComponent(Graphics) method and draw a circle or rounded rectangle (whatever), and turn off the border. Or you can create a custom border that draws a circle or rounded rectangle around any component and set the button?s border to it.

Question How would you detect a keypress in a JComboBox? (Swing)
Answer Add a KeyListener to the JComboBox?s editor component instead of adding a KeyListener to the JComboBox itself.
Question Why should the implementation of any Swing callback (like a listener) execute quickly? (Swing)
Answer Because callbacks are invoked by the event dispatch thread which will be blocked processing other events for as long as your method takes to execute.
Question Why would you use SwingUtilities.invokeAndWait or SwingUtilities.invokeLater? (Swing)

Answer I want to update a Swing component but I?m not in a callback. If I want the update to happen immediately (perhaps for a progress bar component) then I?d use invokeAndWait. If I don?t care when the update occurs, I?d use invokeLater.

Question If your UI seems to freeze periodically, what might be a likely reason? (Swing)

Answer A callback implementation like ActionListener.actionPerformed or MouseListener.mouseClicked is taking a long time to execute thereby blocking the event dispatch thread from processing other UI events.

Question Which Swing methods are thread-safe? (Swing) Answer The only thread-safe methods are repaint(), revalidate(), and invalidate()
Question Why won?t the JVM terminate when I close all the application windows? (Swing) Answer The AWT event dispatcher thread is not a daemon thread. You must explicitly call System.exit to terminate the JVM.
Question JFrame is a heavy weight component. Since it extends an awt Frame, is it Thread Safe? (Swing)

Answer JFrame itself is, since it is just a java.awt.Frame in essence, but the root pane/content pane is not, so it effectively follows the same rules for Swing containers and is not considered thread safe.

Question What is the difference between invokeAndWait() and invokeLater()? (Swing) Answer invokeAndWait() blocks until the Runnable task is complete; it's synchronous.

invokeLater() posts an action event to the event queue and returns immediately; it's asynchronous.
Question Why is not recommended to have instance variables in Interface (Core Java)

Answer By Default, All data members and methods in an Interface are public. Having public variables in a class that will be implementing it will be violation of the Encapsulation principal. I hope that's pretty ok..

Question What is the diffrence between inner class and nested class? (Core Java) Answer When a class is defined within a scope od another class, then it becomes inner class. If the access modifier of the inner class is static, then it becomes nested class.
Question What is a compilation unit? (Core Java) Answer A compilation unit is a Java source code file.
Question What restrictions are placed on method overriding? (Core Java)

Answer Overridden methods must have the same name, argument list, and return type. The overriding method may not limit the access of the method it overrides. The overriding method may not throw any exceptions that may not be thrown by the overridden method.

Question How can a dead thread be restarted? (Core Java) Answer A dead thread cannot be restarted.
Question What happens if an exception is not caught? (Core Java)

Answer An uncaught exception results in the uncaughtException() method of the thread's ThreadGroup being invoked, which eventually results in the termination of the program in which it is thrown.

Question Which arithmetic operations can result in the throwing of an ArithmeticException? (Core Java)

Answer Integer / and % can result in the throwing of an ArithmeticException
Question Can an abstract class be final? (Core Java) Answer An abstract class may not be declared as final
Question What happens if a trycatch-finally statement does not have a catch clause to handle an exception that is thrown within the body of the try statement? (Core Java) Answer The exception propagates up to the next higher level trycatch statement (if any) or results in the program's termination
Question What is numeric promotion? (Core Java)

Answer Numeric promotion is the conversion of a smaller numeric type to a larger numeric type, so that integer and floatingpoint operations may take place. In numerical promotion, byte, char, and short values are converted to int values. The int values are also converted to long values, if necessary. The long and float values are converted to double values, as required.

Question What is the difference between a public and a non-public class? (Core Java)
Answer A public class may be accessed outside of its package. A non-public class may not be accessed outside of its package.
Question To what value is a variable of the boolean type automatically initialized? (Core Java)
Answer The default value of the boolean type is false.
Question Can try statements be nested? (Core Java) Answer Yes
Question What is the difference between the prefix and postfix forms of the ++ operator? (Core Java)

Answer The prefix form performs the increment operation and returns the value of the increment operation. The postfix form returns the current value all of the expression and then performs the increment operation on that value.

Question What is the purpose of a statement block? (Core Java) Answer A statement block is used to organize a sequence of statements as a single statement group.
Question What is a Java package and how is it used? (Core Java)

Answer A Java package is a naming context for classes and interfaces. A package is used to create a separate name space forgroups of classes and interfaces. Packages are also used to organize related classes and interfaces into a single API unit and to control accessibility to these classes and interfaces.

Question What modifiers may be used with a top-level class? (Core Java)
Answer A top-level class may be public, abstract, or final.
Question What are the Object and Class classes used for? (Core Java)
Answer The Object class is the highest-level class in the Java class hierarchy. The Class class is used to represent the classes and interfaces that are loaded by a Java program.
Question How does a try statement determine which catch clause should be used to handle an exception? (Core Java)

Answer When an exception is thrown within the body of a try statement, the catch clauses of the try statement are examined in the order in which they appear. The first catch clause that is capable of handling the exception is executed. The remaining catch clauses are ignored.

Question What are synchronized methods and synchronized statements? (Core Java)

Answer Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

Question What is the difference between an if statement and a switch statement? (Core Java)

Answer The if statement is used to select among two alternatives. It uses a boolean expression to decide which alternative should be executed. The switch statement is used to select among multiple alternatives. It uses an int expression to determine which alternative should be executed.

Question which containers use a border Layout as their default layout? (AWT) Answer The window, Frame and Dialog classes use a border layout as their default layout.
Question What is the preferred size of a component? (AWT) Answer The preferred size of a component is the minimum component size that will allow the component to display normally.
Question Which containers use a FlowLayout as their default layout?

(AWT)
Answer The Panel and Applet classes use the FlowLayout as their default layout.
Question What is the immediate superclass of the Applet class? (AWT)
Answer Panel.
Question Name three Component subclasses that support painting (AWT)
Answer The Canvas, Frame, Panel, and Applet classes support painting.
Question what is the immediate superclass of the Dialog class? (AWT)
Answer Window.
Question what is clipping? (AWT) Answer Clipping is the process of confining paint operations to a limited area or shape.
Question What is the difference between a MenuItem and a CheckboxMenuItem? (AWT) Answer The CheckboxMenuItem class extends the MenuItem class to support a menu item that may be checked or unchecked.
Question What class is the top of the AWT event hierarchy? (AWT) Answer The java.awt.AWTEvent class is the highest-level class in the AWT event-class hierarchy.
Question In which package are most of the AWT events that support the eventdelegation model defined? (AWT)
Answer Most of the AWT-related events of the eventdelegation model are defined in the java.awt.event package. The AWTEvent class is defined in the java.awt package.
Question Which class is the immediate superclass of the MenuComponent class (AWT) Answer Object.
Question Which containers may have a MenuBar? (AWT) Answer Frame.
Question What is the relationship between the Canvas class and the Graphics class? (AWT) Answer A Canvas object provides access to a Graphics object via its paint() method.
Question How are the elements of a BorderLayout organized? (AWT)
Answer The elements of a BorderLayout are organized at the borders (North, South, East, and West) and the center of a container.
Question What is the difference between a Window and a Frame? (AWT)
Answer The Frame class extends Window to define a main application window that can have a menu bar.
Question What is the difference between the Font and FontMetrics classes? (AWT) Answer The FontMetrics class is used to define implementationspecific properties, such as ascent and descent, of a Font object.
Question How are the elements of a CardLayout organized? (AWT) Answer The elements of a CardLayout are stacked, one on top of the other, like a deck of cards.

Question What is the relationship between clipping and repainting? (AWT)
Answer When a window is repainted by the AWT painting thread, it sets the clipping regions to the area of the window that requires repainting.
Question What is the relationship between an event-listener interface and an event-adapter class? (AWT)

Answer An event-listener interface defines the methods that must be implemented by an event handler for a particular kind of event. An event adapter provides a default implementation of an event-listener interface.

Question How can a GUI component handle its own events? (AWT) Answer A component can handle its own events by implementing the required event-listener interface and adding itself as its own event listener.
Question How are the elements of a GridBagLayout organized? (AWT)

Answer The elements of a GridBagLayout are organized according to a grid. However, the elements are of different sizes and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.

Question What advantage do Java's layout managers provide over traditional windowing systems? (AWT)

Answer Java uses layout managers to lay out components in a consistent manner across all windowing platforms. Since Java's layout managers aren't tied to absolute sizing and positioning, they are able to accommodate platform-specific differences among windowing systems.

Question What is the difference between the paint() and repaint() methods? (AWT) Answer The paint() method supports painting via a Graphics object. The repaint() method is used to cause paint() to be invoked by the AWT painting thread.
Question How can the Checkbox class be used to create a radio button? (AWT) Answer By associating Checkbox objects with a CheckboxGroup
Question What is the difference between a Choice and a List? (AWT)

Answer A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices. Only one item may be selected from a Choice. A List may be displayed in such a way that several List items are visible. A List supports the selection of one or more List items.

Question What interface is extended by AWT event listeners? (AWT) Answer All AWT event listeners extend the java.util.EventListener interface.
Question What is a layout manager? (AWT) Answer A layout manager is an object that is used to organize components in a container.
Question Which Component subclass is used for drawing and painting? (AWT) Answer Canvas.
Question What are the problems faced by Java programmers who dont use layout managers? (AWT)

Answer Without layout managers, Java programmers are faced with determining how their GUI will be displayed across multiple windowing systems and finding a common sizing and positioning that will work within the constraints imposed by each windowing system.

Question What is the difference between a Scrollbar and a ScrollPane? (Swing) Answer A Scrollbar is a Component, but not a Container. A ScrollPane is a Container. A ScrollPane handles its own events and performs its own scrolling.
Question What is the Collections API? (Java util) Answer The Collections API is a set of classes and interfaces that support operations on collections of objects.
Question What is the List interface? (Java util) Answer The List interface provides support for ordered collections of objects.
Question What is the Vector class? (Java util) Answer The Vector class provides the capability to implement a growable array of objects.
Question What is an Iterator interface? (Java util) Answer The Iterator interface is used to step through the elements of a Collection.
Question Which java.util classes and interfaces support event handling? (Java util) Answer The EventObject class and the EventListener interface support event processing.
Question What is the GregorianCalendar class? (Java util) Answer The GregorianCalendar provides support for traditional Western calendars.
Question What is the Locale class? (Java util) Answer The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region.
Question What is the SimpleTimeZone class? (Java util) Answer The SimpleTimeZone class provides support for a Gregorian calendar.
Question What is the Map interface? (Java util)
Answer The Map interface replaces the JDK 1.1 Dictionary class and is used

associate keys with values.
Question What is the highest-level event class of the eventdelegation model? (Java util) Answer The java.util.EventObject class is the highest-level class in the eventdelegation class hierarchy.
Question What is the Collection interface? (Java util)
Answer The Collection interface provides support for the implementation of a mathematical bag - an unordered collection of objects that may contain duplicates.
Question What is the Set interface? (Java util) Answer The Set interface provides methods for accessing the elements of a finite mathematical set. Sets do not allow duplicate elements.
Question What is the purpose of the enableEvents() method? (Java util)

Answer The enableEvents() method is used to enable an event for a particular object. Normally, an event is enabled when a listener is added to an object for a particular event. The enableEvents() method is used by objects that handle events by overriding their eventdispatch methods.

