**Java Tutorials - Java Example Codes and Tutorials**

Java is great programming language for the development of enterprise grade applications. This programming Language is evolved from a language named Oak. Oak was developed in the early nineties at Sun Microsystems as a platform-independent language aimed at allowing entertainment appliances such as video game consoles and VCRs to communicate . Oak was first slated to appear in television set-top boxes designed to provide video-on-demand services. Oak was unsuccessful so in 1995 Sun changed the name to Java and modified the language to take advantage of the burgeoning World Wide Web.

Java is an object-oriented language, and this is very similar to C++. Java Programming Language is simplified to eliminate language features that cause common programming errors. Java source code files are compiled into a format called bytecode, which can then be executed by a Java interpreter.

If you are beginner read [**New to programming**](http://www.roseindia.net/java/new-to-programming.shtml) section.

**Java programming tutorials:**

* [**Core Java**](http://www.roseindia.net/java/)
* [**Java SE 6**](http://www.roseindia.net/java/jdk6/index.shtml)
* [**Java SE 7**](http://www.roseindia.net/java/jdk7/index.shtml)
* [**Advanced Java**](http://www.roseindia.net/java/Advanced-Java-Tutorials.shtml)
* [**J2ME**](http://www.roseindia.net/j2me/)
* [**JSP**](http://www.roseindia.net/jsp/)
* [**Servlets**](http://www.roseindia.net/servlets/)
* [**JDBC**](http://www.roseindia.net/jdbc/)
* [**EJB**](http://www.roseindia.net/ejb/)
* [**Web Services**](http://www.roseindia.net/webservices/)
* [**JSTL**](http://www.roseindia.net/jstl/introduction.shtml)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | http://www.roseindia.net/java/images/blue_left_top.gif | **Java Flavors** | http://www.roseindia.net/java/images/blue_right_top.gif | |  | |  |  |  | | --- | --- | --- | | **Java For Beginner** |  | **Master-Java** | | 1. [**New to Java?**](http://www.roseindia.net/java/new-to-java.shtml) If you are new to Java technology and you want to learn Java and make career in the Java technology then this page is for you. 2. [**Use of Java**](http://www.roseindia.net/java/use-java/index.shtml) Java technology is a high-level programming and a platform independent language. Java is designed to work in the distributed environment on the Internet. 3. [**Java Exception**](http://www.roseindia.net/java/java-exception/index.shtml)In this Java exception handling tutorial, you will learn about different types of exception and errors with the solutions to handle it. Basically exceptions in Java are used to handle errors and exceptional events. Go through the given tutorial to find out more on exceptions.. 4. [**Java Get Example**](http://www.roseindia.net/java/java-get-example/index.shtml) An introduction to various methods and functions to get the different values in Java. After going through this section, you will be able to get the date, time, certain values and directory path etc.. 5. [**Pass Value Example**](http://www.roseindia.net/java/pass-value-example/index.shtml) 6. [**Java Break Example**](http://www.roseindia.net/java/java-break-example/index.shtml) 7. [**Java Biginteger**](http://www.roseindia.net/java/java-biginteger/index.shtml) 8. [**Java Bigdecimal**](http://www.roseindia.net/java/java-bigdecimal/index.shtml) 9. [**JavaScript Array**](http://www.roseindia.net/java/javascript-array/index.shtml) 10. [**Java Methods**](http://www.roseindia.net/java/java-methods/index.shtml) 11. [**Java Write**](http://www.roseindia.net/java/java-write/index.shtml) 12. [**Java Programming Introduction**](http://www.roseindia.net/java/java-programming.shtml) Java programming language is useful for development of enterprise grade applications. Java language comes in light from a language named Oak. 13. [**Learn Java In A Day**](http://www.roseindia.net/java/learn-java-in-a-day/index.shtml) Java is the most exciting object oriented programming language in computer programming landscape 14. [**Learn Java Quickly**](http://www.roseindia.net/java/quick-java.shtml) This is one page introduction to java, that will teach you the basics of java. In this quick Java tutorial you will download, install and then create very first java program. 15. [**Java Tools**](http://www.roseindia.net/java/tools/)Java Development Kit or **JDK** comes with lots of tools for compiling and running the programs. These tools also contains programs to test and document the programs. In this Java Tools tutorials we will provide you detailed introduction of these Java Tools. 16. [**Beginners Java Tutorials**](http://www.roseindia.net/java/beginners/index.shtml) Java Tutorial for beginners. 17. [**Java Conversion Tutorials**](http://www.roseindia.net/java/java-conversion)Java conversion tutorials with example. 18. [**Java Comparison**](http://www.roseindia.net/java/java-comparison/index.shtml) 19. [**Java String Examples**](http://www.roseindia.net/java/string-examples) Java String tutorials with example. 20. [**Java Exceptions**](http://www.roseindia.net/java/exceptions)Exceptions are nothing but some anomalous conditions that occur during the execution of the program. Exceptions are the conditions or typically an event which may interrupt the normal flow of the program's instructions. 21. [**Wrapper Class Tutorials and Examples**](http://www.roseindia.net/java/wrapper-class)Wrapper class is a wrapper around a primitive data type. It represents primitive data types in their corresponding class instances e.g. a boolean data type can be represented as a Boolean class instance. 22. [**Reflection API Tutorials and Examples**](http://www.roseindia.net/java/reflect)Reflection API is a powerful technique (that provides the facility) to find-out Its environment as well as to inspect the class itself. Reflection API was included in Java 1.1. 23. [**Java Applet Tutorials**](http://www.roseindia.net/java/example/java/applet/) Introduction to Java Applet and example explaining how to write your first Applet. Java Applet tutorials for beginners, collection many Java Applets example with running code. 24. [**Java iText - Open Source PDF Libraries**](http://www.roseindia.net/java/itext/index.shtml) iText is a java library that provides the facilities for generating PDFs. You can improve the look and feel of your html which is the browser independent by using this java library. You can get more about the iText library inside the topic. 25. [**Java Ftp Libraries**](http://www.roseindia.net/java/javaftp/java-ftp-library.shtml) Here you will get some tutorials related to the java ftp including ftp libraries and some important explained examples. 26. [**Various Commands that are used in java are given below**](http://www.roseindia.net/java/various-commands.shtml)A Java Compiler javac is a computer program or set of programs which translate java source code into java byte code. 27. [**Java ClassPath**](http://www.roseindia.net/java/java-classpath.shtml)Java is an OOP language. In java programming language for compiling the program we use the compiler that converts the source code into the byte code after that, that byte code is interpreted by JVM that converts the bytecode into the machine independent code. 28. [**Task Scheduling in JAVA**](http://www.roseindia.net/java/task-scheduling.shtml) In some applications some task need to run periodically, for example a application of report generating checks for new database entry after one day and make reports according to the entries then save all entries in company's permanent record. |  | 1. [**Master Java Tutorials (TOC)**](http://www.roseindia.net/java/master-java/index.shtml) This tutorial will make you perfect in Java i.e. the master of Java. You can be master of Java by going through the Master Java Tutorial. 2. [**Java as a programming language**](http://www.roseindia.net/java/master-java/java-a-general-purpose-languaage.shtml) Java is an Object oriented application programming language developed by Sun Microsystems. Java is a very powerful general-purpose programming language. 3. [**Java as an Object Oriented Language**](http://www.roseindia.net/java/master-java/java-object-oriented-language.shtml) In this section, we will discuss the OOPs concepts along with  fundamentals  used to develop the java applications and programs. 4. [**JEE 5 Tutorial**](http://www.roseindia.net/java/jee5/) Welcome to JEE 5 tutorial guide, here you will find everything need to know about Java Enterprise Edition 5 for developing best enterprise application based on JEE technologies. 5. [**JDK 6 Tutorial**](http://www.roseindia.net/java/jdk6/index.shtml) The latest news for the java programmers that the Sun MicroSystems has released the Java SE 6 on Monday December 11. So go and grab your copy.  It has been released with the promises to ease the complexities faced in the jdk 1.5. 6. [**Java IO Package Examples**](http://www.roseindia.net/java/example/java/io/) The Java I/O is the collection of java classes and interfaces for reading and writing on to stream, byte stream and array of byte stream. 7. [**Java AWT Package Examples**](http://www.roseindia.net/java/example/java/awt/) In this section you will learn about the AWT package of the Java. Many running examples are provided that will help you master AWT package. 8. [**Java UDP Tutorial**](http://www.roseindia.net/java/example/java/net/udp/) UDP Tutorial with many examples in Java. 9. [**Swing Example**](http://www.roseindia.net/java/example/java/swing/) Here you will find many Java Swing examples with running source code. Source code provide here are fully tested and you can use it in your program. 10. [**Java util Examples**](http://www.roseindia.net/java/example/java/util/) The util package or java provides many utility interfaces and classes for easy manipulation of in-memory data. The java util package tutorials  at RoseIndia.net introduces you with the Java util package of JDK. 11. [**Java Threading Tutorial**](http://www.roseindia.net/java/thread/index.shtml) Threading concept is very important in Java Programming language. A thread is a sequential path of code execution within a program. And each thread has its own local variables, program counter and lifetime. 12. [**Java 5 Tutorials**](http://www.roseindia.net/java5/index.shtml) Java 5.0 tutorials. 13. [**Java-Network Tutorials**](http://www.roseindia.net/java/network/index.shtml) The Java platform is extremely preferable to write an application program require to communicate with the resources on network. 14. [**Simple java program**](http://www.roseindia.net/java/simple-java-program.shtml) Write a simple java program to makes a profit a local store marks up the prices of its items by 25%. 15. [**Java Virtual Machine**](http://www.roseindia.net/java/java-virtual-machine.shtml)JVM is the main component of Java architecture and it is the part of the JRE (Java Runtime Environment) . It provides the cross platform functionality to java. 16. [**Collection**](http://www.roseindia.net/java/collection/index.shtml) 17. [**Java Date**](http://www.roseindia.net/java/javadate/index.shtml) | |  | | http://www.roseindia.net/java/images/blue_left_bottom.gif |  | http://www.roseindia.net/java/images/blue_right_bottom.gif | | | |  |  |  | | --- | --- | --- | | http://www.roseindia.net/java/images/blue_left_top.gif | **Detailed Java Course Material** | http://www.roseindia.net/java/images/blue_right_top.gif | |  | **Section I - Introduction to Java**  This section introduces you the Java programming language. These days Java programming language is being used for programming web  applications. It is also widely used for mobile and electronic items.   1. [**What is Java?**](http://www.roseindia.net/java/java-introduction/what-is-java.shtml) Java is a high-level object-oriented programming language developed by the Sun Microsystems. Though it is associated with the World Wide Web but it is older than the origin of Web. 2. [**Java as an Internet Language**](http://www.roseindia.net/java/java-introduction/java-an-internet-language.shtml) Java is an object oriented language and a very simple language. Because it has no space for complexities. At the initial stages of its development it was called as OAK. OAK was designed for handling set up boxes and devices. 3. [**Java as general purpose language**](http://www.roseindia.net/java/java-introduction/java-general-purpose-language.shtml) Java is an Object oriented application programming language developed by Sun Microsystems.    **Java Features** 4. [**Case sensitivity**](http://www.roseindia.net/java/java-introduction/casesensitivity.shtml)What is case sensitivity:  Case sensitivity is the mechanism in which words can be differ in meaning based on different use of uppercase and lowercase letters. 5. [**Java is Simple and platform Independent**](http://www.roseindia.net/java/java-introduction/java-features.shtml) The concept of Write-once-run-anywhere (known as the Platform independent) is one of the important key feature of java language that makes java as the most powerful language. 6. [**Java Enabled browsers**](http://www.roseindia.net/java/java-introduction/java-enabled-browsers.shtml) Java language is the most powerful language and is widely used in the web application. Today most of the web browser are java compatible.    **Java Tools** 7. [**Java Compiler**](http://www.roseindia.net/java/java-introduction/javatools/java-compiler.shtml) To commence with Java programming, we must know the significance of Java Compiler. 8. [**Java Interpreter**](http://www.roseindia.net/java/java-introduction/javatools/java-inetrpreter.shtml)We can run Java on most platforms provided a platform must has a Java interpreter. 9. [**Java Debugger**](http://www.roseindia.net/java/java-introduction/javatools/java-debugger.shtml) Java debugger helps in finding and the fixing of bugs in Java language programs. 10. [**Java Header File Generator**](http://www.roseindia.net/java/java-introduction/javatools/header-file-generater.shtml) In Java programming we need to implement some native methods. To implement these methods **Javah**generates C header and source files that are used by C programs to reference an Object's instance variables from native source code. 11. [**JavaDoc**](http://www.roseindia.net/java/java-introduction/javatools/javadoc.shtml) This tool is used to generate API documentation into HTML format from Java source code. 12. [**Applet Viewer**](http://www.roseindia.net/java/java-introduction/javatools/java_applet_viewer.shtml) **Applet viewer** is a command line program to run Java applets.   **Section II- Java Language**   1. [**Java Comments**](http://www.roseindia.net/java/language/java-comments.shtml)To comprehend any programming language, there are several kind of comments which are used. 2. [**Java Keywords**](http://www.roseindia.net/java/language/java-keywords.shtml)There are few keywords in Java programming language. Remember, we cannot use these keywords as identifiers in the program. The keywords "const" and "goto" are reserved though, they are not being currently used. 3. [**Java Data Types**](http://www.roseindia.net/java/language/java-data-types.shtml)Java programming language is a language in which all the variables must be declared first and then to be used i.e. to specify the name and the type of the variable. This specifies that Java is a strongly-typed programming language 4. [**Literals**](http://www.roseindia.net/java/language/java-literals.shtml)By literal we mean any number, text, or other information that represents a value. This means what you type is what you get. We will use literals in addition to variables in Java statement. While writing a source code as a character sequence, we can specify any value as a literal such as an integer. 5. [**Arrays**](http://www.roseindia.net/java/language/introduction-to-java-arrays.shtml)In this section you will be introduced to the concept of Arrays in Java Programming language. You will learn how the Array class in java  helps the programmer to organize the same type of data into easily manageable format. 6. [**Operators**](http://www.roseindia.net/java/language/operators.shtml)Operators are symbols that performs some operations on one or more then one operands. Once we declare and initialize variables, we can use operators to perform certain tasks like addition, subtraction etc. 7. [**Controlling your program**](http://www.roseindia.net/java/language/controllingyourprogram.shtml)We all know that the execution of the statements in a program takes place from top to bottom. We will learn how the different kinds of statement have different effects in looping like decision-making statements (if-then, if-then-else, switch), the looping statements (for, while, do-while), and the branching statements (break, continue, return) in Java programming language. 8. [**Java Classes**](http://www.roseindia.net/java/language/classes-java.shtml)In this section you will be introduced to some essential classes in Java like Exceptions, Basic I/O, Concurrency etc. 9. [**Class Inheritance**](http://www.roseindia.net/java/language/inheritance.shtml)To know the concept of inheritance clearly you must have the idea of class and its features like methods, data members, access controls, constructors, keywords this, super etc. 10. [**Summary**](http://www.roseindia.net/java/language/summary.shtml)You have been introduced to the contents of Java language providing a great amount of information. Perhaps it would be a bit arduous to grasp the whole lot of things at one go. However, the implementation of these little basics will make you a substantial programmer.   **Related Java Tutorials/Articles**   1. [**Java Tutorial and articles**](http://www.roseindia.net/javatutorials/) **Here you will find articles related to the Java Programming Language. Some of the articles are:** [**Deadlocks in Java**](http://www.roseindia.net/javatutorials/deadlocksinjava.shtml)**,**[**Java GC**](http://www.roseindia.net/javatutorials/javagc.shtml)**,**[**Java Compile Time Constants**](http://www.roseindia.net/javatutorials/javacompiletimeconstants.shtml)**,**[**Enum Inversion Problem**](http://www.roseindia.net/javatutorials/enum_inversion_problem.shtml)**,**[**Head First Design Pattern**](http://www.roseindia.net/javatutorials/headfirsdesignpatterns.shtml)**,**[**LinkedList vs. ArrayList**](http://www.roseindia.net/javatutorials/linkedlistvsarraylist.shtml)**,**[**Java Break to Label Statement**](http://www.roseindia.net/javatutorials/java_break_to_label_tatement.shtml)**,**[**Strategy Pattern of Hascode Equality**](http://www.roseindia.net/javatutorials/strategy_pattern_of_hashcode_equality.shtml)**,**[**Object Adapter based on Dynamic Proxy**](http://www.roseindia.net/javatutorials/object_adapter_based_on_dynamic_proxy.shtml)**,**[**Making Enumerations Iterable**](http://www.roseindia.net/javatutorials/making_enumerations_iterable.shtml)**,**[**JTable in JDK**](http://www.roseindia.net/javatutorials/JTable_in_JDK.shtml)**,**[**JDK 1.5 Performance Surprises**](http://www.roseindia.net/javatutorials/JDK_1_5%20_performance_surprises.shtml)**...**[**Click Here to View all tutorials**](http://www.roseindia.net/javatutorials/) 2. [**Java Tutorials - Complete Java Tutorial for Beginners and Advance Programmers**](http://www.roseindia.net/javajdktutorials/)    * Chapter 1: [**Overview: The Mental Landscape**](http://www.roseindia.net/javajdktutorials/c1/index.shtml)    * Chapter 2: [**Programming in the Small I: Names and Things**](http://www.roseindia.net/javajdktutorials/c2/index.shtml)    * Chapter 3: [**Programming in the Small II: Control**](http://www.roseindia.net/javajdktutorials/c3/index.shtml)    * Chapter 4: [**Programming in the Large I: Subroutines**](http://www.roseindia.net/javajdktutorials/c4/index.shtml)    * Chapter 5: [**Programming in the Large II: Objects and Classes**](http://www.roseindia.net/javajdktutorials/c5/index.shtml)    * Chapter 6: [**Applets, HTML, and GUI's**](http://www.roseindia.net/javajdktutorials/c6/index.shtml)    * Chapter 7: [**Advanced GUI Programming**](http://www.roseindia.net/javajdktutorials/c7/index.shtml)    * Chapter 8: [**Arrays**](http://www.roseindia.net/javajdktutorials/c8/index.shtml)    * Chapter 9: [**Correctness and Robustness**](http://www.roseindia.net/javajdktutorials/c9/index.shtml)    * Chapter 10: [**Advanced Input/Output**](http://www.roseindia.net/javajdktutorials/c10/index.shtml)    * Chapter 11: [**Linked Data Structures and Recursion**](http://www.roseindia.net/javajdktutorials/c11/index.shtml)    * Chapter 12: [**Generic Programming and Collection Classes**](http://www.roseindia.net/javajdktutorials/c12/index.shtml)    * Appendix 1: [**Other Features of Java**](http://www.roseindia.net/javajdktutorials/advanced.shtml)    * Appendix 2: [**Some Notes on Java Programming Environments**](http://www.roseindia.net/javajdktutorials/progenv.shtml)    * Appendix 3: [**Source Code for All Examples in this Book**](http://www.roseindia.net/javajdktutorials/source/index.html) 3. [**Java Security Packages JCA/JCE**](http://www.roseindia.net/java/java-security.shtml) In this tutorial, the author explains the cryptography-related concepts and packages in JDK, with code examples. Many of the concepts and technical terms thus learnt will be useful in understanding the Cryptography API in MFC also.   [**Browser All Tutorials**](http://www.roseindia.net/javajdktutorials/contents.shtml)  **Latest**[**Java News**](http://www.roseindia.net/java/javanews/index.shtml) |  | | |