Question What is the ResourceBundle class? (Java util) Answer The ResourceBundle class is used to store localespecific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is being run.
Question What is the difference between yielding and sleeping? (Threads)
Answer When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.
Question When a thread blocks on I/O, what state does it enter? (Threads)
Answer A thread enters the waiting state when it blocks on I/O.
Question When a thread is created and started, what is its initial state? (Threads) Answer A thread is in the ready state after it has been created and started.
Question What invokes a thread's run() method? (Threads) Answer After a thread is started, via its start() method or that of the Thread class, the JVM invokes the thread's run() method when the thread is initially executed.
Question What method is invoked to cause an object to begin executing as a separate thread? (Threads)
Answer The start() method of the Thread class is invoked to cause an object to begin executing as a separate thread.
Question What is the purpose of the wait(), notify(), and notifyAll() methods? (Threads)

Answer The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to wait for a shared resource. When a thread executes an object's wait() method, it enters the waiting state. It only enters the ready state after another thread invokes the object's notify() or notifyAll() methods.

Question What are the high-level thread states? (Threads) Answer The high-level thread states are ready, running, waiting, and dead.
Question What happens when a thread cannot acquire a lock on an object? (Threads) Answer If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object's lock, it enters the waiting state until the lock becomes available.
Question How does multithreading take place on a computer with a single CPU? (Threads)

Answer The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.

Question What happens when you invoke a thread's interrupt method while it is sleeping or waiting? (Threads)
Answer When a task's interrupt() method is executed, the task enters the ready state. The next time the task enters the running state, an InterruptedException is thrown.
Question What state is a thread in when it is executing? (Threads) Answer An executing thread is in the running state.
Question What are three ways in which a thread can enter the waiting state? (Threads)

Answer A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully attempting to acquire an object's lock, or by invoking an object's wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.

Question What method must be implemented by all threads? (Threads)
Answer All tasks must implement the run() method, whether they are a subclass of Thread or implement the Runnable interface.
Question What are the two basic ways in which classes that can be run as threads may be defined? (Threads)
Answer A thread class may be declared as a subclass of Thread, or it may implement the Runnable interface.

Question How can you store international / Unicode characters into a cookie? (JSP) Answer One way is, before storing the cookie URLEncode it. URLEnocder.encoder(str); And use URLDecoder.decode(str) when you get the stored cookie.

Question What is the difference between URL instance and URLConnection instance? (Networking)

Answer A URL instance represents the location of a resource, and a URLConnection instance represents a link for accessing or communicating with the resource at the location.

Question What are the two important TCP Socket classes? (Networking)

Answer Socket and ServerSocket. ServerSocket is used for normal two-way socket communication. Socket class allows us to read and write through the sockets. getInputStream() and getOutputStream() are the two methods available in Socket class.

Question How to call a Stored Procedure from JDBC? (JDBC)

Answer The first step is to create a CallableStatement object. As with Statement an and PreparedStatement objects, this is done with an open Connection object. A CallableStatement object contains a call to a stored procedure.

E.g.

```
CallableStatement cs = con.prepareCall("{call
SHOW_SUPPLIERS}");
ResultSet rs = cs.executeQuery();
```

Question What are the implicit objects? (JSP) Answer Implicit objects are objects that are created by the web container and contain information related to a particular request, page, or application. They are:

request response pageContext session application out config page exception

Question Is JSP technology extensible? (JSP) Answer YES. JSP technology is extensible through the development of custom actions, or tags, which are encapsulated in tag libraries

Question What gives java it's "write once and run anywhere" nature? (Core Java)

Answer Java is compiled to be a byte code which is the intermediate language between source code and machine code. This byte code is not platorm specific and hence can be fed to any platform. After being fed to the JVM, which is specific to a particular operating system, the code platform specific machine code is generated thus making java platform independent.

Question What are the four corner stones of OOP ? (Core Java) Answer Abstraction, Encapsulation, Polymorphism and Inheritance.
Question Difference between a Class and an Object ? (Core Java) Answer A class is a definition or prototype whereas an object is an instance or living representation of the prototype.
Question What is the difference between method overriding and overloading? (Core Java) Answer Overriding is a method with the same name and arguments as in a parent, whereas overloading is the same method name but different arguments.
Question What is a "stateless" protocol ? (Core Java)

Answer Without getting into lengthy debates, it is generally accepted that protocols like HTTP are stateless *i.e.* there is no retention of state between a transaction which is a single request response combination.

Question What is constructor chaining and how is it achieved in Java ? (Core Java)

Answer A child object constructor always first needs to construct its parent (which in turn calls its parent constructor.). In Java it is done via an implicit call to the no-args constructor as the first statement.

Question What is passed by ref and what by value ? (Core Java) Answer All Java method arguments are passed by value. However, Java does manipulate objects by reference, and all object variables themselves are references.
Question Can RMI and Corba based applications interact ? (RMI) Answer Yes they can. RMI is available with IIOP as the transport protocol instead of JRMP.
Question You can create a String object as String str = "abc"; Why cant a button object be created as Button bt = "abc";? Explain (Core Java)

Answer The main reason you cannot create a button by Button bt1= "abc"; is because "abc" is a literal string (something slightly different than a String object, by-the-way) and bt1 is a Button object. The only object in Java that can be assigned a literal String is java.lang.String. Important to note that you are NOT calling a java.lang.String constuctor when you type String s = "abc";

Question What does the "abstract" keyword mean in front of a method? A class? (Core Java)

Answer Abstract keyword declares either a method or a class. If a method has a abstract keyword in front of it,it is called abstract method.Abstract method hs no body.It has only arguments and return type.Abstract methods act as placeholder methods that are implemented in the subclasses. Abstract classes can't be instantiated.If a class is declared as abstract,no objects of that class can be created.If a class contains any abstract method it must be declared as abstract.

Question How many methods do u implement if implement the Serializable Interface? (Core Java)
Answer The Serializable interface is just a "marker" interface, with no methods of its own to implement. Other 'marker' interfaces are java.rmi.Remote java.util.EventListener
Question What are the practical benefits, if any, of importing a specific class rather than an entire package (e.g. import java.net.* versus import java.net.Socket)? (Core Java)

Answer It makes no difference in the generated class files since only the classes that are actually used are referenced by the generated class file. There is another practical benefit to importing single classes, and this arises when two (or more) packages have classes with the same name. Take java.util.Timer and javax.swing.Timer, for example. If I import java.util.* and javax.swing.* and then try to use "Timer", I get an error while compiling (the class name is ambiguous between both packages). Let's say what you really wanted was the javax.swing.Timer class, and the only classes you plan on using in java.util are Collection and HashMap. In this case, some people will prefer to import java.util.Collection and import java.util.HashMap instead of importing java.util.*. This will now allow them to use Timer, Collection, HashMap, and other javax.swing classes without using fully qualified class names in.

Question What is the difference between logical data independence and physical data independence? (Core Java)

Answer Logical Data Independence - meaning immunity of external schemas to changeds in conceptual schema. Physical Data Independence - meaning immunity of conceptual schema to changes in the internal schema.

[illegible]

Question What is user defined exception? (Core Java) Answer Apart from the exceptions already defined in Java package libraries, user can define his own exception classes by extending Exception class.

Interview Part 3

Book Part-2 SCJP/SCJD– BONUS
Java Coding Standards.
Clarity and Maintainability.
Core Java Database Issues.

The Java Programmers & Developer exams are challenging. There are a lot of complex design issues to consider, and a host of advanced Java technologies to understand and implement correctly. The exam assessors work under very strict guidelines. You can create the most brilliant application ever to grace a JVM, but if you don't cross your t's and dot your i's the assessors have no choice but to deduct crucial (and sometimes substantial) points from your project. This chapter will help you cross your t's and dot your i's. Following coding standards is not hard; it just requires diligence. If you are careful it's no-brainer stuff, and it would be a shame to lose points because of a curly brace in the wrong place. The Developer exam stresses things that must be done to avoid automatic failure. The exam uses the word must frequently. When we use the word must, we use it in the spirit of the exam, if you must you must, so just get on with it. Let's dive into the fascinating world of Java Coding Standards.
Coding Standards.

Java.lang — The Math Class, Strings, and Wrappers.
Use Sun Java Coding Standards:

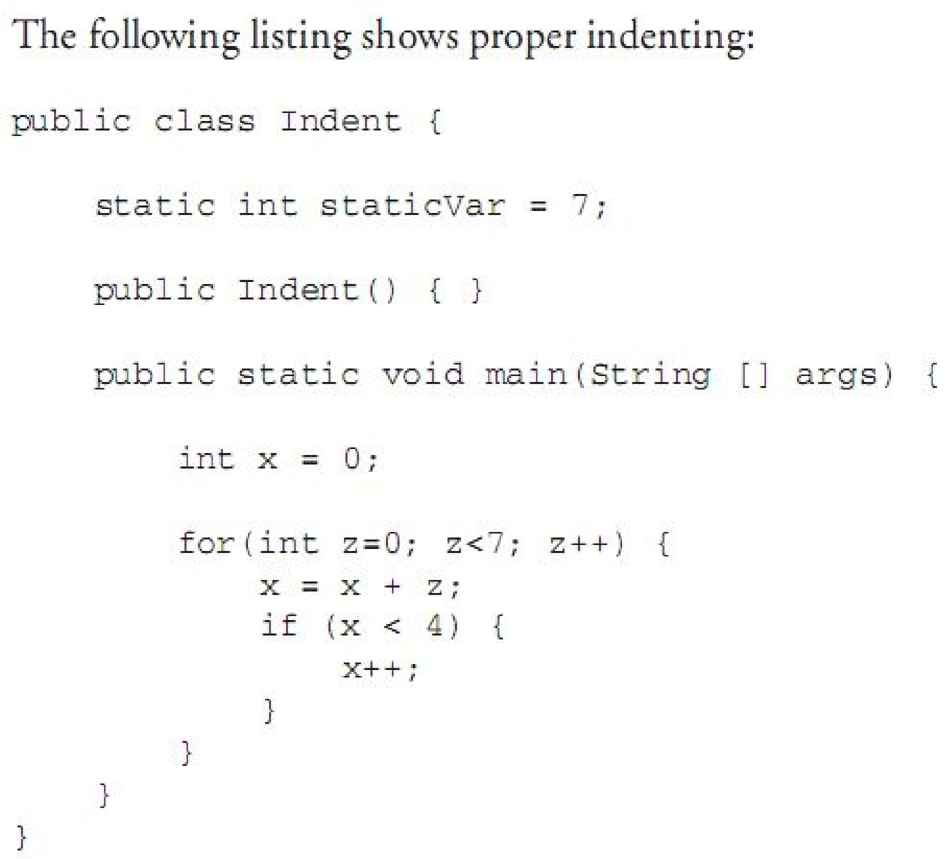
Don't forget the Developer exam is challenging. There are a lot of complex design issues to consider, and a host of advanced Java technologies to understand and implement correctly. The exam assessors work under very strict guidelines. You can create the most brilliant application ever to grace a JVM, but if you don't cross your t's and dot your i's the assessors have no choice but to deduct crucial (and sometimes substantial) points from your project. This chapter will help you cross your t's and dot your i's. Following coding standards is not hard; it just requires diligence. If you are careful it's no-brainer stuff, and it would be a shame to lose points because of a curly brace in the wrong place. The Developer exam stresses things that must be done to avoid automatic failure. The exam uses the word must

frequently. When we use the word must, we use it in the spirit of the exam, if you must you must, so just get on with it. Let's dive into the fascinating world of Java Coding Standards.