# New to Java?

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/index.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/index.shtml)**

If you are new to Java

technology and you want to learn Java and make career in the Java technology then this page is for you. Here we have explained how to learn Java and master the Java technologies.  
  
Java is a vast language and it requires a lot of effort to learn and master the necessary Java technology to start your real life projects.  
  
Here at RoseIndia.Net, we have developed hundreds of tutorials, examples and articles to help you learn Java quickly and easily. We have tried to put support examples related to each Java technology that will help you master the concepts.  
  
These tutorials and examples are arranged in a sequence, so that you can learn Java step by step and master the Java and JEE technologies.   
  
**Basics of Java Technology**  
This section provides an overview of Java technology as programming language and a platform. Java technology is a simple, secure, robust, complete object oriented and platform independent high level programming language.  It is also portable, high performance, multithreaded and networksavy that enable it in constructing software that can run along in small machines.  The whole technology is based on the concept of *Java Virtual Machine* (JVM) that acts as a translators of  byte code into machine language. In other words JVM converts the java byte codes into platform specific machine language.

**Understanding the Java Technology**  
Java technology is high-level, object-oriented, very robust programming language. Java is platform independent programming language and you can run your compiled code on any operating system without recompiling your source code. Java technology is based on the concept of a single Java virtual machine (JVM) -- a translator between the language and the underlying software and hardware. All implementations of the programming language must emulate the JVM, enabling Java programs to run on any system that has a version of the JVM.

If you want to start java programming then you need to use a text editor to create and edit the source code.  By using the Java complier, you can change the source code into byte code. The byte code can be run on any platform having Java interpreter that can convert the byte code into codes suitable for the operating system.   
  
**Why Java Technology is so important?**  
This high-level powerful programming language provides a powerful software platform as the JVM installed on different platform understand the same byte code. This is ideal for server side web programming and runs in a secured manner over internet. It enhance the computing power of the users by taking merely from desktop to the resource of the web. It contains JVM and Java Application Programming Interface (API) that are kinds of readymade software components, and for using any component, the need is just to import a related package in your program use the functionality. It offers development tools that can be used in compiling, running, debugging and documenting the application, making the Java programming fun and easy. The automatic garbage collection mechanism helps in avoiding memory leaks and its coding takes less development time than other programming languages like C++.    
  
**Different Editions of Java Technology**  
  
a) **Java SE**- Java SE or Java Standard Edition provides tools and API's that you can use to create server applications, desktop applications, and even applets. These programs developed using Java SE can be run on almost every popular operating system, including Linux, Macintosh, Solaris, and Windows.  
   
b) **JEE** - Based on the foundation framework of the standard edition, Java Enterprise Edition helps in web application service, component model and enterprise class service oriented architecture (SOA).   
  
c) **JME** - Java Micro Edition or JME for short is an accumulation of Java APIs that are used for the development of software for devices like mobile phones, PDAs, TV set-top boxes, game programming. The platform of micro edition generally consists of an easy user interface, a robust security model and a wide variety of built-in networks for running Java based application.   
  
**Components of each edition**  
  
**JSE Components**  
**JavaBeans** - It is the component architecture for J2SE platform and one can develop and assemble these software programs for better web application. It is a reusable software component that can be manipulated visually in a builder tool. This software assists visual builder tools in using reflection, introspection, and also analyze and customize JavaBeans.

**Java Foundation Classes (JFC)** - It is a part of Java class libraries based on the Java platform used for developing graphical user interface (GUI). JFC helps in 2D graphics, imaging, text formatting and printing with the help of Abstract Window Toolkit (AWT), Swing and Java2D. With the help of input method framework, the JFC technology assists in preparing application that can be accessible to all users around the world in different languages. Drag and Drop is another feature of JFC that supports data transfers between different Java applications.

**JavaHelp** - It is a platform independent and a feature oriented software system that offers developer an automated help component. JavaHelp 2.0 API is useful while building online documentation and presenting online information to the application users.

**Java Web Start** - It is framework in the Java platform that assists in starting Application software directly from the internet by using a web browser. As we know Java applet can run in a browser but in case of Java Web Start, it doesn't run inside and solve many complex problems associated with Java*plugins* and JVM. It also provide many classes that in turn provides various services and allow better access to resources. Version 1.0 was introduced in the year 2001. Now with the release of J2SE 1.4, Java Web Start is included with Java Runtime Environment and does need any separate installation.   
  
**Java Database Connectivity (JDBC)** - JDBC API is a part of Java Standard Edition that helps in accessing data from a SQL based database. Besides, it also process the result and allows in using the programming language with "Write Once, Run Anywhere" feature. Some of its key features are like full access to metadata, no special installation and database identification.

**Java Media Framework (JMF)** - Its an advanced API that allows Java developers to process and add audio-video source to Java application and applets. It is useful for multimedia developers to capture, playback, transcode different media formats.    
   
**JEE - Components:**  
  
**Enterprise JavaBeans (EJB) -**This technology is a server side component of Java platform  used for the construction of enterprise application. It is one of the Java APIs attached with the enterprise edition. By using Java technology, EJB helps in quick development of small, distributed, transactional and secure application.

**JavaMail**- This JavaMail API technology allows to build mails and messaging application in a platform independent and protocol independent framework. It is both a part of JSE and JEE platform. Thus, JavaMail uses an extensible platform for transferring all kinds of Multimedia Internet Mail Extension (MIME).

**Java Message Service (JMS) -**Developed under Java community process, JMS technology is used for sending messages between users. Basically, it  is an enterprising messaging tool used for building enterprising application. The JMS API is a combination of Java technology and enterprising messaging that provides facilities for building small message based application. It functions under two models:*Point-to-Point*and *Publishing & Subscribing* model.   
  
**JavaServer Pages (JSP)** - The JSP technology enables web developers in developing and maintaining web content pages in formats like HTML and XML. With the help of JSP, it becomes very easy to build server and platform independent web based application. This uses HTML and XML tags that offers logical solution for the content. This separately user interface and content development from each other, which allows the designer to change page layout without changing the content.

**Java Servlets -**This enables a developer in adding content to a web server by using Java platform. This provides the mechanism for enhancing the functionality of web server. In short, servlets provides platform independent and component based web based application without the performance limiting of CGI program.   
  
**JME - Components:**  
**Connected Limited Device Configuration (CLDC) -**It is one of the configurations of Java Micro Edition. 'Configuration' describes minimal features of a complete Java. The CLDC specifies the capabilities of JVM, the base set of API for resource limited devices like pager and mobile phones. There are two version of CLDC: version 1.0 was released in 2000 and came to be known as Java Specification Request (JSR)30. Later version 1.1 or JSR 139 but 1.0 is more widely used. The Connected Limited Device Configuration and the Mobile Information Device Profile (MIDP) together provides solid Java platform for developing application to run on less processing power devices.   
  
**Mobile Information Device Profile (MIDP)**- This is another configuration of Java Micro Edition and coupled with CLDC, it provides a farm Java Runtime Environment for various mobile devices and other personal digital assistance (PDA). With the help of MIDP, developers can develop application once and then redistribute them into various mobile information devices in a very small period of time. Its principal functions include the user interface, network connectivity data storage and overall application process management. There are two versions of MIDP: one is MIDP 2.0 or JSR 118 and the second one is the MIDP 1.0 or JSR 37.    
  
**Connected Device Configuration (CDC) -** Developed under the Java Community Process (JCP), it is a standard framework of Java technology used for building and delivering application that can be shared in a wide range of networks and devices ranging from pagers, mobile phones, set top box and other PDA devices. It is in two versions: the JSR 36 (CDC 1.0) and the latest one is the JSR 218 (CDC 1.1).

**What should be my learning path:**

1. [**Learn Core Java**](http://roseindia.net/java) - The Core Java Technology is the foundation of Java Platform of JSE. It is used in all classes of Java programming from desktop to Java Enterprise Edition. This include Java APIs, Java Application, JVM, JavaBeans, JavaScript, JSP etc.
2. [**Learn JSP**](http://www.roseindia.net/jsp/jsp.shtml) - JSP technology assists developers in generating HTML, XML web pages. It uses Java code and some predefined actions while creating web content. This helps in the creation of JSP tag libraries that acts as extensions to HTML and XML tags.
3. [**Learn Servlets**](http://www.roseindia.net/servlets/servlets.shtml) - In a Java Platform, Servlets assists developers in adding content to a web server. Servlets with Java server pages acts as a competitor to various dynamic web content technologies like CGI, ASP.NET, JavaScript etc.
4. **Learn about Tomcat and other servers** - Developed by Apache Software Foundation Tomcat is a Java based web application server used to run Servlet and JSP. It is not merely limited to application server and provides an open platform to develop extensible web and content management service.
5. **Learn Open Source technologies** ([**Struts**](http://www.roseindia.net/struts/), [**Hibernate**](http://www.roseindia.net/hibernate/), [**Spring**](http://www.roseindia.net/spring/)) - Basically in Java there are three open source technologies known as frameworks; these are Spring, Hibernate and Struts. These open source application frameworks solves many problems related to JSE and J2EE, and helps in effective development of web application.
6. [**Learn EJB**](http://www.roseindia.net/java/jee5/) - Enterprise Java Beans are a part of J2EE and also a server sided component used mostly in large projects. It helps in easy and rapid development of distributed, transactional and small application based on Java technology.
7. **Learn about Database Management System** - It is a software designed to manage and run a database. Generally, it is used in company back office work, accounting, customer support system and several other purposes.

# What is Java, it’s history?

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/what-is-programming.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/where-java.shtml)**

Java is a high-level object-oriented programming language developed by the Sun Microsystems. Though it is associated with the World Wide Web but it is older than the origin of Web. It was only developed keeping in mind the consumer electronics and communication equipments. It came into existence as a part of web application, web services and a platform independent programming language in the 1990s.

Earlier, C++ was widely used to write object oriented programming languages, however, it was not a platform independent and needed to be recompiled for each different CPUs. A team of Sun Microsystems including Patrick Naughton, Mike Sheridan in the guidance of James Goslings decided to develop an advanced programming language for the betterment of consumer electronic devices. They wanted to make it new software based on the power of networks that can run on different application areas, such as computers and electronic devices. In the year 1991 they make platform independent software and named it Oak. But later due to some patent conflicts, it was renamed as Java and in 1995 the Java 1.0 was officially released to the world.

Java is influenced by C, C++, Smalltalk and borrowed some advanced features from some other languages. The company promoted this software product with a slogan named “Write Once Run Anywhere” that means it can develop and run on any device equipped with Java Virtual Machine (JVM). This language is applicable in all kinds of operating systems including Linux, Windows, Solaris, and HP-UX etc.

# Where is Java being Used?

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/what-is-java.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/java-releases.shtml)**

The programming language Java was developed by Sun Microsystems in the year 1995. Earlier, it was only used to design and program small computing devices but later adopted as one of the platform independent programming language. The most important feature of Java is its byte code that can be interpreted on any platform including windows, Linux etc. One can also download it freely from the official website of Sun.   
  
As we have mentioned above that java-programming language was only developed for the small devices but now it can be found in a variety of devices like cell phones, e-commerce application, PCs and almost all network or computing devices.   
  
Java is available in different form:  
**JSP**? Like PHP and ASP, Java Server Pages based on a code with normal HTML tags, which helps in creating dynamic web pages.   
  
**Java Applets** ? This is another type of Java program that used within a web page to add many new features to a web browser. These are small program used in the programming of instant messaging, chat service, solving some complex calculation and for many other purposes.   
  
**J2EE** ? The software Java 2 Enterprise Edition are used by various companies to transfer data based on XML structured documents between one another.   
  
**JavaBeans** ? This is something like Visual Basic and a reusable software component that can be easily assemble to create some new and advanced application.  
  
As far as syntax is concerned, Java is similar as the C programming language but a distinct style of coding. It follows all the general programming features like loops, data types, conditions, curly braces, semi-colon etc. Its a fully featured Object Oriented Programming (OOP) language as it supports all OOP features including classes, modules, inheritance, Polymorphism etc.    
  