Spacing Standards

This bonus section covers the standards for indenting, line-length limits, line breaking, and white space. Indenting We said this was going to be fascinating didn't we? Each level of indentation must be four spaces, exactly four spaces, always four spaces. Tabs must be set to eight spaces. If you are in several levels of indentation you can use a combination of tabs and (sets of four) spaces to accomplish the correct indentation. So if you are in a method and you need to indent 12 spaces, you can either press SPACEBAR 12 times, or press TAB once and then press SPACEBAR four times. (Slow down coach.) We recommend not using the TAB key, and sticking to the SPACEBAR—it's just a bit safer.

***When to Indent if you indent like this, you'll make your assessor proud*: ■ Beginning comments, package declarations, import statements, interface declarations, and class declarations should not be indented.**
■ Static variables, instance variables, constructors, methods, and their respective comments* should be indented one level.
■ within constructors and methods, local variables, statements, and their comments should be indented another level.
■ Statements (and their comments) within block statements should be indentedanother level for each level of nesting involved. (Don't worry; we'll give you an example.)

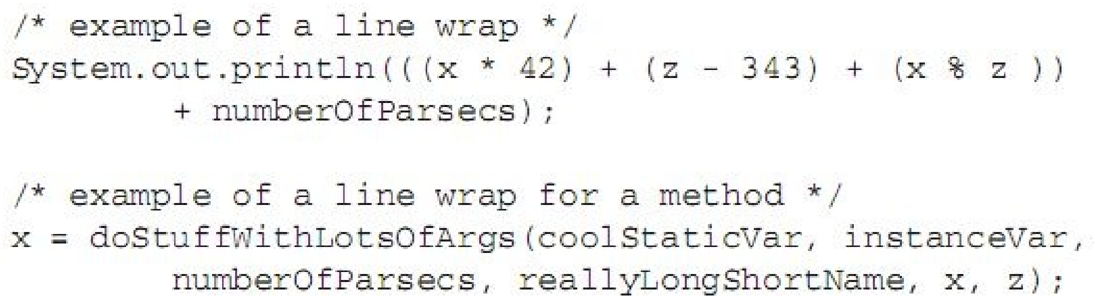

The following listing shows proper indenting:

```
public class Indent {

    static int staticVar = 7;

    public Indent() { }

    public static void main(String [] args) {

        int x = 0;

        for(int z=0; z<7; z++) {
            x = x + z;
            if (x < 4) {
                x++;
            }
        }
    }
}
```

Line Lengths and Line Wrapping

The general rule is that a line shouldn't be longer than 80 characters. We recommend 65 characters just to make sure that a wide variety of editors will handle your code gracefully. When a line of code is longer than will fit on a line there are some lines wrapping guidelines to follow. We can't say for sure that these are a must, but if you follow these guidelines you can be sure that you're on safe ground:

■ Break after a comma.
■ Break before an operator.
■ Align the new line a tab (or eight spaces) beyond the beginning of the line being broken.
■ Try not to break inside an inner parenthesized expression. (Hang on, the example is coming.)
The following snippet demonstrates acceptable line wrapping:



```
/* example of a line wrap */
System.out.println(((x * 42) + (z - 343) + (x % z ))
        + numberOfParsecs);

/* example of a line wrap for a method */
x = doStuffWithLotsOfArgs(coolStaticVar, instanceVar,
        numberOfParsecs, reallyLongShortName, x, z);
```

White Space

Can you believe we have to go to this level of detail? It turns out that if you don’t parcel out your blank spaces as the standards say you should, you can lose points. With that happy thought in mind, let’s discuss the proper use of blank lines and blank statements.

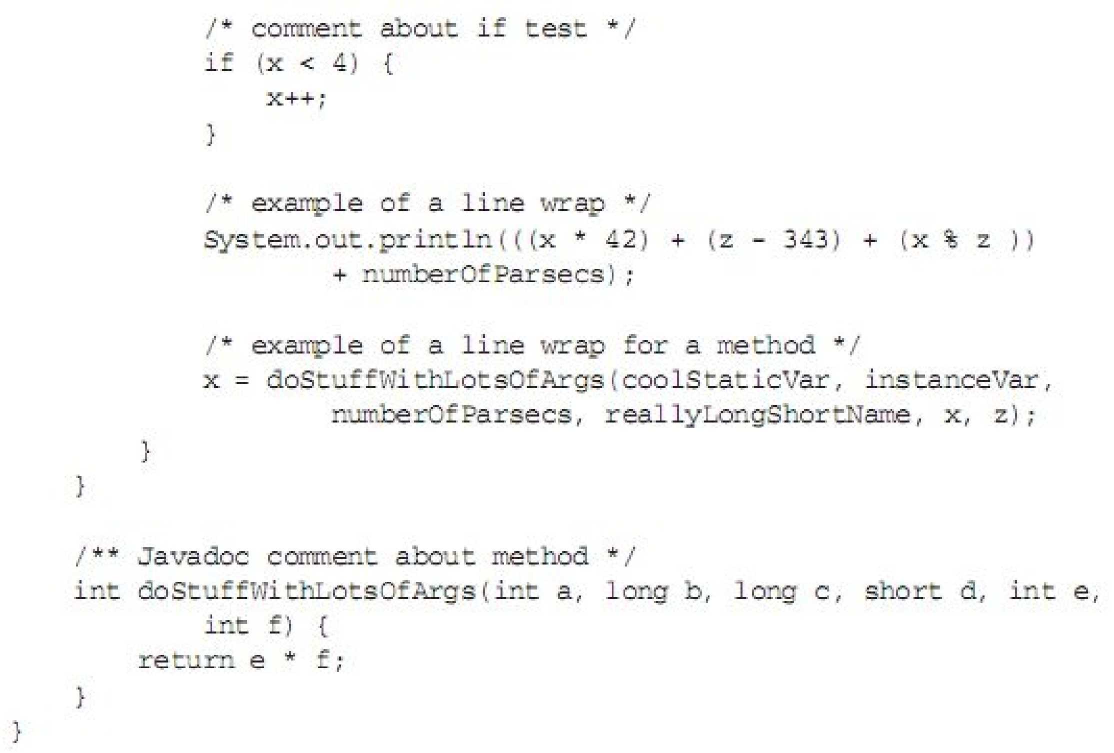
The Proper Use of Blank Lines

Blank lines are used to help readers of your code (which might be you, months after you wrote it) to easily spot the logical blocks within your source file. If you follow these recommendations in your source files, your blank line worries will be over. Use a blank line,

■ Between methods and constructors.
■ After your last instance variable.
■ Inside a method between the local variables and the first statement.
■ Inside a method to separate logical segments of code.
■ Before single line or block comments.
Use two blank lines between the major sections of the source file: the package, the import statement(s), the class, and the interface.
The Proper Use of Blank Spaces **Blank spaces are used to make statements more readable, and less squished together. Use a blank space,**
■ Between binary operators
■ After commas in an argument list
■ After the expressions in a for statement
■ Between a keyword and a parenthesis

■ After casts The following code sample demonstrates proper form to use when indenting, skipping lines, wrapping lines, and using spaces. We haven’t covered all of the rules associated with the proper use of comments; therefore, this sample does not demonstrate standard comments:

```
/*
 * This listing demonstrates only proper spacing standards
 *
 * The Javadoc comments will be discussed in a later chapter
 */


package com.wickedlysmart.utilities;


import java.util.*;


/**
 * CoolClass description
 *
 * @version .97 10 Oct 2002
 * @author  Joe Beets
 */
public class CoolClass {

    /** Javadoc static var comment */
    public static int coolStaticVar;

    /** Javadoc public i-var comment */

      public long instanceVar;

      /* private i-var comment */
      private short reallyLongShortName;

      /** Javadoc constructor comment */
      public CoolClass() {
        // do stuff
      }

      /** Javadoc comment about method */
      void coolMethod() {
          int x = 0;
          long numberOfParsecs = 0;

          /* comment about for loop */
          for(z = 0; z < 7; z++) {
              x = x + z;
```



```
            /* comment about if test */
            if (x < 4) {
                x++;
            }

            /* example of a line wrap */
            System.out.println(((x * 42) + (z - 343) + (x % z ))
                    + numberOfParsecs);

            /* example of a line wrap for a method */
            x = doStuffWithLotsOfArgs(coolStaticVar, instanceVar,
                    numberOfParsecs, reallyLongShortName, x, z);
        }
    }

    /** Javadoc comment about method */
    int doStuffWithLotsOfArgs(int a, long b, long c, short d, int e,
            int f) {
        return e * f;
    }
}
```

Declarations Are Fun

Declarations are a huge part of Java. They are also complex, loaded with rules, and if used sloppily can lead to bugs and poor maintainability. The following set of guidelines is intended to make your code more readable, more debuggable, and more maintainable.

Sequencing Your Declarations

The elements in your Java source files should be arranged in a standard sequence. In some cases the compiler demands it, and for the rest of the cases consistency will help you win friends and influence people. Here goes:

- **class comments**
- **package declaration**
- **import statements**
- **class declaration**
- **static variables**
- **instance variables**
- **constructors**

Location and Initialization

The following guidelines should be considered when making Java declarations:

■ Within methods:
■ Declare and initialize local variables before other statements (whenever possible).
■ Declare and initialize block variables before other block statements (when possible).

■ Declare only one member per line.
■ Avoid shadowing variables. This occurs when an instance variable hasthe same name as a local or block variable. While the compiler will allowit, shadowing is considered very unfriendly towards the next co-worker (remember: potentially psychopathic) who has to maintain your code.

Capitalization Three guesses. You better use capitalization correctly when you declare and use your package, class, interface, method, variable, and constant names. The rules are pretty simple:
■ Package names The safest bet is to use lowercase when possible:
com.wickedlysmart.utilities
■ Class and Interface names Typically they should be nouns; capitalize the first letter and any other first letters in secondary words within the name: Customer or CustomTable
■ Method names Typically they should be verbs; the first word should be lowercase, and if there are secondary words, the first letter of each should be capitalized:
initialize(); or getTelescopicOrientation();

■ Variable names They should follow the same capitalization rules as methods; you should start them with a letter (even though you can use _ or $, don’t), and only temporary variables like looping variables should use single character names:

currentIndex; or name; or x;
■ Constant names To be labeled a constant, a variable must be declared static and final. Their names should be all uppercase and underscores must be used to separate words:
MAX_HEIGHT; or USED;
Clarity and Maintainability.