**Mobile Java -**Besides the above technology, Java is also used for various entertainment devices especially mobile phone. Mobile Information Devices Profile (MIDP) uses Java run time environment in cell phones, mobile tracking systems and other traditional PDA devices. Java technology enabled application is key to the games and services available in the mobile world. This also plays an important role in the field of telemedicine such as PulseMeter. As far as mobile technology is concerned, it offers offline facility, so that users can get service even if they face loss of connection. Today, all leading mobile service provider like Nokia, Siemens, Vodafone are using Java technology. Sun Java Wireless Toolkit offers complete support for developing different MIDP application.

Java technology is enabled with healthy content ecosystem by offering a healthy development and deployment environment, protecting users and operators from down time and viruses. The increase volume of users now encouraging manufactures and developers to apply Java technology in numerous other productive and functional ways including MP3 players, digital TV, video, 3D, simplifying games, etc.

# Java Releases

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/where-java.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/download_jdk.shtml)**

Java is developed by Sun Microsystems in 1996. Now the Java is powerful tools for the development of robust and scalable enterprise applications.

The tables given below provide information about the name and time of releases for different versions of Java technology.

|  |  |  |
| --- | --- | --- |
| **Releases Event** | **Month** | **Year** |
| **Java 1.5.0\_09** | **October** | **2006** |
| **Java 1.5.0\_08** | **August** | **2006** |
| **Java 1.5.0\_06** | **December** | **2005** |
| **Java 1.5.0\_05** | **October** | **2005** |
| **Java 1\_5\_0\_04** | **July** | **2005** |
| **Java 1\_5\_0\_01** | **February** | **2005** |
| **Java 2 Platform, Standard Edition 1.4 (J2SE 1.4)** | **February** | **2002** |
| **Java 2 Platform, Standard Edition 1.3 (J2SE 1.3)** | **May** | **2000** |
| **Java 2 Platform, Enterprise Edition (J2EE)** | **December** | **1999** |
| **Java 2 Platform, Standard Edition (J2SE)** | **August** | **1999** |
| **Java Development Kit 1.2 (JDK 1.2)** | **December** | **1998** |
| **Java Development Kit 1.1 (JDK 1.1)** | **February** | **1997** |
| **Java Development Kit 1.0 (JDK 1.0)** | **January** | **1996** |

# Download JDK

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/java-releases.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/installing_java.shtml)**

What is JDK (Java Development Kit)

JDK is a software development program provided by sun Microsystems. Java Development Kit or JDK comes in various version and can be downloaded free from the sun Microsystems. JVM compiler, debugger and  other tools are used with JDK for developing java based application & java applets. So make sure that your JVM compiler & JDK versions are same.

JDK also known as Java 2 Platform, That comes in three editions J2ME, J2SE & J2EE. If you are beginner or learning Java then start by downloading J2SE.

**Acronyms:**  
**JDK**  Java Development Kit  
**JVM**  Java virtual machine

**Download JDK  
You can download JDK from**[**www.javasoft.com/j2se**](NULL)

**Latest version of JDK**

1. **JDK 5.0 Update 6**  
   The full internal version number for this update release is 1.5.0\_06-b05 (where "b" means "build"). The external version number is 5.0u6. Java Version 1.5.0\_06 introduces various security enhancements in Java Plug-in and Java Web Start to better protect users and enterprises. For more information please visit:[**http://java.sun.com/j2se/1.5.0/ReleaseNotes.html**](http://java.sun.com/j2se/1.5.0/ReleaseNotes.html)

# Beginners Java Tutorials - Installing JDK

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/download_jdk.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/sdk-directory-structure.shtml)**

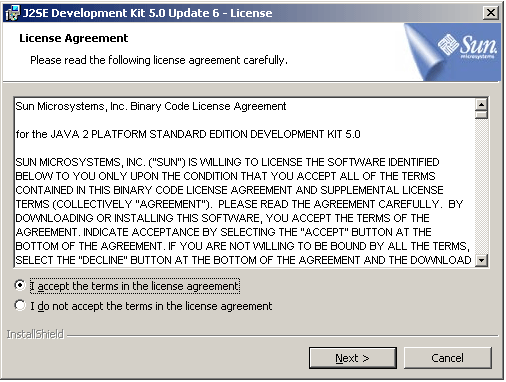
In this beginners java Tutorial ,We will first download, install and configure the J2SE development environment.

**Downloading and Installing J2SE Software on Windows Platform**  
To download J2SE for development visit [**http://www.java.sun.com/j2se**](http://www.java.sun.com/j2se) and download J2SE on your machine. In this tutorial we have used jdk-1\_5\_0\_06-windows-i586.exe.The java 2Platform or (JDK) can be downloaded from the sun. Formerly Known as the java Development kit ,or JDK, Downloading java is really about downloading the java 2 plat form that comes in three editions , J2ME, J2SE and J2EE , if you are learning  java, then, you should start by downloading  J2EE.

Once you have downloaded the j2se on your system, you are ready to install . In the following section we will learn how to install  jdk development environment on your machine. here are the step to install JDK on your windows machine.

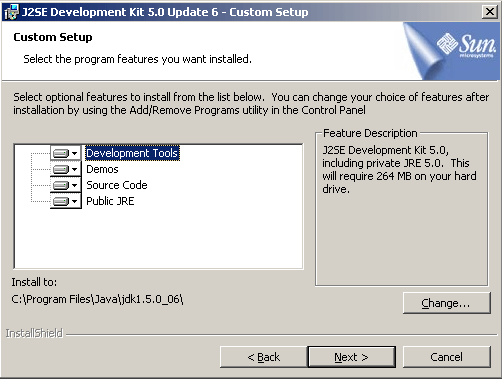
**Step 1**

Double click the JDK down loaded file, the executable extracts the required Contents to the temporary directory and then License agreement screen  appears. On the license agreement page read and accept the license and the click the next button .



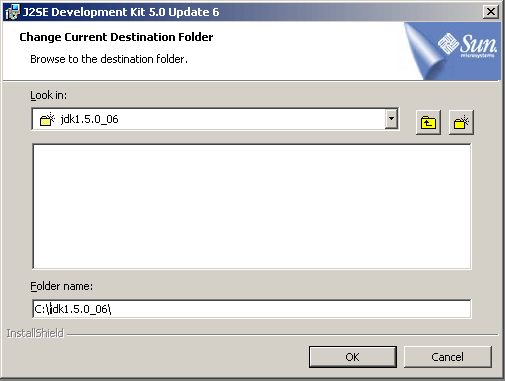
**Step 2**

The custom setup screen appears as follows.

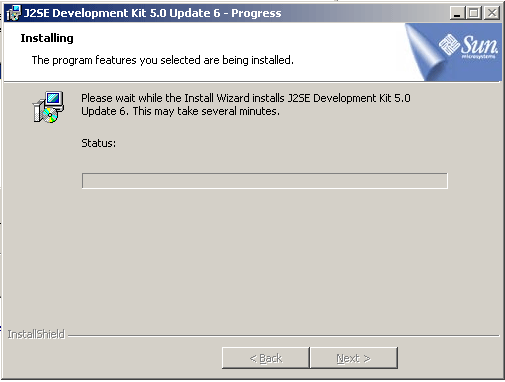


**Step 3**

Click on the change button to change the installation directory to "**c:\jdk1.5.0\_06**" as shown in the following screen.

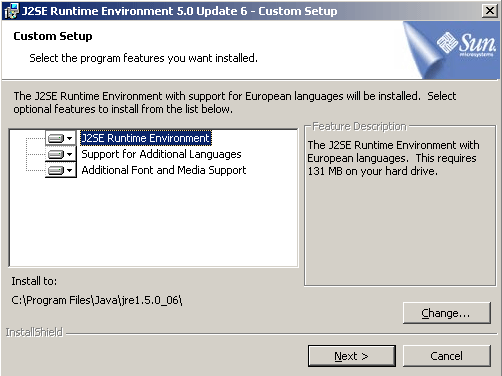


and click on the "OK" button. After clicking on the "OK" button installation begins:



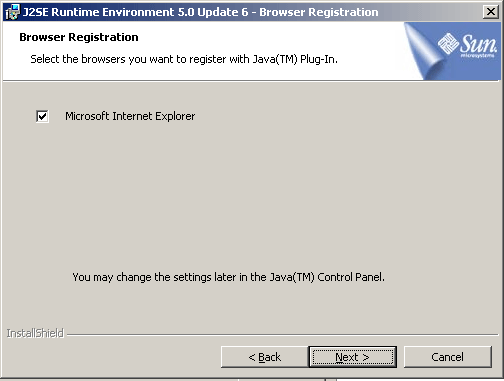
**Step 4**

In the next window installer asks for the installing the runtime as shown in the following screen:



**Step 5**

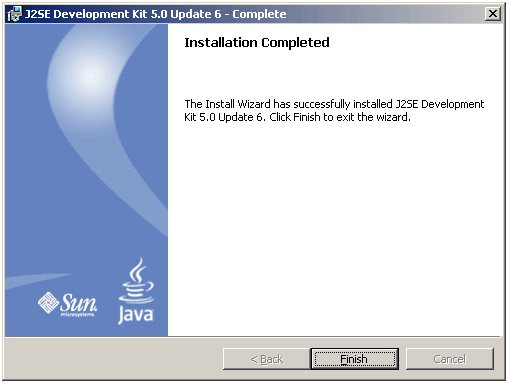
Click on next button install the J2SE runtime on your machine. Next screen shows the browser selection:



Click on the "Next" button.

**Step 6**

Once the installation is finished it shows you the final screen indications the success. Now you have successfully installed J2SE on your machine. Installer shows the following final confirmation window as shown below:

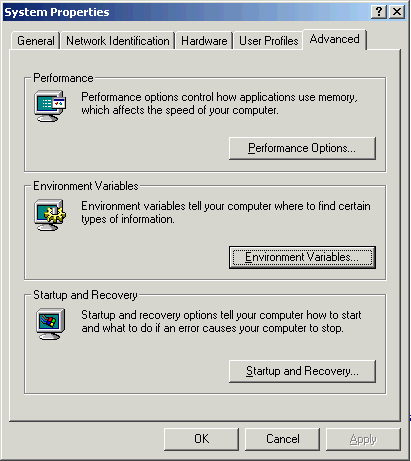


Click on the "Finish" button to exit from the installer.

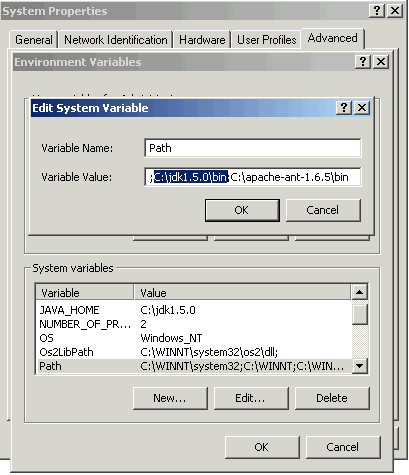
**Configuring the installation on windows machine:**

In this Section we will add some settings to the windows environment so that the java compiler and runtime becomes available for compiling and running the java application.

Go to the control panel and double click on "System Properties" and to to the advance tab.



and add "**c:\jdk1.5.0\_06**" to path variable:



and click on ok button. To save the setting click on "OK" button.

This will make the java environment available for development. Open the dos prompt and type **javac** on the console, it should show the following output:

|  |
| --- |
| Microsoft Windows 2000 [Version 5.00.2195] (C) Copyright 1985-2000 Microsoft Corp.  C:\Documents and Settings\Administrator>java Usage: java [-options] class [args...] (to execute a class) or java [-options] -jar jarfile [args...] (to execute a jar file)  where options include: -client to select the "client" VM -server to select the "server" VM -hotspot is a synonym for the "client" VM [deprecated] The default VM is client.  -cp <class search path of directories and zip/jar files> -classpath <class search path of directories and zip/jar files> A ; separated list of directories, JAR archives, and ZIP archives to search for class files. -D<name>=<value> set a system property -verbose[:class|gc|jni] enable verbose output -version print product version and exit -version:<value> require the specified version to run -showversion print product version and continue -jre-restrict-search | -jre-no-restrict-search include/exclude user private JREs in the version search -? -help print this help message -X print help on non-standard options -ea[:<packagename>...|:<classname>] -enableassertions[:<packagename>...|:<classname>] enable assertions -da[:<packagename>...|:<classname>] -disableassertions[:<packagename>...|:<classname>] disable assertions -esa | -enablesystemassertions enable system assertions -dsa | -disablesystemassertions disable system assertions -agentlib:<libname>[=<options>] load native agent library <libname>, e.g. -agentlib:hprof see also, -agentlib:jdwp=help and -agentlib:hprof=help -agentpath:<pathname>[=<options>] load native agent library by full pathname -javaagent:<jarpath>[=<options>] load Java programming language agent, see java.lang.instrument   C:\Documents and Settings\Administrator> |

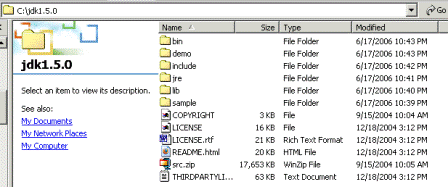
Now your development environment is ready for development.

# Java SDK Directory Structure

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/installing_java.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/hello_world.shtml)**

This section introduces the Directory and file structure of **SDK** (Software Development Kit). The following figure shows jdk1.5.0 software stored in 'c' directory that has multiple subdirectories like bin, demo, include and jre etc. to hold all important developing and testing tools. See detail information bellow.

**SDK Directory Structure:**



**Subdirectories of the SDK:**

***The jdk1.5.0 has following directory:***

**Bin directory** - The bin directory provides all inessential tools for developing and testing the program through the help of command provided by Java compiler.

**Demo directory -**This directory consists many applications and applets with source code.

**Include directory**- It contains all header files like for 'C' programming language that enables you to combine C code into a Java program.

**Jre directory** **-**When you run any java program then you have to compile it by the help of  Java interpreter or Java Runtime Environment (JRE). The SDK uses the internal adaptation of JRE, which containing in the jre root directory.

**Lib directory**- This is a most important directory for development tools that contains libraries and it's supported files.

**Docs directory**- It is the last directory of Software Development Kit that assists you to store the Java documents. The docs directory is an optional directory.

**Files of the SDK:**

***JDK1.5.0 directory has following files that provide the detail information about it.***

**README.html :**The jdk1.5.0 directory provides an html file that contains the detail information of SDK shown on the web browser. This file contains all system requirements, features and documentation links to represents all information's about the jdk1.5.0.

**scr.zip:**The zip file is a collection of one of more files that has been compressed or stored to '.zip' extension. Similarly, the scr.sip file contains the source code of SDK. It become necessary to extract the file that contain the source code from the ZIP file.

# Hello world (First java program)

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/sdk-directory-structure.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/understanding_hello_world_java.shtml)**

Java is a high level programming language and it is used to develop the robust application. Java application program is platform independent and can be run on any operating System. Writing Hello World program is very simple. To write the Hello world program you need  simple text editor like note pad and  jdk must be install in your machine for compiling and running. Hello world program is the first step of java programming  language. Be careful when you write the java code in your text pad because java is a case sensitive programming language.