Clarity and Maintainability

Always Write Clear and Maintainable Codes Now that you’ve made your code readable, does your easy-to-read code actually make sense? Can it be easily maintained? These are huge issues for the exam, worth a very significant chunk of your assessment score. We'll look at everything from class design to error handling. Remember that you're a Team Player. Some key areas of code clarity are covered in more detail in the Documentation chapter, so we won't discuss them here. Those areas include the importance of meaningful comments and self-documenting identifiers. The issues raised in this chapter are

■ General programming style considerations
■ Following OO design principles
■ Reinventing the wheel

■ Error-handling

General Programming Considerations

The coding conventions covered in the previous chapter are a great starting point. But the exam is also looking for consistency and appropriateness in your programming style. The following section lists some key points you should keep in mind when writing your perfectly-formatted code. Some of these will be explained in subsequent sections; several of these points are related to OO design, for example, and we cover them in more detail in that section. Once again, this is no time to debate the actual merits of these principles. Again, imagine you've come into a project team and need to prove yourself as a, what? Yes! Team Player. The first thing the team is looking for is whether you can follow the conventions and standards so that everyone can work together without wanting to throw one another out the seventh floor window and onto the cement fountain below. (Unless you're a dot-com company and your office now looks over an abandoned gas station.) These points are in no particular order, so don't infer that the first ones are more important than the last. You can infer, however, that your exam assessor will probably be asking if you've done these things appropriately.

Keep Variable Scope as Small as Possible

Don't use an instance variable when a local variable will work! Not only does this impact memory use, but it reduces the risk that an object "slips out" to some place it shouldn't be used, either accidentally or on purpose. Wait to declare a variable until just before it's used. And you should always initialize a local variable at the time it is declared (which is just before use), with the exception of try/catch blocks. In that case, if the variable is declared and assigned in the try/catch blocks. In that case, if the variable is declared and assigned in the try/catch block, the compiler won't let you use it beyond that block, so if you need the variable after a try or catch block, then you'll have to declare it first outside the try/catch. Another way to reduce scope is to use a for loop rather than while. Remember from the Programmer's exam chapters that when you declare a variable as part of the for loop declaration (as opposed to merely initializing a variable declared prior to the loop), then the variable's scope ends with the loop. So you get scope granularity that's even smaller than a method.

Avoid Designing a Class That Has No Methods:

Objects are meant to have both state and behavior; they're not simply glorified structs. If you need a data structure, use a Collection. There are exceptions to this, however, that might apply to your exam assignment. Sometimes you do need an object whose sole purpose is to carry data from one location to another —usually as a result of a database request. A row in a table, for example, should be represented as an object in your Java program, and it might not always need methods if its sole job is to be, say, displayed in a GUI table. This is known as the ValueObject pattern. Which brings us to the next issue.

Use Design Patterns

When you use familiar patterns, then you've got a kind of shorthand for discussing your design with other programmers (even if that discussion is between your code/ comments and the other person. If you've done it right, you won't personally be there to talk about it, as is the case with the Developer exam). If you need a Singleton, make a Singleton—don't simply document that there is to be only one of these things. On the other hand, don't go forcing your design into a pattern just for the sake of using a pattern. Simplicity should be your first concern, but if it's a toss-up between your approach and an equally complex, well-known design pattern, go for the pattern.

Reduce the Visibility of Things As Much As Possible

In general, the more public stuff you expose to the world, the less free you are to make changes later without breaking someone else's code. The less you expose, the more flexibility you have for implementation changes later. And you know there are always changes. So, making variables, methods, and classes as restricted as you can while limiting what you expose to your "public interface," you'll be in good shape down the road. Obviously there are other subtle issues about inheritance (as in, what does a subclass get access to?), so there's more to consider here, but in general, be thinking about reducing your exposure (think of it as reducing your liability down the road). This is closely related to reducing the scope of variables.

Use Overloading Rather Than Logic

If you’ve got a method that needs to behave differently depending on the kind of thing it was actually handed, consider overloading it. Any time you see if or switch blocks testing the type of an argument, you should probably start thinking about overloading the method. And while you’re at it…

Avoid Long Argument Lists If you have a ton of arguments coming into a method, perhaps you need to encapsulate the stuff you need in that method into a class of its own type.
Don’t Invoke Potentially Overridable

Methods from a Constructor You already know that you can’t access any nonstatic things prior to your superconstructor running, but keep in mind that even after an object’s superconstructor has completed, the object is still in an incomplete state until after its constructor has finished. Polymorphism still works in a constructor. So if B extends A, and A calls a method in its constructor that B has overridden, well, guess what happens when somebody makes an instance of B. You got it. The B constructor invokes its superconstructor (A’s constructor). But inside the A constructor it invokes one of its own methods, but B has overridden that method. B’s method runs! In other words, an object can have one of its methods invoked even before its constructor has completed! So while B isn’t even a fully formed object, it can still be running code and even accessing its own instance variables. This is a problem because its instance variables have not yet been initialized to anything other than default values, even if they’re given explicit values when they’re declared. Yikes! So don’t do it. If it’s a final or private instance method, then you’re safe since you know it’ll never be overridden.

Code to Interfaces

Polymorphism, polymorphism, polymorphism. Use polymorphic arguments, return types, and variables whenever possible (in other words, declare a variable, return type, or argument as an interface type rather than a specific class type). Using an interface as the type lets you expose only the definition of what your code can do, and leaves the implementation flexible and extensible. And maintainable. And all the other good OO things-that-end-with-ble. But if you can’t…

Use Abstract Classes When You Need Functionality to Be Inherited If you really must have implementation code and/or instance variables, then use an abstract class and use that class as the declared polymorphic variable, argument, and return type.

Make Objects You're Finished

with Eligible for Garbage Collection You already know how to do this. Either explicitly set the reference variable to null when you have no more use of the object, or reassign a different object to that reference variable (thus abandoning the object originally referenced by it). At the same time…

Don't Make More Objects Than You Need To

Just because there's a garbage collector doesn't mean you won't have "memory issues." If you keep too many objects around on the heap, ineligible for garbage collection (but you won't, having read the preceding point), then you can still run out of memory. More likely, though, is just the problem that your performance might be slightly degraded by the overhead of both making all those objects and then having the garbage collector reclaim them. Don't do anything to alter your design just to shave a few objects, but pay attention in your implementation code. In some cases, you might be able to simply reuse an existing object by resetting its state.

Avoid Deeply Nested and Complex Logic

Less is more when it comes to branching. In fact, your assessor may be applying the Cyclomatic Complexity measure to your code, which considers code to be complex not based on lines of code, but rather on how many branch points there are. (It's actually much more complex than that. Ironically, the test for code complexity is itself a rather complex formula.) The bottom line is, whenever you see a nested if or anything other than very simple logic flow in a method, you should seriously consider redesigning that method or splitting functionality into separate methods.

Use Getters and Setters That Follow

the JavaBean Naming Convention That means you should use set<yourPropertyName> for methods that can modify a property (normally a property maps directly to an instance variable, but not necessarily) and get<yourPropertyName> for methods that can read a property. For example, a

String variable name would have the following getter/setter methods: setName(String name)

String getName()

If the property is a boolean, then you have a choice (yes, you actually have a choice) of whether to call the read method get<property> or is<property>. For example, a boolean instance variable motorOn can have the following getter/setter methods:

setMotorOn(boolean state) boolean getMotorOn() boolean isMotorOn()

The beauty of adhering to the JavaBeans naming convention is that, hey, you have to name it something and if you stick with the convention, then most Java-related tools (and some technologies) can read your code and automatically detect that you have editable properties, for example. It's cool; you should do it.

Don't Be a Procedural Programmer in an OO World The two dead giveaways that you haven't really made the transition to a complete object "being," are when you use the following:

■ Really Big Classes that have methods for everything.

■ Lots of static methods. In fact, all methods should be nonstatic unless you have a truly good reason to make them static. This is OO. We don't have global variables and functions. There's no "start here and then keep executing linearly except when you branch, of course…". This is OO, and that means objects all the way down.

Make Variables and Methods As Self-Explanatory As Possible

Don't use variable names like x and y. What the heck does this mean: i nt x = 27; 27 what? Unless you really think you can lock up job security by making sure nobody can understand your code (and assuming the homicidal maniac who tries won't find you), then you should make your identifiers as meaningful as possible.They don't have to be paragraphs. In fact, if it takes a paragraph to explain what a variable represents, perhaps you need to think about your design again. Or at the least, use a comment. But don't make them terse! Take a lesson from the core APIs. They could have called ArInBException, but instead they called it

ArrayIndexOutOfBoundsException. Is there any question about what that exception represents? Of course, the big Sun faux pas was the infamous NullPointerException. But despite the use of the forbidden word pointer, everybody knows what it means when they get it. But there could be some confusion if it were called NPTException or even NullException.

Use the Core APIs!

Do not reinvent the wheel, and do not —or you'll automatically fail for certain use any libraries other than code you developed and the core Java APIs. Resist any temptation to think that you can build something faster, cleaner, more efficient, *etc*. Even if that's true, it isn't worth giving up the benefit of using standard classes that others are familiar with, and that have been extremely, heavily tested in the field.

Make Your Own Exception Classes If You Can't Find One That Suits Your Needs

If there isn't a perfect checked Exception class for you in java.lang, then create your own. And make it specific enough to be meaningful to the catcher. In other words, don't make a BadThingHappenedException and throw it for every possible business error that occurs in your program.

Do Not Return Error Codes!

This is Java. This is OO. If you really need to indicate an exceptional condition, use an Exception! If you really want to annoy an assessor, use error codes as return values from some of your methods. Even one method might do the trick.

Make Your Exceptions with a String Constructor Argument

Doing so gives you a chance to say more about what happened to cause the exception. When you instantiate an Exception, call the constructor that takes a String (or the one that takes another lowerlevel exception if you're doing exception chaining). When you create your own Exception class, be sure to put in a constructor that takes a String.

Follow Basic OO Design Principles

In the preceding section, some of the key points touched on areas we'll dig a bit deeper into here. You don't have to be the World's Best OO Designer, but you do need to follow the basic principles on which the benefits of OO depend. Obviously we can't make this a "How to Be a Good OO Designer in 10 Easy Pages." You need a lot more study and practice, which we assume you've already done. This should be old news by now, but you can bet that your assessor will be looking at these issues, so a refresher won't hurt. We're hitting the highlights of areas where you might get points deducted from your assignment.

Hide Implementation Details

This applies in so many places, but coding with interfaces and using encapsulation is the best way to do it. If you think of your code as little selfcontained, pluggable components, then you don't want anyone who uses one of your components to have to think about how it does what it does. It all comes down to inputs and outputs. A public interface describes what a method needs from you, and what it will return back to you. It says nothing about how that's accomplished. You get to change your implementation (even the class doing the implementing) without affecting calling code. Implementation details can also be propagated through exceptions, so be careful that you don't use an interface but then put implementationspecific exceptions in the throws clause! If a client does a "search," they shouldn't have to catch an SQLException, for example. If your implementation code happens to be doing database work that can generate SQLExceptions (like JDBC code would), the client should not have to know that. It's your job to catch that implementationspecific exception and throw something more meaningful a business-specific exception back to client code.

Use Appropriate Class Granularity

A class should be of the right, you know, granularity. It shouldn't be too big or too tin y. Rarely is the problem a class that's too small; however, most not-quite-OO programmers make classes that are too big. A class is supposed to represent a thing that has state and behaviors. Keep asking yourself, as you write each method, if that behavior might not be better suited for some other thing. For example, suppose younhave a Kitchen class that does all sorts of Kitchen things. Like Oven things and Refrigerator things, *etc.* So now you've got Kitchen things (Kitchen being a room) and Refrigerator things and Oven things all in the same class.

That's three different things. Classes (and thus the objects instantiated from them) really should be specialists. They should do the kinds of behaviors that a thing of that type should do, and no more. So rather than having the Kitchen class include all the code for Refrigerator and Oven behaviors, have the Kitchen class use a Refrigerator and Oven in a HAS-A relationship.