**For Example**

**hello world   !=(not equal to) Hello World**

Write the following code into your note pad to run the Hello World program .

|  |
| --- |
| class HelloWorld {    public static void main(String[] args){    System.out.println("Hello World!");    } } |

Save the file and Please remember the location where you save your file. In this example we have saved the file in the "**C:\javatutorial\example**" directory. The file name should be match the class name and to save the file give the .java extension. e.g. **HelloWorld.java**

Now open the command prompt to compile the HelloWorld.java program. Go to the directory where you have saved the file ( in our case C:\javatutorial\example>) and issue the following command to compile the program:

**C:\javatutorial\example>javac HelloWorld.java**

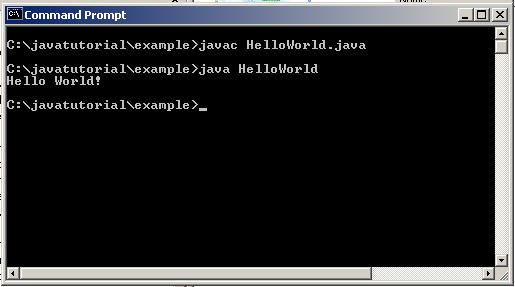
**javac** is a compiler in the java Language. Java compiler change the programming Language into machinery language. So that the java virtual can understand it. Once the compilation is successful a new file will be created with the name **HelloWorld.class.** To run the program issue the following command on command prompt:

**C:\javatutorial\example>java HelloWorld**

You will see the following result on the command prompt.

**Hello World!**

Here is the screen shot of the above steps:



In this lesson you have learned how to write and then test the Hello World! java program.

# Understanding Hello World Java Program

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/hello_world.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Comparing.shtml)**

Now you are familiar with the Java program. In the last lesson you learned how to compile and run the Java program. Before start hard programming in Java, its necessary to understand each and every part of the program. Lets understand the meaning of  public, class, main, String[] args, System.out, and so on.

|  |
| --- |
| public class HelloWorld {    public static void main(String[] args) {    System.out.println("Hello, World");    }    } |

**Class Declaration:**

Class is the building block in Java, each and every methods & variable exists within the class or object. (instance of program is called object ). The *public* word specifies the accessibility of the class. The visibility of the class or function can be public, private, etc. The following code declares a new class "HelloWorld" with the public accessibility:

public class HelloWorld {

The main Method:

The main method is the entry point in the Java program and java program can't run without main method. JVM calls the main method of the class. This method is always first thing that is executed in a java program. Here is the main method:

public static void main(String[] args) {

......  
.....

}

{ and is used to start the beginning of main block and } ends the main block. Every thing in the main block is executed by the JVM.

The code:

System.out.println("Hello, World");

prints the "Hello World" on the console. The above line calls the *println* method of *System.out* class.

**The keyword static:**

The keyword **static** indicates that the method is a *class* method, which can be called without the requirement to instantiate an object of the class. This is used by the Java interpreter to launch the program by invoking the **main** method of the class identified in the command to start the program.

# Comparing Two Numbers

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/understanding_hello_world_java.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/largernumber.shtml)**

This is a very simple example of Java that teaches you the method of comparing two numbers and finding out the greater one. First of all, name a class "Comparing" and take two numbers in this class. Here we have taken a=24 and b=25, now we have to find out whether a=b, a>b or b>a. To find out this apply if and else condition one by one. Now apply the condition "if (a=b)", if this satisfies then type that both are equal in the system class. If this doesn't satisfy, then check whether a>b by applying the "else if" condition and type the message "a is greater than b" in the system class. Again this doesn't satisfy then 'else' condition as shown in the example will show that b is greater than a.

Now compile and run the program and you will find the desired output. If you are getting any error then check the whole program thoroughly and surely you will get correct result. By compiling and running this exact program, you will find that b is greater than a.

|  |
| --- |
| **class**Comparing{   **public static void**main(String[] args) {   **int**a=24, b=25;   **if**(a == b){   System.out.println("Both are equal");   }   **else if**(a>b){   System.out.println("a is greater than b");   }   **else**{   System.out.println("b is greater than a");   }   } } |

# Determining the largest number

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Comparing.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/AllEvenNum.shtml)**

This example of Java programming will teach you the coding for determining the largest number amongst three. Here we have taken three integers as x = 500, y = 70 and z = 3000. After defining these three integers under the class "largernumber" apply "if" and "else" conditions that can help you in finding the largest value one by one.

First check if "x>y". If this satisfies then check whether x>z or not. Again if this satisfies then write in the system class that "x is greater". Again the term "else" comes when "x" is not greater than "z". So check again, if "z" is greater than "y" or not. If this satisfies then type in the system class as "z is greater" otherwise (in the else condition) "y" is greater. Now check whether "y" is greater than "z" or not.

 If "x" is not greater than "y" as per the first condition, then the condition "else" comes and now you have to check if "y>z" or not. If this satisfies then the output comes as "y is greater".

Don't get confuse and analyze every condition one by one and follow this example.

**Here is the code of program:**

|  |
| --- |
| **class**largernumber{   **public static void**main(String[] args) {   **int**x=500, y=70, z=3000;   **if**(x>y){   **if**(x>z){   System.out.println("x is greater");   }   **else**{   **if**(z>y){   System.out.println("z is greater");    }   **else**{   System.out.println("y is greater");   }   }   }   **else**{   **if**(y>z){   System.out.println("y is greater");   }   }   } } |

**Write a program to list all even numbers between two numbers**Posted on: June 2, 2007 at 12:00 AM

**Here you will learn to write a program for listing out all the even numbers between two numbers.**

# Write a program to list all even numbers between two numbers

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/largernumber.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/CircleArea.shtml)**

**Java Even Numbers - Even Numbers Example in Java:**  
Here you will learn to write a program for listing out all the even numbers between two numbers. For this first create a class named **AllEvenNum**under the java.io package. Now use the try/catch exception to avoid any kind of input error. After this create a buffer class in which all the input data are stored and modified. Then give message as to "Enter number" in the System method.

As we have to find out all the even numbers between 1 and the input number, define an integer variable 'num'. Now apply *ParseInt* method that parses the string character into decimal integer. Again apply ***for***loop in which define an integer i=1 and i<=  num also with an increment operator. Then apply the **if**condition that i/2=0 i.e. to find even numbers which are divided by the integer 2. In the end apply the catch exception.

Now and compile and run the program, and enter your desired number to get all even numbers between 1 and this numbers.

**Here is the code of the program:**

|  |
| --- |
| **import**java.io.\*;  **class**AllEvenNum{   **public static void**main(String[] args) {   **try**{   BufferedReader br1 = **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.println("Enter number : ");   **int**num = Integer.parseInt(br1.readLine());   System.out.println("Even Numbers:");   **for**(**int**i=1;i <=num ; i++){   **if**(i%2==0 ){   System.out.print(i+",");   }   }   }   **catch**(Exception e){}      } } |

**[Download this example.](http://www.roseindia.net/java/beginners/AllEvenNum.java)**

# Write a program to calculate area and perimeter of a circle

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/AllEvenNum.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Factorial.shtml)**

The given example will teach you the method for preparing a program to**calculate the area and perimeter of a circle**. First of all name a class as "CircleArea" under Java I/O package and define and integer r=o, which is the radius of the circle. Now use *try* exception to handle errors and other exceptional events. As we have to input the value of radius here create a buffered class with an object as 'br1'. This create a buffering character input stream that uses a default sized input buffer. The InputStreamReader here works as a translator that converts byte stream to character stream. Now type message that "Enter radius of circle" in the System.out.println method.

Now use the parseInt() method of the Integer class in order to convert from external numeric format to internal format. Now create the Math class in which all the mathematical functions are defined. This Math class can be imported from the java.lang.\* package. Write the program for both the cases: radius and perimeter.

Before ending the program use the *Catch* mechanism that detects and catch user input errors. In the end compile and run the program and enter your desired value as radius for calculating the radius and perimeter of the circle.

**Here is the code of the program:**

|  |
| --- |
| **import**java.io.\*; **class**CircleArea{   **public static void**main(String[] args){   **int**r=0;  **try**{   BufferedReader br1 = **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.println("Enter Radius of Circle  : ");   r = Integer.parseInt(br1.readLine());   **double**area = java.lang.Math.PI\*r\*r;   System.out.println("Area of Circle : "+area);   **double**perimeter =2\*java.lang.Math.PI\*r ;   System.out.println("Perimeter of Circle : "+perimeter);   }   **catch**(Exception e){   System.out.println("Error : "+e);   }     } } |

[**Download this example.**](http://www.roseindia.net/java/beginners/CircleArea.java)

# Palindrome Number Example in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Factorial.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/RecArea.shtml)**

In this section, you will learn about the palindrome number and how to determine any number is palindrome or not. First of all we are going to read about the palindrome number.  This is the number that the actual number and after reversing this, in both cases the number is same that is called palindrome number otherwise not. Brief description below:

**Description of program:**

With the help of this program, we are going to determine whether the given number is palindrome or not.  To achieve the desired result, firstly we have to define a class named "Palindrome". After that we will ask the user to enter any integer type number and then we will reverse it. After reversing the number we will check whether the given number is palindrome or not. If the given number is larger, then it will display a message "Out of range!".

**Here is the code of this program**

|  |
| --- |
| **import**java.io.\*;  **public class**Palindrome  {   **public static void**main(String [] args){   **try**{   BufferedReader object = **new**BufferedReader( **new**InputStreamReader(System.in));   System.out.println("Enter number");   **int**num= Integer.parseInt(object.readLine());   **int**n = num;   **int**rev=0;   System.out.println("Number: ");   System.out.println(" "+ num);   **for**(**int**i=0; i<=num; i++){   **int**r=num%10;   num=num/10;   rev=rev\*10+r;   i=0;   }   System.out.println("After reversing the number: "+ " ");   System.out.println(" "+ rev);     **if**(n == rev){   System.out.print("Number is palindrome!");   }   **else**{   System.out.println("Number is not palindrome!");   }   }   **catch**(Exception e){   System.out.println("Out of range!");   }   } } |

[**Download this example**](http://www.roseindia.net/java/beginners/Palindrome.java)

# Write a program for calculating area and perimeter of a rectangle

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Palindrome.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/triangle.shtml)**

If you are a newbie in Java programming then our tutorials and examples will be helpful in understanding Java programming in the most simplest way. Here after reading this lesson, you will be able to write program for calculating the area and perimeter of a rectangle.

First of all create a class named **RecArea** under Java.io package. Now define two integer variable '**l'** and '**w'.** As the program will be based on keyboard numerical input, it is important for every programmer to use correct data without any mistake. In this case the exception methods like **try/catch** mechanism helps in detecting user input errors. So before starting the functional code, enclosed it with *try*clause so that any error in the statement causes the execution of the catch clauses.

Now create an abstract buffer class which is the super class of all classes and represents a stream of input bytes.  The InputSreamReader reads the character stream and stores it in the buffer class. Now use ***parseInt*** for both length and width of the rectangle. This is an instance of class method and is used to convert a string to an integer. Define the area as l\*w and perimeter as 2\*(l+w) and in the end use the catch exception.

Now compile and run the program and input the value as you see the message and get the ultimate result. If you find any kind of error, then check the whole program again.

**Here is the code of the program:**

|  |
| --- |
| **import**java.io.\*; **class**RecArea  {   **public static void**main(String[] args)    {    **int**l=0;   **int**w=0;    **try**{      BufferedReader br1 = **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.println("Enter length of rectangle : ");   l = Integer.parseInt(br1.readLine());   System.out.println("Enter width of rectangle : ");   w = Integer.parseInt(br1.readLine());   **int**area = l\*w;   System.out.println("Area of Rectangle : "+area);   **int**perimiter = 2\*(l+w);   System.out.println("Perimeter: " + perimiter);    }**catch**(Exception e){System.out.println("Error : "+e);}    }   } |

# Checking whether a year is leap or not

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/triangle.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/leapyears.shtml)**

This tutorial is going to teach you the coding for checking whether a year is a leap year or not. Here, we have taken the year 2000. So define an integer n=2000 in the class "Leapyear" and now apply "if else" condition. As we know leap year is divided by the integer 4 and so applying if condition as n/4=0, then "n" is a leap year. Now in the System.out.println write the message that the year is a leap year. Again applying "else" condition the output will be that the year is not a leap year.

**Here is the code of program:**

|  |
| --- |
| **class**Leapyear {   **public static void**main(String[] args)    {   **int**n=2000;   **if**(n%4==0){   System.out.println("The given year is a leap year");   }   **else**{   System.out.println("This is not a leap year");   } } } |

[**Download the program:**](http://www.roseindia.net/java/beginners/Leapyear.java)

# Listing out leap years between certain period

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Leapyear.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/PreparingTable.shtml)**

The programming lesson will teach you the coding for finding and listing out the leap years between two years. In the following example we have to find out the leap years between 1990 and 2006. First define the two years under a class "leapyears". Let i = 2006 and n=1990. Now with the help of for loop method initialize the year as n=1990 and n<=i. Also apply the increment statement in the loop as we have to check one by one.

As we know a leap year is divisible by 4, define an integer l=n%4. So if 'n' is divisible by 4 or l=0, then the particular year can be a leap year. For checking this, apply the **if** statement and if this satisfies then, the year will be a leap year. For listing out each year write "+n" in the System.out.println.

Now compile and run the program in the command window and see the result. If you find any error, check the whole program and find out the     
 **Here is the code of the program:**

|  |
| --- |
| **class**leapyears  {   **public static void**main(String[] args)    {   **int**i=2006;   **int**n;   **for**(n=1990; n<=i ; n++){   **int**l=n%4;   **if**(l==0){   System.out.println("leap year: "+n);   }   }   } } |

# Preparing table of a number by using loop

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/leapyears.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Prime_number.shtml)**

This tutorial will teach you the methods of preparing the table of a given number by using loop condition. As we know the loop statements are used to repeat a statement or process multiple times according to a specified condition. Loop checks certain condition and if it finds the condition is valuable then all the statements written under loop are executed.

Here we will take a number a=25 of which we have to prepare a table. Define the integer a=25 and b=1 as the initial point. Now apply "while" condition of loop and confine b<=10 as we have to make a table of 25. Again define another integer as c=a\*b, this will be the result when we multiply 'a' with 'b'. Here we have to multiply 'a' with 'b' up to 10 times like a\*1, a\*2....................a\*9, a\*10. So make define b=b+1 as increment operator.

Now compile and run the program on the command window.

**Here is the code of the prorgram:**

|  |
| --- |
| **class**PreparingTable{   **public static void**main(String[] args) {   **int**a=25, b=1;   System.out.println("the table of "+a+"= ");   **while**(b<=10){   **int**c = a\*b;   System.out.println(c);   b = b+1;   }   } } |

# Find out the prime number

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/PreparingTable.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/PrimeNumber.shtml)**

This lesson of Java programming language will teach you the coding to find out whether a given number is prime or not. Here we have used the 'for loop' statement and given the required condition for a prime number. As we know, a prime number is only divided by 1 and itself, in other words it has no other factorial other than 1 and the number itself.

Here, first make a class and named as "Primenumber" and take an integer as num=11, and define an integer 'i' as the integer other than 1 and the given number. That means, i>2 and i<num. Now apply this in the**"for loop"** statement and define an integer n=num/i as given below in the example. Now apply the **"if"** condition and if the reminder of the earlier equation comes "0", then the result will be not prime. Again the loop system will check the above condition until it has not satisfied from the starting point(2) to the end(10). Here under this loop we have to use the **"break"** statement for unnecessary checking further one point where the reminder comes zero(0).

Now after checking the whole condition, if the reminders does not come "zero", then we have to again apply the "if" condition and check whether i=num or not. If it is true then number (num) is prime. As we have taken here as num=11, then after compiling and running the program, the result will show that num is prime number.

**Here is the code program:**

|  |
| --- |
| **class**Prime\_number {   **public static void**main(String[] args) {   **int**num = 11;   **int**i;   **for**(i=2; i < num ;i++ ){   **int**n = num%i;   **if**(n==0){   System.out.println("not Prime!");   **break**;   }   }   **if**(i == num){   System.out.println("Prime number!");   }   } } |

# Prime Number in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Prime_number.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/oop-in-java.shtml)**

This Java programming tutorial, we will be read how to get prime number between 1 to given number. First of all we have to define a class "**PrimeNumber**". Java I/O package has a input stream and a output stream in which input stream is used for reading the stream and memory allocating and the output stream used for writing bytes. As in this program we are going to insert certain instruction by creating buffer reader class. Here we have to create a buffer for the string class that can be used to instantiate a changeable object for storing and processing a string of character. Now use the ParseInt method for converting the parses the string argument and define 'num' as an integer.

Now applying in this program we use two '**for**' loop. For loop will start from 1 to entered number. And another loop will start and divide it from 2 to less than those number. If number is divided by any number that means it is not prime otherwise prime number.

**Here is the code of the Program**

|  |
| --- |
| **import**java.io.\*;  **class**PrimeNumber {   **public static void**main(String[] args) **throws**Exception{   **int**i;   BufferedReader bf = **new**BufferedReader(   **new**InputStreamReader(System.in));   System.out.println("Enter number:");   **int**num = Integer.parseInt(bf.readLine());   System.out.println("Prime number: ");   **for**(i=1; i < num; i++ ){   **int**j;   **for**(j=2; j<i; j++){   **int**n = i%j;   **if**(n==0){   **break**;   }   }   **if**(i == j){   System.out.print("  "+i);   }   }   } } |

# OOPs and Its Concepts in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/PrimeNumber.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/ExceptionHandle.shtml)**

**Brief Introduction to OOP**  
Object Oriented Programming or OOP is the technique to create programs based on the real world. Unlike procedural programming, here in the OOP programming model programs are organized around objects and data rather than actions and logic. Objects represent some concepts or things and like any other objects in the real Objects in programming language have certain behavior, properties, type, and identity. In OOP based language the principal aim is to find out the objects to manipulate and their relation between each other. OOP offers greater flexibility and compatibility and is popular in developing larger application. Another important work in OOP is to classify objects into different types according to their properties and behavior. So OOP based software application development includes the analysis of the problem, preparing a solution, coding and finally its maintenance.

Java is a object oriented programming  and to understand the functionality of OOP in Java, we first need to understand several fundamentals related to objects. These include class, method, inheritance, encapsulation, abstraction, polymorphism etc.

**Class** - It is the central point of OOP and that contains data and codes with behavior. In Java everything happens within class and it describes a set of objects with common behavior. The class definition describes all the properties, behavior, and identity of objects present within that class. As far as types of classes are concerned, there are predefined classes in languages like C++ and Pascal. But in Java one can define his/her own types with data and code.

**Object** - Objects are the basic unit of object orientation with behavior, identity. As we mentioned above, these are part of a class but are not the same. An object is expressed by the variable and methods within the objects. Again these variables and methods are distinguished from each other as instant variables, instant methods and class variable and class methods.

**Methods** -  We know that a class can define both attributes and behaviors. Again attributes are defined by variables and behaviors are represented by methods. In other words, methods define the abilities of an object.

**Inheritance**- This is the mechanism of organizing and structuring software program. Though objects are distinguished from each other by some additional features but there are objects that share certain things common. In object oriented programming classes can inherit some common behavior and state from others. Inheritance in OOP allows to define a general class and later to organize some other classes simply adding some details with the old class definition. This saves work as the special class inherits all the properties of the old general class and as a programmer you only require the new features. This helps in a better data analysis, accurate coding and reduces development time.

**Abstraction** - The process of abstraction in Java is used to hide certain details and only show the essential features of the object. In other words, it deals with the outside view of an object (interface).

**Encapsulation** - This is an important programming concept that assists in separating an object's state from its behavior. This helps in hiding an object's data describing its state from any further modification by external component. In Java there are four different terms used for hiding data constructs and these are public, private, protected and package. As we know an object can associated with data with predefined classes and in any application an object can know about the data it needs to know about. So any unnecessary data are not required by an object can be hidden by this process. It can also be termed as information hiding that prohibits outsiders in seeing the inside of an object in which abstraction is implemented.

**Polymorphism -**It describes the ability of the object in belonging to different types with specific behavior of each type. So by using this, one object can be treated like another and in this way it can create and define multiple level of interface. Here the programmers need not have to know the exact type of object in advance and this is being implemented at runtime.

# Java Exception - Exception Handling in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/oop-in-java.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/identifierandpdatatype.shtml)**

*Exception,* that means exceptional errors. Actually *exceptions* are used for handling errors in programs that occurs during the program execution. During the program execution if any error occurs and you want to print your own message or the system message about the error then you write the part of the program which generate the error in the try{} block and catch the errors using catch() block. Exception turns the direction of normal flow of the program control and send to the related catch() block. Error that occurs during the program execution generate a specific object which has the information about the errors occurred in the program.

In the following example code you will see that how the exception handling can be done in java program. This example reads two integer numbers for the variables a and b. If you enter any other character except number ( 0 - 9 ) then the error is caught by *NumberFormatException* object. After thatex.getMessage() prints the information about the error occurring causes.

**Code of the program :**

|  |
| --- |
| **import**java.io.\*;  **public class**exceptionHandle{   **public static void**main(String[] args) **throws**Exception{   **try**{   **int**a,b;   BufferedReader in =    **new**BufferedReader(**new**InputStreamReader(System.in));   a = Integer.parseInt(in.readLine());   b = Integer.parseInt(in.readLine());   }   **catch**(NumberFormatException ex){   System.out.println(ex.getMessage()    + " is not a numeric value.");   System.exit(0);   }   } } |

[**Download Exception Handling Example**](http://www.roseindia.net/java/beginners/exceptionHandle.java)

# Java - Identifier and Primitive Data Types in java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ExceptionHandle.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Varconstltr.shtml)**

In this example you will learn what is**identifier** and **primitive data types** of a identifier. This tutorials will teach you how to use these identifier in you  java programming. Identifier is is a simple variable name which is defined as the value container. The type of value stored by identifier is defined by the special java keyword is termed as primitive data type.

In the given example there are some identifiers have been used like *byteident, shortident, intident, longident, charident, stringident, floatident, doubleident*. And there are some primitive data types of used identifiers have been also used in the program like *byte, short, int, long, float, double, char*and*String.*

All the data type has it's own capacity to keep the maximum value. Which have been mentioned below :

|  |  |  |
| --- | --- | --- |
| **Primitive Data Types** | | |
| **Keyword** | **Description** | **Size/Format** |
| Integers | | |
| byte | Byte-length integer | 8-bit two's complement |
| short | Short integer | 16-bit two's complement |
| int | Integer | 32-bit two's complement |
| long | Long integer | 64-bit two's complement |
| Real numbers | | |
| float | Single-precision floating point | 32-bit IEEE 754 |
| double | Double-precision floating point | 64-bit IEEE 754 |
| Other types | | |
| char | A single character | 16-bit Unicode character |
| boolean | A boolean value (true or false) | true or false |

Source: [**http://java.sun.com/docs/books/tutorial/java/nutsandbolts/datatypes.html**](http://java.sun.com/docs/books/tutorial/java/nutsandbolts/datatypes.html)

**Code of the Program :**

|  |
| --- |
| **public class**identifierandpdatatype{   **public static void**main(String[] args){   **byte**byteident = 3;   **short**shortident=100;   **int**intident = 10;   **long**longident = 40000;   **char**charident = 'a';   String stringident = "chandan";   **float**floatident = 12.0045f;   **double**doubleident = 2333333.000000000033343343434f;   System.out.println(byteident + " is the value of identifire  named 'byteident' which primitive data type is byte.");   System.out.println(shortident + " is the value of  identifire named 'shortident' which primitive data type is short.");   System.out.println(intident + " is the value  of identifire named 'intident' which primitive data type is int.");   System.out.println(longident + " is the value  of identifire named 'longident' which primitive data type is long.");   System.out.println(charident + " is the value  of identifire named 'charident' which primitive data type is char.");   System.out.println(stringident + " is the value   of identifire named 'stringident' which primitive data type is string.");   System.out.println(floatident + " is the value  of identifire named 'floatident' which primitive data type is float.");   System.out.println(doubleident + " is the value   of identifire named 'doubleident' which primitive data type is double.");   } } |

# Java - Variable, Constant and Literal in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/identifierandpdatatype.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/java-read-file-line-by-line.shtml)**

In this example you will see that how you can use variables, constants and literals in your program easily.

**Variable :** You can assign the values to the variable once it has been declared. The values of the variable can be changed anywhere in the program if the variable is accessible in that scope. In this example we have used the variable  ***intvariable*** to illustrate this.

**Constants:** Constants are declared using the final keyword. The values of the constant can't be changed once its declared.

**Literal :** literal is an explicit number or string constant used in Java programs. This specifies the syntax of your declaration of different types of values and operations. That mean literal is totally based on the syntax. Whenever you want to show the message with java special symbols then you have to use the literals. For example to show the message "Directory of this file : c:\code\varconstltr.java" then you can write as shown below

System.out.println("Directory of this file : c:\\code\\varconstltr.java");

There are three type of literals : *Numeric Literals, Char type Literals, String Literals*as follows.

**Code of the Program :**

|  |
| --- |
| **public class**varconstltr{   **public static final int**constint=5;   **public static void**main(String[] args){   **int**intvariable;     **for**(**int**i = 0;i <= 10;i++){   intvariable = i;   System.out.println("All the values are : \n intvariable = "  + intvariable + "\nconstint = " + constint);   }   System.out.println("Directory of this file : c:\\code\\varconstltr.java");   } } |

# Java Read File Line by Line - Java Tutorial

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Varconstltr.shtml)    [http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)  [http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/CopyFile.shtml)**

In the section of Java Tutorial you will learn how to write java program to read file line by line. We will use the DataInputStream class to Read text File Line by Line.

**Class DataInputStream**  
A data input stream is use to read primitive Java data types from an underlying input stream in a machine-independent way. An application uses a data output stream to write data that can later be read by a data input stream.

Data input streams and data output streams represent Unicode strings in a format that is a slight modification of UTF-8. (For more information, see X/Open Company Ltd., "File System Safe UCS Transformation Format (FSS\_UTF)", X/Open Preliminary Specification, Document Number: P316. This information also appears in ISO/IEC 10646, Annex P.) Note that in the following tables, the most significant bit appears in the far left-hand column.

**BufferedReader**

Read text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines.

The buffer size may be specified, or the default size may be used. The default is large enough for most purposes.

In general, each read request made of a Reader causes a corresponding read request to be made of the underlying character or byte stream. It is therefore advisable to wrap a BufferedReader around any Reader whose read() operations may be costly, such as FileReaders and InputStreamReaders. For example,

BufferedReader in

= new BufferedReader(new FileReader("foo.in"));

will buffer the input from the specified file. Without buffering, each invocation of read() or readLine() could cause bytes to be read from the file, converted into characters, and then returned, which can be very inefficient.

Programs that use DataInputStreams for textual input can be localized by replacing each DataInputStream with an appropriate BufferedReader.

**Here is the code of java program to Read text File Line by Line:**

|  |
| --- |
| **import**java.io.\*; **class**FileRead  {  **public static void**main(String args[])   {   **try**{   // Open the file that is the first    // command line parameter   FileInputStream fstream = **new**FileInputStream("textfile.txt");   // Get the object of DataInputStream   DataInputStream in = **new**DataInputStream(fstream);   BufferedReader br = **new**BufferedReader(**new**InputStreamReader(in));   String strLine;   //Read File Line By Line   **while**((strLine = br.readLine()) != **null**)   {   // Print the content on the console   System.out.println (strLine);   }   //Close the input stream   in.close();     }**catch**(Exception e){//Catch exception if any   System.err.println("Error: " + e.getMessage());   }   } } |

# Java - Copying one file to another

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/java-read-file-line-by-line.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/java-write-to-file.shtml)**

This example illustrates how to copy contents from one file to another file. This topic is related to the I/O (input/output) ofjava.io package.

In this example we are using File class of *java.io* package. The File class is an abstract representation of file and directory pathnames. This class is an abstract, system-independent view of hierarchical pathnames. An abstract pathname has two components:

1. An optional system-dependent prefix string,  
   such as a disk-drive specifier, "/" for the UNIX root directory, or "\\" for a Win32 UNC pathname, and
2. A sequence of zero or more string names.

**Explanation**

This program copies one file to another file. We will be declaring a function called *copyfile* which copies the contents from one specified file to another specified file.

*copyfile(String srFile, String dtFile)*

The function copyfile(String srFile, String dtFile) takes both file name as parameter. The function creates a new File instance for the file name passed as parameter

*File f1 = new File(srFile);  
File f2 = new File(dtFile);*

and creates another InputStream instance for the input object and OutputStream instance for the output object passed as parameter

*InputStream in = new FileInputStream(f1);  
OutputStream out = new FileOutputStream(f2);*

and then create a byte type buffer for buffering the contents of one file and write to another specified file from the first one specified file.

*// For creating a byte type buffer  
byte[] buf = new byte[1024];  
// For writing to another specified file from buffer buf  
out.write(buf, 0, len);*

**Code of the Program :**

|  |
| --- |
| **import**java.io.\*;  **public class**CopyFile{   **private static void**copyfile(String srFile, String dtFile){   **try**{   File f1 = **new**File(srFile);   File f2 = **new**File(dtFile);   InputStream in = **new**FileInputStream(f1);      //For Append the file. //  OutputStream out = new FileOutputStream(f2,true);    //For Overwrite the file.   OutputStream out = **new**FileOutputStream(f2);    **byte**[] buf = **new byte**[1024];   **int**len;   **while**((len = in.read(buf)) > 0){   out.write(buf, 0, len);   }   in.close();   out.close();   System.out.println("File copied.");   }   **catch**(FileNotFoundException ex){   System.out.println(ex.getMessage() + " in the specified directory.");   System.exit(0);   }   **catch**(IOException e){   System.out.println(e.getMessage());     }   }   **public static void**main(String[] args){   **switch**(args.length){   **case**0: System.out.println("File has not mentioned.");     System.exit(0);   **case**1: System.out.println("Destination file has not mentioned.");   System.exit(0);   **case**2: copyfile(args[0],args[1]);   System.exit(0);   **default**: System.out.println("Multiple files are not allow.");   System.exit(0);   }   } } |

# Java Write To File - Java Tutorial

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CopyFile.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/java-create-directory.shtml)**

In the section of Java Tutorial you will learn how to write java program to write to a file. We will use the class FileWriter and BufferedWriter to write to a file.

**Class FileWriter**  
The FileWriter is a class used for writing character files. The constructors of this class assume that the default character encoding and the default byte-buffer size are acceptable. To specify these values yourself, construct an OutputStreamWriter on a FileOutputStream.  
  
**BufferedWriter**

The BufferWriter class is used to write text to a character-output stream, buffering characters so as to provide for the efficient writing of single characters, arrays, and strings.  
  
**Here is the code of java program to write text to a file:**

|  |
| --- |
| **import**java.io.\*; **class**FileWrite  {  **public static void**main(String args[])   {   **try**{   // Create file    FileWriter fstream = **new**FileWriter("out.txt");   BufferedWriter out = **new**BufferedWriter(fstream);   out.write("Hello Java");   //Close the output stream   out.close();   }**catch**(Exception e){//Catch exception if any   System.err.println("Error: " + e.getMessage());   }   } } |

[**Download the code**](http://www.roseindia.net/java/beginners/FileWrite.java)

# Java Create Directory - Java Tutorial

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/java-write-to-file.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DeleteFile.shtml)**

In the section of Java Tutorial you will learn how to create directory using java program. This program also explains the process of creating all non-existent ancestor directories automatically. We will use the class File class to crate the directory.

**Class File**  
The File class an abstract representation of file and directory pathnames. File class is used to interact with the files system.  
  