This keeps all three classes simple, and reusable. And that solves your naming problem, so that you don´t have to name your do-everything Kitchen class KitchenFridgeOven.

Another possible cause of a Big Class is that you've got too many inner classes defined. Too many meaning some of the inner classes should have been either top-level classes (for reuse) or simply methods of the enclosing class. Make sure your inner or nested classes really need to be included.

Limit Subclassing

If you need to make a new subclass to add important functionality, perhaps that functionality should really be in the parent class (thus eliminating the need for the subclass—you just need to fix the superclass). When you feel the need to extend a class, always look at whether the parent class should change, or whether you need composition (which means using HAS-A rather than IS-A relationships). Look in the core Java API for a clue about subclassing versus composition: the core API inheritance hierarchy is really wide but very shallow. With a few exceptions (like GUI components), most class hierarchies are no more than two to three levels deep.

Use Appropriate Method Granularity

Just as classes should be specialists, so too should methods. You'll almost certainly be docked points for your assignment if your methods are long (although in some cases, especially in your Swing GUI code, long methods aren't necessarily a reflection of bad design). In most cases, though, the longer the method the more complex, because often a long method is a reflection of a method doing too much. You're all programmers so we don't have to hammer the point about smaller modular functionality much easier to debug, modify, reuse, *etc.* Always see if it makes sense to break a longer method up into smaller ones. But while in a deadline crunch you might get away with long methods in the real world (feeling guilty of course), it won't fly for your Developer assignment.

Use Encapsulation

Your assignment will be scrutinized for this most fundamental OO principle. Expect the assessor to look at the way in which you've controlled access to the state of your object. In other words, the way you've protected your instance variables with setters and getters. No need to discuss it here, just do it. Allow access to your data (except for constants, of course) only through more accessible methods. Be careful about your access modifiers. Having a nice set of accessor methods doesn't matter if you've left your variables wide-open for direct access. Again, make things as private and scope-limited as you can.

Isolate Code That Might Change from Code That Won't Have To

When you design your classes, be sure to separate out the functionality that might change into separate classes. That way, you restrict the places where you'll have to track down and make modifications as the program evolves.

Use Core APIs

Always always always check the core APIs, and know that occasionally you might find the class you're looking for in a package other than where you'd expect it. So be sure to really search through the APIs, even digging into packages and classes you might think are a little off the path. Sometimes a solution can be where you least expect it, so stay open to approaches that aren't necessarily the ones you would normally take. Flipping through a reference API book can help. A method might catch your eye and even if it turns out not to be your solution, it might spark an idea about a different solution. In some cases, you might not find exactly what you're looking for, but you might find a class you can extend, thus inheriting a bunch of functionality that you now won't have to write and test (subject to the warnings about subclassing we mentioned previously). Using core API's (besides being essential for the exam) lets you take advantage of a ton of expertise and testing, plus you're using code that hundreds of thousands of other Java developers are familiar with.

Use Standard Design Patterns

We can't tell you which ones you'll actually need for your assignment; that depends on both your assignment and your particular approach. But there are plenty of standard design patterns that let you take advantage of the collective experience of all those who've struggled with your issue before you (although usually at a fairly abstract level—that's usually where most patterns do their work). So while the core APIs let you take advantage of someone else's implementation code, design patterns let you take advantage of someone else's approach to a problem. If you put a gun to our heads, though, we'd probably have to say that Singleton should be way up on your list of things to consider when developing your assignment.

But you might also take a look at MVC (for your client GUI), Façade, Decorator, Observer, Command, Adapter, Proxy, and Callback, for starters. Pick up a book on design patterns (the classic reference is known as the “Gang of Four” (GOF) book,

Design Patterns: Elements of Reusable ObjectOriented Software, by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides) and take time to step back and look at where your program might be trying to do something well-solved by a design pattern. The patterns don’t tell you how to construct your algorithms and implement your code line by line, but they can guide you into a sound and maintainable design. Perhaps most importantly, as design patterns are becoming more and more well-known, developers have a common vocabulary to discuss design trade-offs and decisions. We believe that the use of design patterns has recently become more important in the exam assessment than it has been in the past, due in large part to their growth in popularity.

Handle Errors Appropriately

You’ll be evaluated for appropriate and clear error -handling throughout your project. You might do really well with it in your GUI and then fall down in your server, but it matters everywhere in your program.

Don’t Return Error Codes This is Java. Using error codes as return values, rather than using exceptions, is a Really Bad Idea. We’re pretty sure your exam assessor knows that.
Don’t Send Out Excessive CommandLine Messages

Don’t be too verbose with your command -line messages, and be sure not to leave debugging messages in! Your commandline messages should include only what’s necessary to verify the startup of your programs and a very minimal amount of status messages that might be crucial if the program fails. But in general, if something goes wrong that you know could go wrong, you should be handling it with exceptions. Whatever you do, don’t use commandline messages to send alert messages to the user! Use a proper dialog box if appropriate.

Use Dialog Boxes Where Appropriate

On the other hand, don’t use dialog boxes for every possible message the user might need to know about. If you need to display information to the user that isn’t of an urgent nature (urgent being things like a record-locking problem or if you need to offer a “Are you sure you want to Quit?” option). In many cases, a dialog box is what you’ll use to alert the user when something in your program has caught an exception, and you need user input to deal with it appropriately.

Throw Checked Exceptions Appropriately There’s a correct time and place for throwing checked exceptions, and being reluctant to throw them can be just as bad as throwing them carelessly.

■ Use runtime exceptions for programming errors. ■ Use checked exceptions for things that your code might recover from (possibly with help from the user).

■ Checked exceptions are only for truly exceptional conditions.
■ Do not use exceptions for flow control! Well, not if you hope to do well both on the exam and in real life.
Remember, checked exceptions sure don’t come for free at runtime; they’ve got overhead. Use them when, but only when, you need them.
Create and Throw Your Own Exceptions When Appropriate

Make use of standard exceptions when they make sense, but never hesitate to create your own if appropriate. If there’s a reasonable chance that an exceptional condition can be recovered from, then use a checked exception and try to handle it. Normally, the exceptions that you create can be thought of as Business Exceptions—in other words, things like “RecordLockedException” or “InsufficientSearchCriteriaException”.

The more specific your exception, the more easily your code can handle it, and you get the benefit of providing specific catch blocks, thus keeping the granularity of your catch blocks useful. The opposite of that strategy would be to simply have everything in one big try block that catches Exception (or worse, Throwable!).

Catch Low-Level Implementation Exceptions

and Throw a Higher-Level Business Exception Say you catch an SQLException (not likely on the Developer exam). Do you throw this back to a client? Of course not. For a client, it falls into the category of “too much information.” The client should not know—or care—that the database server happens to be using SQL. Instead, throw back to the client a more meaningful custom business exception that he or she can deal with.

That more meaningful business exception is defined in your public interface, so the client is expecting it as a possibility. But simply passing a low-level exception all the way to a client reflects a poor design, since it couples the client with implementation details of the server that's never a good idea in an OO design.

Never, Ever, Ever Eat an Exception
By eat we mean the following horrible practice:

```
try {
doRiskyThing();
}
catch(Exception e) {}
```

See what's missing? By catching the exception and then not handling it in any way , it goes completely unnoticed, as if it never occurred. You should at the least print the stack trace. Putting something like this in your exam project might be the death blow.

Understanding Core Java Database Issues

Introduction:

This is very important area according to me , This is the area where your solution to the problem is going to have the greatest impact on your score. You're going to be asked to build a database. From scratch. And since there will be concurrent clients (or at least the possibility of concurrent clients), you'll have to be certain dead certain that you correctly manage record locking.

How you implement your searching, updating, and locking mechanism is entirely up to you. Again, there is definitely no One Right Answer for your solutions to these issues. But however you choose to do it, be certain that the logic is sound. For example, even if you never experience deadlock during testing, if there's even the slightest possibility (no matter how remote the chance) that it could happen, you could easily fail the exam even if nearly everything else in your application is perfect. The two biggest issues are locking and searching, but locking is where the Big Money really is. We'll start with a brief overview of the key concepts, followed by yet another inspiring list of thought provoking questions.

Building a java Database
If you remember from Chapter 10, you're the one who has to build the database; the client's too cheap or neurotic to invest in a commercial database, even a free one. So what is a database?

neurotic to invest in a commercial database, even a free one. So what is a database?

That depends on your assignment, but for the purposes of the exam, software-that lets-youaccess-a-set-of-records will do. You have some data, in some file format somewhere, with a known schema, and your job is to write an application that allows that data to be searched and modified.

You might also need to add and delete records. So the concept is simple: the client makes a request, based on some search criteria, and your database returns a result. Sometimes the client might want to, say, book a Horse Cruise, in which case one or more records will have to be updated. And you might need to insert a new cruise or delete a cancelled cruise. Regardless of the actual scenario, the Really Big Issue is

How do I protect the data from concurrent access? In other words, how do I lock the records?

NOTE: Your locking design and implementation decisions (and execution) are the most important parts of your Developer assignment. Spend the greatest percentage of your time making sure you have a sound solution. Be sure you've met any requirements in your assignment document that pertain to locking and unlocking. If part of your assignment specification is vague or ambiguous, you need to make an interpretation (your best guess about what to do) and then document your assumption and strategy.

And remember, the clients could be either local or remote (in other words, on the same machine as the database or on a machine somewhere else on the network), so you'll have to think of issues related to both of those scenarios. Locking is crucial, but fortunately the Developer exam isn't asking you to implement a complete distributed transaction system using the two-phase commit protocol. In fact, this is much simpler than transactions, but it will require you to understand the fundamental issues surrounding concurrent access to data. Remember the bank account example from Chapter 9? The one where the husband and wife both shared the same account? If you're not absolutely clear about how to handle synchronization, then reread that chapter. In order to correctly implement your locking strategy, you're going to need a solid grasp on synchronization, wait(), notify(), and notifyAll(). So, ready for some questions? Once again, these are in no particular order.

Questions to Ask Yourself

We've split these into two categories, searching and locking. But there's a lot about searching that also falls into the category of GUI issues (Chapter 13). Specifically, you'll need to be certain that your end-users know how to build a search query.

Searching

■ How easy is it for clients to perform a search? Assuming the GUI itself is user-friendly (and we have a lot to say about that in Chapter 13), what about the criteria?

■ How are the search criteria items represented? A String? A CriteriaObject?
■ How does a client know exactly what they can base a search on?
■ Does your search support boolean matches? Does it need to?
■The database won't necessarily be indexed, so have you thought about other ways to make the search as efficient as possible?

NOTE: Don't sacrifice clarity and simplicity for a small performance gain. If the performance gain is big, then redesign so that you can have a reasonably efficient algorithm that is also clear and maintainable.