  
**Here is the code for creating directory and all non-existing ancestor directories:**

|  |
| --- |
| **import**java.io.\*; **class**CreateDirectory  {  **public static void**main(String args[])   {   **try**{   String strDirectoy ="test";   String strManyDirectories="dir1/dir2/dir3";    // Create one directory   **boolean**success = (   **new**File(strDirectoy)).mkdir();   **if**(success) {   System.out.println("Directory: "     + strDirectoy + " created");   }     // Create multiple directories   success = (**new**File(strManyDirectories)).mkdirs();   **if**(success) {   System.out.println("Directories: "     + strManyDirectories + " created");   }    }**catch**(Exception e){//Catch exception if any   System.err.println("Error: " + e.getMessage());   }   } } |

[**Download the code**](http://www.roseindia.net/java/beginners/CreateDirectory.java)

# ava - Deleting the file or Directory

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/java-create-directory.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DirectoryListing.shtml)**

This example illustrates how to delete the specified file or directory after checking weather the file exists or not. This topic is related to the I/O (input/output) of java.iopackage.

In this example we are using File class of *java.io* package. The File class is an abstract representation of file and directory pathnames.

**Explanation**

This program deletes the specified file if that exists. We will be declaring a function called *deletefile*which deletes the specified directory or file.

*deletefile(String file)*

The function deletefile(String file) takes file name as parameter. The function creates a new File instance for the file name passed as parameter

*File f1 = new File(file);*

and delete the file using delete function *f1.delete();*which return the Boolean value (true/false). It returnstrue if and only if the file or directory is successfully deleted; false otherwise.

***delete****()*

Deletes the file or directory denoted by this abstract pathname. If this pathname denotes a directory, then the directory must be empty in order to be deleted.

**Returns:**

true if and only if the file or directory is successfully deleted; false otherwise

**Code of the Program :**

|  |
| --- |
| **import**java.io.\*;  **public class**DeleteFile{   **private static void**deletefile(String file){   File f1 = **new**File(file);   **boolean**success = f1.delete();   **if**(!success){   System.out.println("Deletion failed.");   System.exit(0);   }**else**{   System.out.println("File deleted.");     }   }   **public static void**main(String[] args){   **switch**(args.length){   **case**0: System.out.println("File has not mentioned.");   System.exit(0);   **case**1: deletefile(args[0]);   System.exit(0);   **default**: System.out.println("Multiple files are not allow.");   System.exit(0);   }   } } |

[**Download File Deletion Example**](http://www.roseindia.net/java/beginners/DeleteFile.java)

# Java Word Count - Word Count Example in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/DirectoryListing.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Applet.shtml)**

This example illustrates how to count the number of lines, number of words and number of characters in the specified file. Program takes the file name as parameter and it counts the number of words and lines present in the file. Parameter is optional and if you simply run the program without mentioning the file name then you will have to input some strings and program will count the number of characters and number of words for your given strings. This topic is related to the I/O (input/output) of java.io package.

In this example we are using FileReader class of *java.io* package. The File class is an abstract representation of file and directory pathnames.

**Explanation**

This program counts the number of lines, number of words and number of characters in the specified file. We will be declaring two functions called *wordcount*and *linecount* in the program. The function linecount has been overloaded according to the passing argument. If you input contents through the file then *linecount* function will be called (If specified file exists) otherwise main function counts the number of characters and number of lines (always, the number of line will be only 1 in this condition) itself but for the counting of the number of words by using the *wordcount*function.

*wordcount(String line)*

The function wordcount(String line) takes either the content of the specified file or arguments passed with the run command for a java program as parameter '*String line'*. The *wordcount()* function is using*arrayname.charAt(index)* to find position of space in the string.  A counter variable '*numWords'*is used to count the number of words.

*linecount(String fileName);*

The function linecount(String fileName) takes the specified file name as a string parameter and create a instance for the FileReader class to buffering then the file or string and it is passed to the function linecount(String fName, BufferedReader in).

*linecount(String fileName, BufferedReader);*

The function linecount(String fName, BufferedReader in) takes the specified file name and created instance *in*for the BufferedReader class by calling function linecount(String fileName) and assign the content of the buffer in a string variable *line.*And then the function linecount(String fileName, BufferedReader) counts and print the number of characters, number of lines. To count the number of words call the wordcount(String line) function.

**Code of the Program :**

|  |
| --- |
| **import**java.io.\*;  **public class**WordCount{   **private static void**linecount(String fName, BufferedReader in)  **throws**IOException{   **long**numChar = 0;   **long**numLine=0;   **long**numWords = 0;   String line;   **do**{   line = in.readLine();   **if**(line != **null**){   numChar += line.length();   numWords += wordcount(line);   numLine++;   }   }**while**(line != **null**);   System.out.println("File Name: " + fName);   System.out.println("Number of characters: " + numChar);   System.out.println("Number of words: " + numWords);   System.out.println("Number of Lines: " + numLine);   }   **private static void**linecount(String fileName){   BufferedReader in = **null**;   **try**{   FileReader fileReader = **new**FileReader(fileName);   in = **new**BufferedReader(fileReader);   linecount(fileName,in);   }   **catch**(IOException e){   e.printStackTrace();   }   }   **private static long**wordcount(String line){   **long**numWords = 0;   **int**index = 0;   **boolean**prevWhiteSpace = **true**;   **while**(index < line.length()){   **char**c = line.charAt(index++);   **boolean**currWhiteSpace = Character.isWhitespace(c);   **if**(prevWhiteSpace && !currWhiteSpace){   numWords++;   }   prevWhiteSpace = currWhiteSpace;   }   **return**numWords;   }   **public static void**main(String[] args){     **long**numChar = 0;   **long**numLine=0;   String line;   **try**{   **if**(args.length == 0)   {   BufferedReader in =    **new**BufferedReader(**new**InputStreamReader(System.in));   line = in.readLine();   numChar = line.length();   **if**(numChar != 0){   numLine=1;   }   System.out.println("Number of characters: " + numChar);   System.out.println("Number of words: " + wordcount(line));   System.out.println("Number of lines: " + numLine);   }**else**{   **for**(**int**i = 0; i < args.length; i++){   linecount(args[i]);   }   }   }   **catch**(IOException e){   e.printStackTrace();   }   } } |

# ava - Math class in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/BEOJ.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/StringBuffer.shtml)**

In this example you will learn about *Math*class. This example explains how you can use functions provided by the *Math* class like *E, PI, round, abs, ceil, exp, floor,IEEEremainder, max, min, pow, random,rint, sqrt* etc. to manipulate the mathematical operation in your program. The *Math* class is used to operate the calculations. There is not necessary to import any package for the *Math*class because this is already in java.lang package.

Any expressions can be operated through certain method calls. There are some functions have been used in the given example. All the functions have been explained below with example :

*E*

This is E field of the Math class which returns you a default exponent value that is closer than any other to e, the base of the natural logarithms.

*PI*

This is also a field of the *Method*class which returns you a default pi value, the ratio of the circumference of a circle to its diameter.

*abs()*

This is the abs() function which returns you the absolute number.

*ceil()*

This is the ceil() function which returns you the smallest value but greater than the argument.

*exp()*

This is the exp() function which returns you the exponential value raised to the power of a double value.

*floor()*

This is the floor() function which returns you the largest value but less than the argument.

*IEEEremainder()*

This is the IEEEremainder() which returns you the remainder for the given dividend and divisor.

*max()*

This is the max() function which distinguishes the maximum value from the two given value.

*min()*

This is the min() function which distinguishes the minimum value from the two given value.

*pow()*

This is the pow() function which returns you the number raised to the power of a first given value by the another one.

*random()*

This is the random() function which returns you the random number. It is absolutely system generated.

*rint()*

This is the rint() function which returns you a value closest to the given value.

*round()*

This is the round() function which returns you a value that is in the rounded form.

*sqrt()*

This is the sqrt() function which returns you the square root of the specified value.

**Code for the program :**

|  |
| --- |
| **public class**mathclass{   **public static void**main(String[] args){   //E and round()   System.out.println("e = " + Math.round(Math.E\*100)/100f);   //PI   System.out.println("pi = " + Math.round(Math.PI\*100)/100f);   //abs()   System.out.println("Absolute number = " + Math.abs(Math.PI));   //ceil()   System.out.println("Smallest value but greater than  the argument = " + Math.ceil(Math.PI));     //exp()   System.out.println("Exponent number powered by  the argument = " + Math.exp(0));   //floor()   System.out.println("Largest value but less  than the argument = " + Math.floor(Math.E));   //IEEEremainder()   System.out.println("Remainder = " +  Math.IEEEremainder(5.3f,2.2f));   //max()   System.out.println("Maximum Number = " +  Math.max(10,10.3));   //min()   System.out.println("Minimum Number = " +  Math.min(10,10.3));   //pow()   System.out.println("Power = " + Math.pow(10,3));   //random()   System.out.println("Random Number = " +  Math.random());   //rint()   System.out.println("Closest to the Argument  = " + Math.rint(30));   //round()   System.out.println("Round = " + Math.round(Math.E));   //sqrt()   System.out.println("Square Root = " + Math.sqrt(400));   } } |

# Java - StringBuffer class in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/MathClass.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/TypeCasting.shtml)**

In this example you will learn about StringBuffer class. This example explains how you can use functions provided by the StringBuffer class like *append, insert, reverse, setCharAt, charAt, length, deleteCharAt, substring, delete, capacity*etc. to manipulate the string operation in your program.

The StringBuffer class is used to represent characters that can be modified. This is simply used for concatenation or manipulation of the strings.

*StringBuffer* is mainly used for the dynamic string concatenation which enhances the performance. A string buffer implements a mutable sequence of characters. A string buffer is like a [**String**](http://java.sun.com/j2se/1.4.2/docs/api/java/lang/String.html), 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. There are some functions used in the given example. All the functions have been explained below with example :

*append()*

This is the append() function used for the concatenate the string in string buffer. This is better to use for dynamic string concatenation. This function works like a simple string concatenation such as : String str = str + "added string";.

*insert()*

This is the insert() function used to insert any string or character at the specified position in the given string.

*reverse()*

This is the reverse() function used to reverse the string present in string buffer.

*setCharAt()*

This is the setCharAt() function which is used to set the specified character in buffered string at the specified position of the string in which you have to set the given character.

*charAt()*

This is the charAt() function which is used to get the character at the specified position of the given string.

*substring()*

This is the substring() function which is used to get the sub string from the buffered string from the initial position to end position (these are fixed by you in the program).

*deleteCharAt()*

This is the deleteCharAt() function which is used to delete the specific character from the buffered string by mentioning that's position in the string.

*length()*

This is the length() function is used to finding the length of the buffered string.

*delete()*

This is the delete() function is used to delete multiple character at once from *n*position to *m*position (n and m are will be fixed by you.) in the buffered string.

*capacity()*

This is the capacity() function is used to know about the current characters kept which is displayed like : number of characters + 6.

Code for the program :

|  |
| --- |
| **import**java.io.\*;  **public class**stringBuffer{   **public static void**main(String[] args) **throws**Exception{   BufferedReader in =  **new**BufferedReader(**new**InputStreamReader(System.in));   String str;   **try**{   System.out.print("Enter your name: ");   str = in.readLine();   str += ",  This is the example of SringBuffer class and it's functions.";    //Create a object of StringBuffer class   StringBuffer strbuf = **new**StringBuffer();   strbuf.append(str);   System.out.println(strbuf);   strbuf.delete(0,str.length());      //append()   strbuf.append("Hello");   strbuf.append("World");     //print HelloWorld   System.out.println(strbuf);      //insert()   strbuf.insert(5,"\_Java ");   //print Hello\_Java World   System.out.println(strbuf);        //reverse()   strbuf.reverse();   System.out.print("Reversed string : ");   System.out.println(strbuf);     //print dlroW avaJ\_olleH   strbuf.reverse();   System.out.println(strbuf);   //print Hello\_Java World      //setCharAt()   strbuf.setCharAt(5,' ');   System.out.println(strbuf);     //prit Hello Java World      //charAt()   System.out.print("Character at 6th position : ");   System.out.println(strbuf.charAt(6));     //print J      //substring()   System.out.print("Substring from position 3 to 6 : ");   System.out.println(strbuf.substring(3,7));   //print lo J      //deleteCharAt()   strbuf.deleteCharAt(3);   System.out.println(strbuf);   //print Helo java World      //capacity()   System.out.print("Capacity of StringBuffer object : ");   System.out.println(strbuf.capacity());    //print 21      //delete() and length()   strbuf.delete(6,strbuf.length());       System.out.println(strbuf);     //no anything   }   **catch**(StringIndexOutOfBoundsException e){   System.out.println(e.getMessage());   }   } } |

# Java - Arithmetic Operation, Conversion and Casts in java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/StringBuffer.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/array.shtml)**

This example illustrates that what is *type casting*? **Type Casting**  refers to changing an entity of one [**datatype**](http://en.wikipedia.org/wiki/Datatype) into another. This is important for the type conversion in developing any application. If you will store a int value into a byte variable directly, this will be illegal operation. For storing your calculated int value in a byte variable you will have to change the type of resultant data which has to be stored. This type of operation has illustrated below :

In this example we will see that how to convert the data type by using type casting. In the given line of the code *c = (****char****)(t?1:0);*illustrates that if t which is boolean type variable is true then value of c which is the char type variable will be 1 but 1 is a numeric value. So, 1 is changed into character according to the Unicode value. But in this line *c = (****char****)(t?'1':'0');* 1 is already given as a character which will be stored as it is in the char type variable c.

**Code of the program :**

|  |
| --- |
| **public class**conversion{   **public static void**main(String[] args){   **boolean**t = **true**;   **byte**b = 2;   **short**s = 100;   **char**c = 'C';   **int**i = 200;   **long**l = 24000;   **float**f = 3.14f;   **double**d = 0.000000000000053;   String g = "string";   System.out.println("Value of all the variables like");   System.out.println("t = " + t );   System.out.println("b = " + b );   System.out.println("s = " + s );   System.out.println("c = " + c );   System.out.println("i = " + i );   System.out.println("l = " + l );   System.out.println("f = " + f );   System.out.println("d = " + d );   System.out.println("g = " + g );   System.out.println();   //Convert from boolean to byte.   b = (**byte**)(t?1:0);   System.out.println("Value of b after conversion : " + b);   //Convert from boolean to short.   s = (**short**)(t?1:0);   System.out.println("Value of s after conversion : " + s);   //Convert from boolean to int.   i = (**int**)(t?1:0);   System.out.println("Value of i after conversion : " + i);   //Convert from boolean to char.   c = (**char**)(t?'1':'0');   System.out.println("Value of c after conversion : " + c);   c = (**char**)(t?1:0);   System.out.println("Value of c after conversion in unicode : " + c);   //Convert from boolean to long.   l = (**long**)(t?1:0);   System.out.println("Value of l after conversion : " + l);   //Convert from boolean to float.   f = (**float**)(t?1:0);   System.out.println("Value of f after conversion : " + f);   //Convert from boolean to double.   d = (**double**)(t?1:0);   System.out.println("Value of d after conversion : " + d);   //Convert from boolean to String.   g = String.valueOf(t);   System.out.println("Value of g after conversion : " + g);   g = (String)(t?"1":"0");   System.out.println("Value of g after conversion : " + g);   **int**sum = (**int**)(b + i + l + d + f);   System.out.println("Value of sum after conversion : " + sum);   } } |

# Array Example - Array in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/TypeCasting.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/CopyArray.shtml)**

**Array:**Array is the most important thing in any programming language. By definition, array is the static memory allocation. It allocates the memory for the same data type in sequence. It contains multiple values of same types. It also store the values in memory at the fixed size. Multiple types of arrays are used in any programming language such as: one - dimensional, two - dimensional or can say multi - dimensional.

**Declaration of an array:**  
int num[]; or int num = new int[2];  
Some times user declares an array and it's size simultaneously. You may or may not be define the size in the declaration time. such as:  
int num[] = {50,20,45,82,25,63};

In this program we will see how to declare and implementation. This program illustrates that the array working way. This program takes the numbers present in the num[] array in unordered list and prints numbers in ascending order. In this program the sort() function of the java.util.\*; package is using to sort all the numbers present in the num[] array. The Arrays.sort() automatically sorts the list of number in ascending order by default. This function held the argument which is the array name num.

**Here is the code of the program:-**

|  |
| --- |
| **import**java.util.\*;  **public class**array{   **public static void**main(String[] args){   **int**num[] = {50,20,45,82,25,63};   **int**l = num.length;   **int**i,j,t;   System.out.print("Given number : ");   **for**(i = 0;i < l;i++ ){   System.out.print("  " + num[i]);   }   System.out.println("\n");   System.out.print("Accending order number : ");   Arrays.sort(num);     **for**(i = 0;i < l;i++){   System.out.print("  " + num[i]);   }   } } |

**Output of the program:**

|  |
| --- |
| **C:\chandan>javac array.java  C:\chandan>java array Given number : 50 20 45 82 25 63  Ascending order number : 20 25 45 50 63 82** |

# Copying an array to another

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/array.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/charat.shtml)**

**Java Copy Array Example:**  
In this tutorial, you will learn how to copy data from one array to another. Here, providing you an example with code and its explanation to given below.

This is a program that follows one dimensional array. In  this program, firstly we have to define a class**"CopyArray"**. Now, we take two integer type array like: **array1[]** and **array2[]**. The array1[] contains some integer type values (2,3,4,5,8,9) and another  is blank. After initializing this, now we get number of rows and columns by using the **array1.length.** We use the '**for**' loop statement for displaying and copying ( **array1[j] = array2[j]**) the array.

**Here is the code of this program:**

|  |
| --- |
| **public class**CopyArray {   **public static void**main(String[] args) {   **int**array1[]= {2,3,4,5,8,9};   **int**array2[] = **new int**[6];   System.out.println("array:");    System.out.print("[");   **for**(**int**i=0; i<array1.length; i++){   System.out.print(" "+array1[i]);   }   System.out.print("]");   System.out.println("\narray1:");   System.out.print("[");   **for**(**int**j=0; j<array1.length; j++){   array2[j] = array1[j];   System.out.print(" "+ array2[j]);   }   System.out.print("]");   } } |

[**Download this Example**](http://www.roseindia.net/java/beginners/CopyArray.java)

# charAt() Method In Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CopyArray.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/length.shtml)**

In this section, you will get the detailed explanation about the **charAt()** method of**String** class. We are going for using***charAt()***method. You can see how to use the method by syntax. There is a section provided for understanding the usage of the method practically.

**Description of program:**

Here, you will understand about the procedure of the **charAt()** method through the following java program. This program reads a string as input through the keyboard. And first this shows the length of the string then find the character at the 4th (mentioned as a parameter of the method) position of the string that is retrieved by using **charAt()** method. This method takes a parameter that is the position of the string. The **charAt()** method returns a character value for the character at the given position of the string.

**Here is the code of this program:**

|  |
| --- |
| **import**java.io.\*; **class**CharAt{   **public static void**main(String[] args){   **try**{   BufferedReader object= **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.println("Enter the String");   String s=object.readLine();   **int**len=s.length();   System.out.println(len);   **char**char1=s.charAt(4);   System.out.println(char1);   }   **catch**(Exception e){}   } } |

**OOutput this program:**

|  |
| --- |
| **C:\java\_work>java CharAt Enter the String roseindia 9 i C:\java\_work>** |

# length() Method In Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/charat.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/ConvertInUpperCase.shtml)**

In this section, you will learn how to use**length()** method of the String class. We are going to using ***length()*** method. These method are illustration as follow here:

**Description of program:**

Here, we will know that how many character of length in String. First of all, we have to define class named "**StringLength**". Inside of class we have to define method of class in  main class. After that we create one method which returns the integer type values. So we are using to length() method of the String class for the purpose.

**length() method :**This method return the length of this string as an integer value.

**Here is the the code of this program:**

|  |
| --- |
| **import**java.io.\*;  **class**StringLength{   **public static void**main(String[] args){   **try**{   BufferedReader object=   **new**BufferedReader (**new**InputStreamReader(System.in));   System.out.println("Eneter string value:");   String s=object.readLine();   **int**len=s.length();   System.out.println(len);   }   **catch**(Exception e){}   } } |

**Output this program:**

|  |
| --- |
| **C:\java\_work>javac StringLength.java  C:\java\_work>java StringLength Eneter string value: amar 4 C:\java\_work>** |

**toUpperCase() Method In Java**Posted on: June 5, 2007 at 12:00 AM

**In this section, you will learn how to use toUpperCase() method of the String class.**

# toUpperCase() Method In Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/length.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/string-array.shtml)**

In this section, you will learn how to use**toUpperCase()** method of the String class. We are going for using ***toUpperCase()***method. This method is explained as follows:

**Description of program:**

Here, you will see the procedure of converting letters of the string in uppercase letter. So, we are using toUpperCase() method of the String class for the purpose.

The following program convert the string "india" into "INDIA" by using toUpperCase() method.

**toUpperCase():** This method returns a string value.

**Here is the code of this program:**

|  |
| --- |
| **public class**ConvertInUpperCase{   **public static void**main(String args[]){   String roseindia = "india";   System.out.println("String is : " + roseindia);   String upper = roseindia.toUpperCase();   System.out.println("String in uppercase letter: " + upper);   } } |

**Output of program:**

|  |
| --- |
| **C:\java\_work>javac ConvertInUpperCase.java  C:\java\_work>java ConvertInUpperCase String is : india String in uppercase letter: INDIA  C:\java\_work>** |

# String Array In Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ConvertInUpperCase.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/ArrayAverage.shtml)**

In this section, you will learn how to use**string array** in Java. Here, you will see how to declare a string array and the syntax for using in the program. This section provides you a simple java program which illustrates about the string array in very efficient manner.

**Program Description:**

Following code of the program declares a string array and store some strings like "chandan", "tapan", "Amar", "santosh", and "deepak" to it. And in the main method these string are displayed one by one by retrieving from the specified string array named roseindia. In this program all the string values are stored in the roseindia string array at the declaration time.

**Here is the code of this program:**

|  |
| --- |
| **class**StringCharacter  {   **static**String[] roseindia={"chanan","tapan","Amar","santosh","deepak"};   **public static void**main(String args[]){   **for**(**int**i=0;i<5;i++){   System.out.println(roseindia[i]);   }   } } |

**Output of program:**

|  |
| --- |
| **C:\>javac StringCharacter.java  C:\>java StringCharacter chanan tapan Amar santosh deepak  C:\>** |

[**Download this Example.**](http://www.roseindia.net/java/beginners/StringCharacter.java)

# Average of Array

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/string-array.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/MatrixExample.shtml)**

In this section, we will learn how to get an average of array. For this, first of all we have to define a class name **"ArrayAverage"**that has double type array to contain some values. Now we take a double type data that calculates the average of array (**result/nums.length**). And finally it will display the result on the command prompt with message by using the **System.out.println()**.

**Here is the code of this program**

|  |
| --- |
| **public class**ArrayAverage{   **public static void**main(String[] args) {   **double**nums[]={1.0,2.3,3.4,4.5,40.5};   **double**result=0.0;   **int**i=0;   **for**(i=0; i < nums.length; i++){   result=result + nums[i];   }   System.out.println("Average is =" + result/nums.length);   } } |

# Matrix Example in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CopyArray.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/MatrixSum.shtml)**

In Java tutorial, you will learn about array and matrix. An array is the collection of same data type values. If we create a variable of integer type then, the array of int can only store the int values. It can't store other than int data type.

**Matrix:** A matrix is a collection of data in rows and columns format.

**Description of program:**

In this program we are going to implement a matrix. To make a program over the two dimensional array, first of all we have to declare class named as **MatrixExample** that has one static method***outputArray()*** which takes integer type array and represents it. For displaying the matrix we need to its rows and column by using the**array.length** method. Now, we use the **for** loop to print all the values stored in the array. At last use the **main()** method inside which we are going to declare the values of the multidimensional array which we are going to use. Call the ***outputArray()*** method inside the main method. The output will be displayed to the user by **println()** method.

**Here is the code of this Example:**

|  |
| --- |
| **class**MatrixExample{   **public static void**main(String[] args)  {   **int**array[][]= {{1,3,5},{2,4,6}};   System.out.println("Row size= " + array.length);   System.out.println("Column size = " + array[1].length);   outputArray(array);   }     **public static void**outputArray(**int**[][] array) {  **int**rowSize = array.length;  **int**columnSize = array[0].length;  **for**(**int**i = 0; i <= 1; i++) {  System.out.print("[");  **for**(**int**j = 0; j <= 2; j++) {  System.out.print(" " + array[i][j]);  }  System.out.println(" ]");  }  System.out.println();  } } |

# Sum of  two Matrix

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/MatrixExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/MatrixMultiply.shtml)**

In this section, we are going to calculate the sum of two matrix and containing its rows and columns. See below for better understanding to this.

In this program we are going to calculate the sum of two matrix. To make this program, we need to declare two dimensional array of type integer. Firstly it calculates the length of the both the arrays. Now we need to make a matrix out of it. To make the matrix we will use the for loop. By making use of the for loop the rows and column will get divide. This process will be performed again for creating the second matrix.

After getting both the matrix with us, we need to sum both the matrix. The both matrix will be added by using the for loop with array[i][j]+array1[i][j]. The output will be displayed by using the **println()** method.

**Here is the code of this program:**

|  |
| --- |
| **class**MatrixSum{   **public static void**main(String[] args)  {   **int**array[][]= {{4,5,6},{6,8,9}};   **int**array1[][]= {{5,4,6},{5,6,7}};   System.out.println("Number of Row= " + array.length);   System.out.println("Number of Column= " + array[1].length);   **int**l= array.length;   System.out.println("Matrix 1 : ");   **for**(**int**i = 0; i < l; i++) {   **for**(**int**j = 0; j <= l; j++) {   System.out.print(" "+ array[i][j]);   }     System.out.println();   }     **int**m= array1.length;   System.out.println("Matrix 2 : ");   **for**(**int**i = 0; i < m; i++) {   **for**(**int**j = 0; j <= m; j++) {   System.out.print(" "+array1[i][j]);   }     System.out.println();   }   System.out.println("Addition of both matrix : ");   **for**(**int**i = 0; i < m; i++) {   **for**(**int**j = 0; j <= m; j++) {   System.out.print(" "+(array[i][j]+array1[i][j]));   }     System.out.println();   }   } } |

Output of program:

|  |
| --- |
| C:\amar work>javac MatrixSum.java  C:\amar work>java MatrixSum Number of Row= 2 Number of Column= 3 Matrix 1 : 4 5 6 6 8 9 Matrix 2 : 5 4 6 5 6 7 Addition of both matrix : 9  9   12 11 14  16 |

# Multiplication of two Matrix

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/MatrixSum.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/SquareMatrix.shtml)**

This is a simple java program that teaches you for multiplying two matrix to each other. Here providing you Java source code with understanding the Java developing application program. We are going to make a simple program that will multiply two matrix. Two dimensional array represents the matrix.

Now, make this program, you have to declare two multidimensional array of type integer. Program uses two for loops to get number of rows and columns by using the **array1.length**. After getting both matrix then multiply to it. Both matrix will be multiplied to each other by using '**for'** loop. So the output will be displayed on the screen command prompt by using the **println()** method.

**Here is the code of this program:**

|  |
| --- |
| **class**MatrixMultiply{   **public static void**main(String[] args)  {   **int**array[][] = {{5,6,7},{4,8,9}};   **int**array1[][] = {{6,4},{5,7},{1,1}};   **int**array2[][] = **new int**[3][3];   **int**x= array.length;   System.out.println("Matrix 1 : ");   **for**(**int**i = 0; i < x; i++) {   **for**(**int**j = 0; j <= x; j++) {   System.out.print(" "+ array[i][j]);   }   System.out.println();   }     **int**y= array1.length;   System.out.println("Matrix 2 : ");   **for**(**int**i = 0; i < y; i++) {   **for**(**int**j = 0; j < y-1; j++) {   System.out.print(" "+array1[i][j]);   }     System.out.println();   }      **for**(**int**i = 0; i < x; i++) {   **for**(**int**j = 0; j < y-1; j++) {   **for**(**int**k = 0; k < y; k++){      array2[i][j] += array[i][k]\*array1[k][j];   }   }    }   System.out.println("Multiply of both matrix : ");   **for**(**int**i = 0; i < x; i++) {   **for**(**int**j = 0; j < y-1; j++) {   System.out.print(" "+array2[i][j]);   }     System.out.println();   }   } } |

# Square Elements of Two Dimensional Array

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/MatrixMultiply.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/IfElse.shtml)**

This is a simple java program for implementing the two dimensional array program and its square. This session provide you the best explanation with mathematical operation.

**Description of progarm:**

We are going to display the square of two matrix. Firstly, we have to define a class "**SquareMatrix**".  Then we take an integer type array that contains integer type values. After this, we use two **'for'** loop that denotes rows and columns of a matrix. After getting both the matrix with us we need to square both matrix. When we go to square this array then we use "**square[i][j] =square[i][j] \* square[i][j]**". So, the output will be displayed on the screen command prompt by using the **println()** method.

**Here is the code of this program**

|  |
| --- |
| **public class**SquareMatrix {   **public static void**main(String[] args) {   **int**square[][]= {{1,3,5},{2,4,6}};   System.out.println("Your Original Matrix: ");   **for**(**int**i = 0; i < 2; i++){   **for**(**int**j = 0; j < 3; j++){   System.out.print(square[i][j] + " ");   }   System.out.println();   }   **for**(**int**i = 0; i <= 1; i++) {   **for**(**int**j = 0; j <= 2; j++) {   square[i][j] = square[i][j] \* square[i][j];   }    }   System.out.println("Matrix after changes: ");   **for**(**int**i = 0; i < 2; i++){   **for**(**int**j = 0; j < 3; j++){   System.out.print(square[i][j] + " ");   }   System.out.println();   }   } } |

[**Download this Example**](http://www.roseindia.net/java/beginners/SquareMatrix.java)

# Check Even-Odd:-Example

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/SquareMatrix.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Break.shtml)**

Now, you have to learn about the syntax of If - Else construct. How the If - Else construct used to flow the program control as for as needed. If - Else construct has illustrated by a given example.

Given example reads a integer value and check weather number is Odd or Even. In this example you will learn how to determine the Even or Odd for the given number entered through the keyboard. If the given condition is true then print the message Given number is Even otherwise the control of the program will jumps in the else block and print the message Given number is Odd. In this example a special exception has been also used, that is *NumberFormatException* which holds the error during the checking the entered data format. Entered data must be a number not a character or string. If you enter anything except numeric value then normal flow of the program is sent to the catch block and print the specified message. Full running code is provided with the example.

**Here is the code of the program:-**

|  |
| --- |
| **import**java.io.\*;  **public class**IfElse{   **public static void**main(String[] args) **throws**IOException{   **try**{   **int**n;   BufferedReader in =  **new**BufferedReader(**new**InputStreamReader(System.in));   n = Integer.parseInt(in.readLine());   **if**(n % 2 == 0)   {   System.out.println("Given number is Even.");   }   **else**   {   System.out.println("Given number is Odd.");   }   }   **catch**(NumberFormatException e){   System.out.println(e.getMessage()  + " is not a numeric value.");   System.exit(0);   }   } } |

[**Download Even-Odd Example**](http://www.roseindia.net/java/beginners/IfElse.java)

# Java - Break statement in java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/IfElse.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Continue.shtml)**

The java programming language supports the following types of controlling statements such as:  
  **1.The break statement   
  2.The continue statement  
  3.The return statement**

**Break:**The break statement is used in many programming languages such as c, c++, java etc. Some times we need to exit from a loop before the completion of the loop then we use break statement and exit from the loop and loop is terminated. The break statement is used in while loop, do - while loop, for loop and also used in the switch statement.