■ ***Have you documented your search algorithm?***
If you find yourself writing a lot of documentation to explain your search algorithm, there's probably something wrong with the design.
■ ***Is the documentation of your search algorithm easy to read and understand?***
■ ***When the client submits a search query, is a specific piece of the search criteria explicitly matched to a particular field? Or do you search all fields for each search?***

■ ***If you're using 1.4, have you investigated whether regular expressions would help?*** ■***What happens if nothing matches the client's search criteria?*** ■ ***Will it need to be an exact match?*** ■ ***Could there be a scenario in which too many records match the search criteria?***

■ ***Have you considered bandwidth issues when designing and implementing the format of the search criteria requests and server results? Are you shipping things over the wire that are bigger than they need to be?***

■ ***Is your search capability flexible for the end-user?***
■ ***Is your search capability flexible for future changes to the program?***

■ ***How much code, if any, would have to change if the database schema changes? Have you isolated the places where changes can occur to avoid maintenance problems?*** ■***Are you absolutely certain that you've met the searching requirements defined in your assignment specification? Go back and reread them. Slooooooowly.***

Locking
■ ***Are you absolutely certain that your locking scheme works in all possible scenarios?***

■ Does your exam assignment specify a particular kind of locking with respect to reads and writes?
■ What happens when a client attempts to get a record and the record is already locked? What does the client experience?

NOTE: This is crucial. Think long and hard about what you want to happen.
■ How will you keep track of which records are locked?
■ How will you keep track of who locked each record? Do you need to know that?

■ How will you uniquely identify clients in such a way that you can know which client locked which record? Is it the server's responsibility or the client's?
■ Have you considered whether the ID of a thread is appropriate to uniquely identify a client? ■ Have you considered whether a Math.random() number is appropriate to uniquely identify a client?
■If a client makes a request on a locked record, how will you verify that it's the same client who holds the lock?
■ What happens if a client attempts to use a locked record when that client is not the client holding the lock?

■ Is it possible to have a record locked for too long a time? How much time is too long? ■ Is there anything you can or should do about the duration of a lock?
■ What happens if a client goes down without releasing a lock?

■ Does the server need a way to know a client went down? (As opposed to simply taking their sweet time or if they're on a painfully slow connection.)
■ Is there any possibility of a deadlock? Where two or more clients are waiting for each other's locks?

NOTE: Check for this more than you check for anything else. ■ Are you correctly using wait(), notify(), and notifyAll()? ■ Are you clear about the implications of notify() versus notifyAll()?

Interview Part 4

Core java interview questions and answers for freshers
**1.How could Java classes direct program messages to the system console, but error messages, say to a
file?**
**The class System has a variable out that represents the standard output, and the variable err that
represents the standard error device. By default, they both point at the system console. This how the
standard output could be redirected:**
**Stream st = new Stream(new FileOutputStream("output.txt"));
System.setErr(st); System.setOut(st);**
2.How would you create a button with rounded edges?
**There's 2 ways. The first thing is to know that a JButton's edges are drawn by a Border. so you can
override the Button's paintComponent(Graphics) method and draw a circle or rounded rectangle
(whatever), and turn off the border. Or you can create a custom border that draws a circle or rounded
rectangle around any component and set the button's border to it. 3.Why should the implementation of any Swing callback (like a listener) execute quickly?**
**A: Because callbacks are invoked by the event dispatch thread which will be blocked processing other
events for as long as your method takes to execute.**
4. Question: How you can force the garbage collection?
**Garbage collection automatic process and can't be forced. You could request it by calling System.gc().
JVM does not guarantee that GC will be started immediately.**
**Garbage collection is one of the most important feature of Java, Garbage collection is also called
automatic memory management as JVM automatically removes the unused variables/objects (value is
null) from the memory. User program can't directly free the object from memory, instead it is the job of
the garbage collector to automatically free the objects that are no longer**

referenced by a program.
Every class inherits finalize() method from java.lang.Object, the finalize() method is called by garbage
collector when it determines no more references to the object exists. In Java, it is good idea to explicitly
assign null into a variable when no more in use. I Java on calling System.gc() and Runtime.gc(), JVMtries
to recycle the unused objects, but there is no guarantee when all the objects will garbage collected.
5. What's the difference between constructors and normal methods?

Constructors must have the same name as the class and can not return a value. They are only called
once while regular methods could be called many times and it can return a value or can be void.

6. When should the method invokeLater()be used?
This method is used to ensure that Swing components are updated through the eventdispatching
thread.
7.Explain the usage of Java packages.
This is a way to organize files when a project consists of multiple modules. It also helps resolve naming
conflicts when different packages have classes with the same names. Packages access level also allows
you to protect data from being used by the non-authorized classes.
8.What are some advantages and disadvantages of Java Sockets?
Some advantages of Java Sockets:
Sockets are flexible and sufficient. Efficient socket based programming can be easily implemented for
general communications. Sockets cause low network traffic. Unlike HTML forms and CGI scripts that
generate and transfer whole web pages for each new request, Java applets can send only necessary
updated information.
Some disadvantages of Java Sockets:
Security restrictions are sometimes overbearing because a Java applet running in a Web browser is only

able to establish connections to the machine where it came from, and to

nowhere else on the network
Despite all of the useful and helpful Java features, Socket based communications allows only to send

packets of raw data between applications. Both the client-side and server-side have to provide
mechanisms to make the data useful in any way.
9.What is Collection API?
The Collection API is a set of classes and interfaces that support operation on collections of objects.
These classes and interfaces are more flexible, more powerful, and more regular than the vectors,
arrays, and hashtables if effectively replaces.
Example of classes: HashSet, HashMap, ArrayList, LinkedList, TreeSet and TreeMap.
Example of interfaces: Collection, Set, List and Map.
10.Explain the usage of the keyword transient?
Transient keyword indicates that the value of this member variable does not have to be serialized with
the object. When the class will be deserialized, this variable will be initialized with a default value of its
data type (i.e. zero for integers).
11.What's the difference between the methods sleep() and wait()
The code sleep(1000); puts thread aside for exactly one second. The code wait(1000), causes a wait of
up to one second. A thread could stop waiting earlier if it receives the notify() or notifyAll() call. The
method wait() is defined in the class Object and the method sleep() is defined in the class Thread.
12.Why would you use a synchronized block vs. synchronized method?
Synchronized blocks place locks for shorter periods than synchronized methods.
13.Can an inner class declared inside of a method access local variables of this method?
It's possible if these variables are final.
14.Explain the user defined Exceptions?
User defined Exceptions are the separate Exception classes defined by the user for specific purposed. An
user defined can created by simply subclassing it to the Exception class.

This allows custom exceptions
to be generated (using throw) and caught in the same way as normal exceptions.
Example:

```
class myCustomException extends Exception {
/ The class simply has to exist to be an exception
}
```

15.Describe the wrapper classes in Java.
Wrapper class is wrapper around a primitive data type. An instance of a wrapper class contains, or
wraps, a primitive value of the corresponding type.
Following table lists the primitive types and the corresponding wrapper classes:
Primitive Wrapper
boolean – java.lang.Boolean
byte – java.lang.Byte
char – java.lang.Character
double – java.lang.Double
float – java.lang.Float
int – java.lang.Integer long – java.lang.Long
short – java.lang.Short
void – java.lang.Void
16.Which of the following are valid definitions of an application’s main( ) method?
a) public static void main();
b) public static void main(String args);
c) public static void main(String args[]);
d) public static void main(Graphics g); e) public static boolean main(String args[]);

JAVA INTERVIEW QUESTIONS PART 4

1. What is a transient variable?
A transient variable is a variable that may not be serialized.
2. Which containers use a border Layout as their default layout?
The window, Frame and Dialog classes use a border layout as their default layout.

3. Why do threads block on I/O?
Threads block on I/O (that is enters the waiting state) so that other threads may execute while the I/O

Operation is performed.
4. How are Observer and Observable used?
Objects that subclass the Observable class maintain a list of observers. When an Observable object is
updated it invokes the update() method of each of its observers to notify the observers that it has
changed state. The Observer interface is implemented by objects that observe Observable objects.
5. What is synchronization and why is it important?
With respect to multithreading, synchronization of multiple
is the capability to control the access threads to shared resources. Without synchronization, it is possible for one thread to modify a shared
object while another thread is in the process of using or updating that object's value. This often leads to
significant errors.
6. Can a lock be acquired on a class?
Yes, a lock can be acquired on a class. This lock is acquired on the class's Class object.
7. What's new with the stop(), suspend() and resume() methods in JDK 1.2?
The stop(), suspend() and resume() methods have been deprecated in JDK 1.2.
8. Is null a keyword?
The null value is not a keyword.

9. What is the preferred size of a component? The preferred size of a component is the minimum component size that will allow the component to

display normally.
10. What method is used to specify a container's layout?
The setLayout() method is used to specify a container's layout.
11. Which containers use a FlowLayout as their default layout?
The Panel and Applet classes use the FlowLayout as their default layout.
12. What state does a thread enter when it terminates its processing?
When a thread terminates its processing, it enters the dead state.
13. What is the Collections API?
The Collections API is a set of classes and interfaces that support operations on collections of objects.
14. which characters may be used as the second character of an identifier, but not as the first character
of an identifier?
The digits 0 through 9 may not be used as the first character of an identifier but they may be used after
the first character of an identifier.
15. What is the List interface?
The List interface provides support for ordered collections of objects.
16. How does Java handle integer overflows and underflows?
It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.
17. What is the Vector class?
The Vector class provides the capability to implement a growable array of objects 18. What modifiers may be used with an inner class that is a member of an outer class?
A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.
19. What is an Iterator interface?
The Iterator interface is used to step through the elements of a Collection.
20. What is the difference between the >> and >>> operators?
The >> operator carries the sign bit when shifting right. The >>> zero-fills bits that have been shifted out.
21. Which method of the Component class is used to set the position and size of a component?
setBounds()
22. How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?
Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII

character set uses only 7 bits, it is
usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses
16-bit and larger bit patterns.
23 What is the difference between yielding and sleeping?
When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep()
method, it returns to the waiting state.
24. Which java.util classes and interfaces support event handling?
The EventObject class and the EventListener interface support event processing.
25. Is sizeof a keyword?
The sizeof operator is not a keyword.
26. What are wrapper classes? Wrapper classes are classes that allow primitive types to be accessed as objects.
27. Does garbage collection guarantee that a program will not run out of memory?
Garbage collection does not guarantee that a program will not run out of memory. It is possible for
programs to use up memory resources faster than they are garbage collected. It is also possible for
programs to create objects that are not subject to garbage collection.
28. What restrictions are placed on the location of a package statement within a source code file?
A package statement must appear as the first line in a source code file (excluding blank lines and
comments).
29. Can an object's finalize() method be invoked while it is reachable?
An object's finalize() method cannot be invoked by the garbage collector while the object is still
reachable. However, an object's finalize() method may be invoked by other objects.
30. What is the immediate superclass of the Applet class?
Panel
31. What is the difference between preemptive scheduling and time slicing?
Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states
or a higher priority task comes into existence. Under time slicing, a task executes for a predefinedslice

of time and then reenters the pool of ready tasks. The scheduler then determines which task should
execute next, based on priority and other factors. 32. Name three Component subclasses that support painting.
The Canvas, Frame, Panel, and Applet classes support painting.
33. What value does readLine() return when it has reached the end of a file?
The readLine() method returns null when it has reached the end of a file.
34. What is the immediate superclass of the Dialog class?
Window.
35. What is clipping?
Clipping is the process of confining paint operations to a limited area or shape.
36. What is a native method?
A native method is a method that is implemented in a language other than Java.
37. Can a for statement loop indefinitely?
Yes, a for statement can loop indefinitely. For example, consider the following:
for(;;) ;
38. What are order of precedence and associativity, and how are they used?
Order of precedence determines the order in which operators are evaluated in expressions. Associatity
determines whether an expression is evaluated left-to-right or right-to-left
39. When a thread blocks on I/O, what state does it enter?
A thread enters the waiting state when it blocks on I/O.
40. To what value is a variable of the String type automatically initialized?
The default value of a String type is null.
41. What is the catch or declare rule for method declarations?
If a checked exception may be thrown within the body of a method, the method must either catch the
exception or declare it in its throws clause. 42. What is the difference between a MenuItem and a CheckboxMenuItem?
The CheckboxMenuItem class extends the MenuItem class to support a menu item that may be checked
or unchecked.
43. What is a task's priority and how is it used in scheduling?
A task's priority is an integer value that identifies the relative order in which it should be executed with
respect to other tasks. The scheduler attempts to schedule higher priority

tasks before lower priority
tasks.
44. What class is the top of the AWT event hierarchy?
The java.awt.AWTEvent class is the highest-level class in the AWT event-class hierarchy.
45. When a thread is created and started, what is its initial state?
A thread is in the ready state after it has been created and started.
46. Can an anonymous class be declared as implementing an interface and extending a class?
An anonymous class may implement an interface or extend a superclass, but may not be declared to do
both.
47. What is the range of the short type?
The range of the short type is -(2^15) to 2^15 - 1.
48. What is the range of the char type?
The range of the char type is 0 to 2^16 - 1.

49. In which package are most of the AWT events that support the eventdelegation model defined?
Most of the AWT-related events of the eventdelegation model are defined in the java.awt.event

package. The AWTEvent class is defined in the java.awt package.
50. What is the immediate superclass of Menu?
MenuItem
51. What is the purpose of finalization?
The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup
processing before the object is garbage collected.
52. Which class is the immediate superclass of the MenuComponent class.
Object
53. What invokes a thread's run() method?
After a thread is started, via its start() method or that of the Thread class, the JVM invokes the thread's
run() method when the thread is initially executed.
54. What is the difference between the Boolean & operator and the && operator?
If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the &
operator is applied to the operand. When an expression involving the &&

operator is evaluated, the first
operand is evaluated. If the first operand returns a value of true then the second operand is evaluated.
The && operator is then applied to the first and second operands. If the first operand evaluates to false,
the evaluation of the second operand is skipped.

55. Name three subclasses of the Component class. Box.Filler, Button, Canvas, Checkbox, Choice, Container, Label, List, Scrollbar, or TextComponent

56. What is the GregorianCalendar class?
The GregorianCalendar provides support for traditional Western calendars.
57. Which Container method is used to cause a container to be laid out and redisplayed?
validate()
58. What is the purpose of the Runtime class?
The purpose of the Runtime class is to provide access to the Java runtime system.
59. How many times may an object's finalize() method be invoked by the garbage collector?
An object's finalize() method may only be invoked once by the garbage collector.
60. What is the purpose of the finally clause of a trycatch-finally statement?
The finally clause is used to provide the capability to execute code no matter whether or not an
exception is thrown or caught.
61. What is the argument type of a program's main() method?
A program's main() method takes an argument of the String[] type.
62. Which Java operator is right associative?
The = operator is right associative.
63. What is the Locale class?
The Locale class is used to tailor program output to geographic, political, or cultural region.64. Can a double value be cast to a byte?
Yes, a double value can be cast to a byte. the conventions of a particular

65. What is the difference between a break statement and a continue statement? A break statement results in the termination of the statement to

which it applies (switch, for, do, or

while). A continue statement is used to end the current loop iteration and return control to the loop
statement.
66. What must a class do to implement an interface?
It must provide all of the methods in the interface and identify the interface in its implements clause.
67. What method is invoked to cause an object to begin executing as a separate thread?
The start() method of the Thread class is invoked to cause an object to begin executing as a separate
thread.
68. Name two subclasses of the TextComponent class.
TextField and TextArea
69. What is the advantage of the eventdelegation model over the earlier eventinheritance model?
The eventdelegation model has two advantages over the eventinheritance model. First, it enables
event handling to be handled by objects other than the ones that generate the events (or their
containers). This allows a clean separation between a component's design and its use. The other
advantage of the eventdelegation model is that it performs much better in applications where many
events are generated. This performance improvement is due to the fact that the eventdelegation model
does not have to repeatedly process unhandled events, as is the case of the eventinheritance model.
70. Which containers may have a MenuBar?
Frame
71. How are commas used in the initialization and iteration parts of a for statement?
Commas are used to separate multiple statements within the initialization and iteration parts of a for
statement.
72. What is the purpose of the wait(), notify(), and notifyAll() methods?
The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to wait for a
shared resource. When a thread executes an object's wait() method, it

shared resource. When a thread executes an object's wait() method, it enters the waiting state. It only
enters the ready state after another thread invokes the object's notify() or notifyAll() methods.
73. What is an abstract method?
An abstract method is a method whose implementation is deferred to a subclass.
74. How are Java source code files named?
A Java source code file takes the name of a public class or interface that is defined within the file. A
source code file may contain at most one public class or interface. If a public class or interface is defined
within a source code file, then the source code file must take the name of the public class or interface. If
no public class or interface is defined within a source code file, then the file must take on a namethat is
different than its classes and interfaces. Source code files use the .java extension. 75. What is the relationship between the Canvas class and the Graphics class?
A Canvas object provides access to a Graphics object via its paint() method.
76. What are the high-level thread states?
The high-level thread states are ready, running, waiting, and dead.
77. What value does read() return when it has reached the end of a file?
The read() method returns -1 when it has reached the end of a file.
78. Can a Byte object be cast to a double value?
No, an object cannot be cast to a primitive value.
79. What is the difference between a static and a nonstatic inner class?
A nonstatic inner class may have object instances that are associated with instances of the class's outer
class. A static inner class does not have any object instances.
80. What is the difference between the String and StringBuffer classes?
String objects are constants. StringBuffer objects are not.
81. If a variable is declared as private, where may the variable be accessed?
A private variable may only be accessed within the class in which it is declared.
82. What is an object's lock and which objects have locks?
An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the
object. A thread may execute a synchronized method of an object only after it has acquired the object's

lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.
83. What is the Dictionary class?
The Dictionary class provides the capability to store key-value pairs.

84. How are the elements of a BorderLayout organized? The elements of a BorderLayout are organized at the borders (North, South, East, and West) and the

center of a container.
85. What is the % operator?
It is referred to as the modulo or remainder operator. It returns the remainder of dividing the first
operand by the second operand.
86. When can an object reference be cast to an interface reference?
An object reference be cast to an interface reference when the object implements the referenced
interface.
87. What is the difference between a Window and a Frame?
The Frame class extends Window to define a main application window that can have a menu bar.
88. Which class is extended by all other classes?
The Object class is extended by all other classes.
89. Can an object be garbage collected while it is still reachable?
A reachable object cannot be garbage collected. Only unreachable objects may be garbage collected..
90. Is the ternary operator written x : y ? z or x ? y : z ?
It is written x ? y : z.
91. What is the difference between the Font and FontMetrics classes?
The FontMetrics class is used to define implementationspecific properties, such as ascent and descent,
of a Font object.
92. How is rounding performed under integer division? The fractional part of the result is truncated. This is known as rounding toward zero.
93. What happens when a thread cannot acquire a lock on an object?
If a thread attempts to execute a synchronized method or synchronized statement and is unable to
acquire an object's lock, it enters the waiting state until the lock becomes available.

94. What is the difference between the Reader/Writer class hierarchy and the
InputStream/OutputStream class hierarchy?
The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class
hierarchy is byte-oriented.
95. What classes of exceptions may be caught by a catch clause?
A catch clause can catch any exception that may be assigned to the Throwable type. This includes the
Error and Exception types.
96. If a class is declared without any access modifiers, where may the class be accessed?
A class that is declared without any access modifiers is said to have package access. This means that the
class can only be accessed by other classes and interfaces that are defined within the same package.
97. What is the SimpleTimeZone class?
The SimpleTimeZone class provides support for a Gregorian calendar.
98. What is the Map interface?
The Map interface replaces the JDK 1.1 Dictionary class and is used associate keys with values.99. Does a
class inherit the constructors of its superclass?
A class does not inherit constructors from any of its super classes. 100. For which statements does it make sense to use a label?
The only statements for which it makes sense to use a label are those statements that can enclose a
break or continue statement.
101. What is the purpose of the System class?
The purpose of the System class is to provide access to system resources.
102. Which TextComponent method is used to set a TextComponent to the readonly state?
setEditable()
103. How are the elements of a CardLayout organized?
The elements of a CardLayout are stacked, one on top of the other, like a deck of cards.
104. Is &&= a valid Java operator?
No, it is not.
105. Name the eight primitive Java types.
The eight primitive types are byte, char, short, int, long, float, double, and

boolean.
106. Which class should you use to obtain design information about an object?
The Class class is used to obtain information about an object's design.
107. What is the relationship between clipping and repainting?
When a window is repainted by the AWT painting thread, it sets the clipping regions to the area of the
window that requires repainting.
108. Is "abc" a primitive value?
The String literal "abc" is not a primitive value. It is a String object.
109. What is the relationship between an event-listener interface and an event-adapter class?

An event-listener interface defines the methods that must be implemented by an event handler for a
particular kind of event. An event adapter provides a default implementation of an event-listener

interface.
110. What restrictions are placed on the values of each case of a switch statement?
During compilation, the values of each case of a switch statement must evaluate to a value that can be
promoted to an int value.
111. What modifiers may be used with an interface declaration?
An interface may be declared as public or abstract.
112. Is a class a subclass of itself?
A class is a subclass of itself.
113. What is the highest-level event class of the eventdelegation model?
The java.util.EventObject class is the highest-level class in the eventdelegation class hierarchy.
114. What event results from the clicking of a button?
The ActionEvent event is generated as the result of the clicking of a button.
115. How can a GUI component handle its own events?
A component can handle its own events by implementing the required event-listener interface and
adding itself as its own event listener.
116. What is the difference between a while statement and a do statement?
A while statement checks at the beginning of a loop to see whether the next loop iteration should occur

loop iteration should occur?
A do statement checks at the end of a loop to see whether the next iteration of a loop should occur. The
do statement will always execute the body of a loop at least once. 117. How are the elements of a GridBagLayout organized?
The elements of a GridBagLayout are organized according to a grid. However, the elements are of
different sizes and may occupy more than one row or column of the grid. In addition, the rows and
columns may have different sizes.
118. What advantage do Java's layout managers provide over traditional windowing systems?
Java uses layout managers to lay out components in a consistent manner across all windowing
platforms. Since Java's layout managers aren't tied to absolute sizing and positioning, they are able to
accommodate platform-specific differences among windowing systems.
119. What is the Collection interface?
The Collection interface provides support for the implementation of a mathematical bag - an unordered
collection of objects that may contain duplicates.
120. What modifiers can be used with a local inner class?
A local inner class may be final or abstract.
121. What is the difference between static and nonstatic variables?
A static variable is associated with the class as a whole rather than with specific instances of a class.
Nonstatic variables take on unique values with each object instance.
122. What is the difference between the paint() and repaint() methods?
The paint() method supports painting via a Graphics object. The repaint() method is used to cause
paint() to be invoked by the AWT painting thread. 123. What is the purpose of the File class?
The File class is used to create objects that provide access to the files and directories of a local file
system.
124. Can an exception be rethrown?
Yes, an exception can be rethrown.
125. Which Math method is used to calculate the absolute value of a number?
The abs() method is used to calculate absolute values.

The abs() method is used to calculate absolute values.
126. How does multithreading take place on a computer with a single CPU?
The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching
between executing tasks, it creates the impression that tasks execute sequentially.
127. When does the compiler supply a default constructor for a class?
The compiler supplies a default constructor for a class if no other constructors are provided.
128. When is the finally clause of a trycatch-finally statement executed?
The finally clause of the trycatch-finally statement is always executed unless the thread of execution
terminates or an exception occurs within the execution of the finally clause.
129. Which class is the immediate superclass of the Container class?
Component
130. If a method is declared as protected, where may the method be accessed?
A protected method may only be accessed by classes or interfaces of the same package or by subclasses
of the class in which it is declared.
131. How can the Checkbox class be used to create a radio button? By associating Checkbox objects with a CheckboxGroup.
132. Which non-Unicode letter characters may be used as the first character of an identifier?
The non-Unicode letter characters $ and _ may appear as the first character of an identifier
133. What restrictions are placed on method overloading?
Two methods may not have the same name and argument list but different return types.
134. What happens when you invoke a thread's interrupt method while it is sleeping or waiting?
When a task's interrupt() method is executed, the task enters the ready state. The next time the task
enters the running state, an InterruptedException is thrown.
135. What is casting?
There are two types of casting, casting between primitive numeric types and casting between object
references. Casting between numeric types is used to convert larger values, such as double values, to
smaller values, such as byte values. Casting between object references is

used to refer to an objectby a
compatible class, interface, or array type reference.
136. What is the return type of a program's main() method?
A program's main() method has a void return type.
137. Name four Container classes.
Window, Frame, Dialog, FileDialog, Panel, Applet, or ScrollPane
138. What is the difference between a Choice and a List?

A Choice is displayed in a compact form that requires you to pull it down to see the list of available
choices. Only one item may be selected from a Choice. A List may be displayed in such a way that

several List items are visible. A List supports the selection of one or more List items.
139. What class of exceptions are generated by the Java runtime system?
The Java runtime system generates RuntimeException and Error exceptions.
140. What class allows you to read objects directly from a stream?
The ObjectInputStream class supports the reading of objects from input streams.
141. What is the difference between a field variable and a local variable?
A field variable is a variable that is declared as a member of a class. A local variable is a variable that is
declared local to a method.
142. Under what conditions is an object's finalize() method invoked by the garbage collector?
The garbage collector invokes an object's finalize() method when it detects that the object has become
unreachable.
143. How are this () and super () used with constructors?
this() is used to invoke a constructor of the same class. super() is used to invoke a superclass
constructor.
144. What is the relationship between a method's throws clause and the exceptions that can be thrown
during the method's execution?
A method's throws clause must declare any checked exceptions that are not caught within the body of

the method. 145. What is the difference between the JDK 1.02 event model and the eventdelegation model

introduced with JDK 1.1?
The JDK 1.02 event model uses an event inheritance or bubbling approach. In this model, components
are required to handle their own events. If they do not handle a particular event, the event is inherited
by (or bubbled up to) the component's container. The container then either handles the event or it is
bubbled up to its container and so on, until the highest-level container has been tried.
In the eventdelegation model, specific objects are designated as event handlers for GUI components.
These objects implement event-listener interfaces. The eventdelegation model is more efficient than
the eventinheritance model because it eliminates the processing required to support the bubbling of
unhandled events.
146. How is it possible for two String objects with identical values not to be equal under the ==
operator?
The == operator compares two objects to determine if they are the same object in memory. It is possible
for two String objects to have the same value, but located indifferent areas of memory.
147. Why are the methods of the Math class static?
So they can be invoked as if they are a mathematical code library.
148. What Checkbox method allows you to tell if a Checkbox is checked?
getState() 149. What state is a thread in when it is executing?
An executing thread is in the running state.
150. What are the legal operands of the instanceof operator?
The left operand is an object reference or null value and the right operand is a class, interface, or array
type.
151. How are the elements of a GridLayout organized?
The elements of a GridBad layout are of equal size and are laid out using the squares of a grid.
152. What an I/O filter?

An I/O filter is an object that reads from one stream and writes to another, usually altering the data in
some way as it is passed from one stream to another.
153. If an object is garbage collected, can it become reachable again?
Once an object is garbage collected, it ceases to exist. It can no longer become reachable again.
154. What is the Set interface?
The Set interface provides methods for accessing the elements of a finite mathematical set. Sets do not
allow duplicate elements.
155. What classes of exceptions may be thrown by a throw statement?
A throw statement may throw any expression that may be assigned to the Throwable type.
156. What are E and PI?
E is the base of the natural logarithm and PI is mathematical value pi.
157. Are true and false keywords?
The values true and false are not keywords. 158. What is a void return type?
A void return type indicates that a method does not return a value.
159. What is the purpose of the enableEvents() method?
The enableEvents() method is used to enable an event for a particular object. Normally, an event is
enabled when a listener is added to an object for a particular event. The enableEvents() method is used
by objects that handle events by overriding their eventdispatch methods.
160. What is the difference between the File and RandomAccessFile classes?
The File class encapsulates the files and directories of the local file system. The RandomAccessFile class
provides the methods needed to directly access data contained in any part of a file.
161. What happens when you add a double value to a String?
The result is a String object.162. What is your platform's default character encoding?
If you are running Java on English Windows platforms, it is probably Cp1252. If you are running Javaon
English Solaris platforms, it is most likely 8859_1..
163. Which package is always imported by default?
The java.lang package is always imported by default.
164. What interface must an object implement before it can be written to a

stream as an object?
An object must implement the Serializable or Externalizable interface before it can be written to a
stream as an object.

165. How are this and super used? this is used to refer to the current object instance. super is used to refer to the variables and methods of

the superclass of the current object instance.
166. What is the purpose of garbage collection?
The purpose of garbage collection is to identify and discard objects that are no longer needed by a
program so that their resources may be reclaimed and
reused.
167. What is a compilation unit?
A compilation unit is a Java source code file.
168. What interface is extended by AWT event listeners?
All AWT event listeners extend the java.util.EventListener interface.
169. What restrictions are placed on method overriding?
Overridden methods must have the same name, argument list, and return type.
The overriding method may not limit the access of the method it overrides.
The overriding method may not throw any exceptions that may not be thrown
by the overridden method.
170. How can a dead thread be restarted?
A dead thread cannot be restarted.
171. What happens if an exception is not caught?
An uncaught exception results in the uncaughtException() method of the thread's ThreadGroup being
invoked, which eventually results in the termination of the program in which it is thrown.
172. What is a layout manager?
A layout manager is an object that is used to organize components in a container. 173. Which arithmetic operations can result in the throwing of an ArithmeticException?
Integer / and % can result in the throwing of an ArithmeticException.
174. What are three ways in which a thread can enter the waiting state?
A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully

attempting to acquire an object's lock, or by invoking an object's wait() method. It can also enter the
waiting state by invoking its
(deprecated) suspend() method.
175. Can an abstract class be final?
An abstract class may not be declared as final.176. What is the ResourceBundle class?
The ResourceBundle class is used to store localespecific resources that can be loaded by a program to
tailor the program's appearance to the particular locale in which it is being run.
177. What happens if a trycatch-finally statement does not have a catch clause to handle an exception
that is thrown within the body of the try statement?
The exception propagates up to the next higher level trycatch statement (if any) or results in the
program's termination.
178. What is numeric promotion?
Numeric promotion is the conversion of a smaller numeric type to a larger numeric type, so that integer
and floatingpoint operations may take place. In numerical promotion, byte, char, and short values are

converted to int values. The int values are also converted to long values, if necessary. The long and float values are

converted to double values, as required.
179. What is the difference between a Scrollbar and a ScrollPane?
A Scrollbar is a Component, but not a Container. A ScrollPane is a Container. A ScrollPane handles its
own events and performs its own scrolling.
180. What is the difference between a public and a non-public class?
A public class may be accessed outside of its package. A non-public class may not be accessed outside of
its package.
181. To what value is a variable of the boolean type automatically initialized?
The default value of the boolean type is false.
182. Can try statements be nested?

Try statements may be tested.
183. What is the difference between the prefix and postfix forms of the ++ operator?
The prefix form performs the increment operation and returns the value of the increment operation.
The postfix form returns the current value all of the expression and then performs the increment
operation on that value.
184. What is the purpose of a statement block?
A statement block is used to organize a sequence of statements as a single statement group.
185. What is a Java package and how is it used?

A Java package is a naming context for classes and interfaces. A package is used to create a separate
name space for groups of classes and interfaces. Packages are also used to organize related classes and

interfaces into a single API unit and to control accessibility to these classes and interfaces.
186. What modifiers may be used with a top-level class?
A top-level class may be public, abstract, or final.
187. What are the Object and Class classes used for?
The Object class is the highest-level class in the Java class hierarchy. The Class class is used to represent
the classes and interfaces that are loaded by a Java program.
188. How does a try statement determine which catch clause should be used to handle an exception?
When an exception is thrown within the body of a try statement, the catch clauses of the try statement
are examined in the order in which they appear. The first catch clause that is capable of handling the
exception is executed.
The remaining catch clauses are ignored.
189. Can an unreachable object become reachable again?
An unreachable object object's finalize()
may become reachable again. This can happen when the method is invoked and accessible to
the object performs an operation which causes it to become reachable
objects 190. When is an object subject to garbage collection?

objects.190. When is an object subject to garbage collection?
An object is subject to garbage collection when it becomes unreachable to the program in which it is
used. 191. What method must be implemented by all threads?
All tasks must implement the run() method, whether they are a subclass of Thread or implement the
Runnable interface.
192. What methods are used to get and set the text label displayed by a Button object?
getLabel() and setLabel()
193. Which Component subclass is used for drawing and painting?
Canvas
194. What are synchronized methods and synchronized statements?
Synchronized methods are methods that are used to control access to an object. A thread only executes
a synchronized method after it has acquired the lock for the method's object or class. Synchronized
statements are similar to synchronized methods. A synchronized statement can only be executed after a
thread has acquired the lock for the object or class referenced in the synchronized statement195. What
are the two basic ways in which classes that can be run as threads may be defined?
A thread class may be declared as a subclass of Thread, or it may implement the Runnable interface.
196. What are the problems faced by Java programmers who don't use layout managers?
Without layout managers, Java programmers are faced with determining how their GUI will be displayed
across multiple windowing systems and finding a common sizing and positioning that will work within
the constraints imposed by each windowing system. 197. What is the difference between an if statement and a switch statement?
The if statement is used to select among two alternatives. It uses a boolean expression to decide which
alternative should be executed. The switch statement is used to select among multiple alternatives. It
uses an int expression to determine which alternative should be executed.

END.

We Want to Hear from You!

As the reader of this book, you are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way. You can email or write me directly to let me know what you did or didn't like about this book—as well as what we can do to make our books stronger. Please note that I cannot help you with technical problems related to the topic of this book, and that due to the high volume of mail I receive, I might not be able to reply to every message. When you write, please be sure to include this book's title and author as well as your name and phone or email address. I will carefully review your comments and share them with the author and editors who worked on the book.

One more thing don't forget to give us best star Reviews rate on Amazon.com

8. Java Interview Questions & Answers Complete Reference Best Selling Edition 2013 Beginners To Experts Approach Guide.
Authored By: Hariom Choudhary.