**Code of the program :**

|  |
| --- |
| **public class**Break{   **public static void**main(String[] args){   **int**i,j;   System.out.println("Prime numbers between 1 to 50 : ");   **for**(i = 1;i < 50;i++ ){   **for**(j = 2;j < i;j++ ){   **if**(i % j == 0)   {     **break**;   }   }   **if**(i == j)   {   System.out.print("  " + i);   }   }   } } |

**Output of the program :**

|  |
| --- |
| **C:\chandan>javac Break.java  C:\chandan>java Break The Prime number in between 1 - 50 : 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47** |

# Java - Continue statement in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Break.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/switch.shtml)**

**Continue:**The continue statement is used in many programming languages such as C, C++, java etc. Sometimes we do not need to execute some statements under the loop then we use the continue statement that stops the normal flow of the control and control returns to the loop without executing the statements written after the continue statement. There is the difference between break and continue statement that the break statement exit control from the loop but continue statement keeps continuity in loop without executing the statement written after the continue statement according to the conditions.

In this program we will see that how the continue statement is used to stop the execution after that.

**Here is the code of the program :**

|  |
| --- |
| **public class**Continue{   **public static void**main(String[] args){   Thread t = **new**Thread();   **int**a = 0;   **try**{   **for**(**int**i=1;i<10;i++)     {     **if**(i == 5)   {   **continue**;   //control will never reach here (after the continue statement).   //a = i;   }   t.sleep(1000);   System.out.println("chandan");   System.out.println("Value of a : " + a);   }   }   **catch**(InterruptedException e){}   } } |

**Output of the program :**

If we write the code in the given program like this :   
if (i == 5 )  
{  
  continue;  
  a = i;  
}

Then the program will generate a error on compile time like :

|  |
| --- |
| **C:\chandan>javac Continue.java Continue.java:12: unreachable statement a = i; ^ 1 error** |

If we write the code in the given program like this :   
if (i == 5 )  
{  
  continue;  
}

Then the program prints the output like :

|  |
| --- |
| **C:\chandan>javac Continue.java**  **C:\chandan>java Continue chandan Value of a : 0 chandan Value of a : 0 chandan Value of a : 0 chandan Value of a : 0 chandan Value of a : 0 chandan Value of a : 0 chandan Value of a : 0 chandan Value of a : 0** |

# Java - The switch construct in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Continue.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/SwitchExample.shtml)**

Switch is the control statement in java which also turns the normal flow control of the program as per conditions. It works same as If-Else construct. This is the difference between Switch and If-Else construct that switch is used for reduce the if statements. If the multiple choices are available then we generally use the If-Else construct otherwise Switch is easier than the If-Else construct. Switch checks your choice and jump on that case label if the case exists otherwise control is sent to the default label.

In this Program you will see that how to use the switch statement. This program take a number and check weather the number lies between 1 to 7. If user enters the number between 1 to 7 then program print the name of the day in sequence like for 1 prints Sunday, for 2 prints Monday and so on otherwise prints the message Invalid entry!. If user enters any character then the message will be printed by catchblock. This program also using the break statement. The normal flow of control quits from the Switch block whenever break statement occurs. Full running program code is provided with the example.

**Here is a code of program:-**

|  |
| --- |
| **import**java.io.\*;  **public class**Switch{   **public static void**main(String args[]) **throws**Exception{   **int**ch;   System.out.println("Enter 1 for Sunday.");   System.out.println("Enter 2 for Monday.");   System.out.println("Enter 3 for Tuesday.");   System.out.println("Enter 4 for Wednesday.");   System.out.println("Enter 5 for Thrusday.");   System.out.println("Enter 6 for Friday.");   System.out.println("Enter 7 for Saturday.");   System.out.print("your choice is : ");   BufferedReader in =  **new**BufferedReader(**new**InputStreamReader(System.in));   **try**{   ch=Integer.parseInt(in.readLine());   **switch**(ch){       **case**1:  System.out.println("Sunday");   **break**;   **case**2:  System.out.println("Monday");   **break**;   **case**3:  System.out.println("Tuesday");   **break**;   **case**4:  System.out.println("Wednesday");   **break**;   **case**5:  System.out.println("Thrusday");   **break**;   **case**6:  System.out.println("Friday");   **break**;   **case**7:  System.out.println("Saturday");   **break**;   **default**: System.out.println("Invalid entry!");    **break**;   }   }   **catch**(NumberFormatException ex){   System.out.println(ex.getMessage()  + " is not a numeric value.");   System.exit(0);   }   } } |

[**Download The Switch Example**](http://www.roseindia.net/java/beginners/Switch.java)

# While Loop

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/SwitchExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DoWhile.shtml)**

Loop is the control statement of any language in which whenever you want to perform the repetitious work then you use the **Loop control statement**. There are mainly three types of loops. Loop repeats a statement or a process multiple times according to the specified conditions. It allows the multiple statements or process to be run for the specified time or it also follows the certain conditions. Loop makes your program readable, flexible and reliable.

**While loop**: While loop checks the certain condition first, if the condition is true then all the statements or processes written under the while loop are executed otherwise ignored all.

**The Syntax for the while loop:**  
  while(condition)  
  {  
  statements;  
   }

In this program you will see how to use the while loop statement. This program take the value of variable n and generate the table of that number which is 2.  
  
**Here is a code of program**:-

|  |
| --- |
| **public class**table{   **public static void**main(String[] args){   **int**n = 2;   **int**i = 1;   System.out.println("The table of "+n+" = ");   **while**(i<=10)   {   **int**t = n \* i;   System.out.println(t);   i++;   }   } } |

[**Download this example.**](http://www.roseindia.net/java/beginners/while-loop.java)

**Related Tags for While Loop:**

# do-while Loop in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/WhileLoop.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/ForLoop.shtml)**

In some conditions in java, you need to run some statements and processes once at least. For that you have to use the do while loop in  after that check the condition then we use the do-while loop statement. do-while loop is same as the while loop statement but while loop checks the certain condition is first and if condition is true then statements or processes are executed otherwise all the statements written under the while loop is ignored by the interpreter but do - while loop executes all the statements first at once and then check the condition if the condition is true then all the statements are also executed in second times otherwise second times ignored all the statements.

**The Syntax for the do-while loop:**

  do  
  {  
  statements;  
  }  
  while(condition);

In this example you will see how to use the do-while loop statement. The values are already given the variable n and r in the program after that it calculate the program and its perform the its original number and its reverse number.

**Here is the code of the program:-**

|  |
| --- |
| **public class**DoWhile{   **public static void**main(String[] args){   **int**n = 12345;   **int**t,r = 0;   System.out.println("The original number : " + n);   **do**{   t = n % 10;   r = r \* 10 + t;   n = n / 10;   }**while**(n > 0);   System.out.println("The reverse number : " + r);   } } |

# For Loop in Java - Java For Loop Examples & Syntax

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/DoWhile.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/ObjectClass.shtml)**

The for loop is the type of  looping construct. It also works as while loop construct but it provide the initialization, condition and the increment  is same written in the for construct. All the statements which has to be executed written in the for block. We can use the multiple for loop in a program or a for loop also. When we write a loop under another loop then the second loop is called nested loop.

**The Syntax for the For Loop:**

  for( initialization; termination; increment)  
  {  
  statements;  
  }

The given example illustrates that how to use the for loop for developing a application or a program. In this program we will see that the initialization of the variable, condition and increment used for the for loop. Initialization, Condition and Incrementation, all are optional but two semicolons are compulsory which separates Initialization, Condition and Incrementation.

**Initialization:** It allows the variable to be initialize. Such as:   
  int i = 1;  
  int j = 1;  
**Termination (or condition):** It allows to check the certain condition. If condition is true then all statements and processes written in the for block will be executed otherwise ignored. Condition such as:  
  i <= 5;  
  j <= i;  
**Increment:** It allows the how much increase the given variable. Such as:  
  i++;  
  j++;

**Code of the program :**

|  |
| --- |
| **public class**ForLoop{   **public static void**main(String[] args){   **for**(**int**i = 1;i <= 5;i++){   **for**(**int**j = 1;j <= i;j++){   System.out.print(i);   }   System.out.println();   }   } } |

**Output of the program :**

1  
22  
333  
4444  
55555

# Java - Class, Object and Methods in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ForLoop.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Construct.shtml)**

**Class :**Whatever we can see in this world all the things are a object. And all the objects are categorized in a special group. That group is termed as a class. All the objects are direct interacted with its class that mean almost all the properties of the object should be matched with it's own class. Object is the feature of a class which is used for the working on the particular properties of the class or its group. We can understand about the class and object like this : we are all the body are different - different objects of the human being class. That means all the properties of a proper person relates to the human being. Class has many other features like creation and implementation of the object, Inheritance etc.

In this Program you will see how to use the class, object and it's methods. This program uses the several values of several defined variables for getting the Area and Perimeter of the Square and Rectangle  by calling the different functions from the different classes through the several objects created for the several class. In this program there are two classes has been used except the main class in which the main function is declared. First class is square which is using for getting the Area and Perimeter of the square and another class is rectangle which is using for getting the Area and Perimeter of the Rectangle. All the functions in the square and rectangle class are calling with different - different arguments two times for getting the Area and Perimeter of square and rectangle for two different sides. This program gives us the Area and Perimeter for the different sided Square and Rectangle separately. Full running code is given with the example :

**Here is the code of the program :**

|  |
| --- |
| **class**square{   **int**sqarea(**int**side){   **int**area = side \* side;   **return**(area);   }   **int**sqpari(**int**side){   **int**pari = 4 \* side;   **return**(pari);   } } **class**rectangle{   **int**rectarea(**int**length,int breadth){   **int**area = length \* breadth;   **return**(area);   }   **int**rectpari(**int**length,int breadth){   **int**pari = 2\*(length + breadth);   **return**(pari);   } } **public class**ObjectClass{   **public static void**main(String args[]){   **int**sarea1,sarea2,pari1,pari2;   **int**rectarea1,rectarea2,rectpari1,rectpari2;   square sq = **new**square();   rectangle rect = **new**rectangle();   **int**a=20;   System.out.println("Side of first square = " + a);   sarea1 = sq.sqarea(a);   pari1 = sq.sqpari(a);   System.out.println("Area of first square = " + sarea1);   System.out.println("Parimeter of first square = " + pari1);   a = 30;     System.out.println("Side of second square = " + a);   sarea2 = sq.sqarea(a);   pari2 = sq.sqpari(a);   System.out.println("Area of second square = " + sarea2);   System.out.println("Parimeter of second square = " + pari2);   **int**x = 10, y = 20;   System.out.println("Length of first Rectangle = " + x);     System.out.println("Breadth of first Rectangle = " + y);   rectarea1 = rect.rectarea(x,y);   rectpari1 = rect.rectpari(x,y);   System.out.println("Area of first Rectangle = " + rectarea1);   System.out.println("Parimeter of first Rectangle = " + rectpari1);   x = 15;   y = 25;   System.out.println("Length of second Rectangle = " + x);   System.out.println("Breadth of second Rectangle = " + y);   rectarea2 = rect.rectarea(x,y);   rectpari2 = rect.rectpari(x,y);   System.out.println("Area of second Rectangle = " + rectarea2);   System.out.println("Parimeter of first Rectangle = " + rectpari2);   } } |

**Descriptions of the program:**

**Object :** Objects are the basic run time entity or in other words object is a instance of a class . An object is a software bundle of variables and related methods of the special class. In the above example the sq is the object of square class and rect is the object of the rectangle class. In real-world objects share two characteristics: They have all state and behavior. For example, The squares have state such as : sides and behaviors such as its areas and perimeters. Rectangles have state such as: length, breadth and behavior such as its areas and perimeters. A object implements its behavior with methods of it's classes. A method is a function (subroutine) associated with an object.

**Syntax for the Object :**

*class\_name* *object\_name* = new *class\_name*();

**Output of the program :**

|  |
| --- |
| C:\chandan>javac ObjectClass.java  C:\chandan>java ObjectClass Side of first square = 20 Area of first square = 400 Parimeter of first square = 80 Side of second square = 30 Area of second square = 900 Parimeter of second square = 120 Length of first Rectangle = 10 Breadth of first Rectangle = 20 Area of first Rectangle = 200 Parimeter of first Rectangle = 60 Length of second Rectangle = 15 Breadth of second Rectangle = 25 Area of second Rectangle = 375 Parimeter of first Rectangle = 80 |

# Java - Constructor in java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ObjectClass.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/constructoroverloading.shtml)**

Every class has at least one it's ownconstructort. Constructor creates a instance for the class. Constructor initiates (initialize) something related to the class's methods. Constructor is the method which name is same to the class. But there are many difference between the method (function) and the Constructor.

In this example we will see that how to to implement the constructor feature in a class. This program is using two classes. First class is another and second is the main class which name is Construct. In theConstruct class two objects (a and b) are created by using the overloaded another Constructor by passing different arguments and calculated the are of the different rectangle by passing different values for the another constructor.

**Here is the code of the program :**

|  |
| --- |
| **class**another{   **int**x,y;   another(**int**a, **int**b){   x = a;   y = b;   }   another(){   }   **int**area(){   **int**ar = x\*y;   **return**(ar);   } } **public class**Construct{    **public static void**main(String[] args)  {  another b = **new**another();  b.x = 2;  b.y = 3;  System.out.println("Area of rectangle : " + b.area());  System.out.println("Value of y in another class : " + b.y);  another a = **new**another(1,1);  System.out.println("Area of rectangle : " + a.area());  System.out.println("Value of x in another class : " + a.x);  } } |

**Output of the program :**

|  |
| --- |
| **C:\chandan>javac Construct.java  C:\chandan>java Construct Area of rectangle : 6 Value of x in another class : 3 Area of rectangle : 1 Value of x in another class : 1** |

[**Download this example.**](http://www.roseindia.net/java/beginners/Construct.java)

# Constructor Overloading in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Construct.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/useofthisoperator.shtml)**

Here, you will learn more about**Constructor** and how constructors are overloaded in Java. This section provides you a brief introduction about the**Constructor** that are overloaded in the given program with complete code absolutely in running state i.e. provided for best illustration about the constructor overloading in Java.

Constructors are used to assign initial values to instance variables of the class. A default constructor with no arguments will be called automatically by the Java Virtual Machine (JVM). Constructor is always called by new operator.Constructor are declared just like as we declare methods, except that the constructor don't have any return type. Constructor can be overloaded provided they should have different arguments because JVM differentiates constructors on the basis of arguments passed in the constructor.

Whenever we assign the name of the method same as  class name. Remember this method should not have any return type. This is called as constructor overloading.

We have made one program on a constructor overloading, after going through it the concept of constructor overloading will get more clear. In the example below we have made three overloaded constructors each having different arguments  types so that the JVM can differentiates between the various constructors.

**The code of the program is given below:**

|  |
| --- |
| **public class**ConstructorOverloading{   **public static void**main(String args[]){   Rectangle rectangle1=**new**Rectangle(2,4);   **int**areaInFirstConstructor=rectangle1.first();   System.out.println(" The area of a rectangle in    first constructor is :  " + areaInFirstConstructor);   Rectangle rectangle2=**new**Rectangle(5);   **int**areaInSecondConstructor=rectangle2.second();   System.out.println(" The area of a rectangle in     first constructor is :  " + areaInSecondConstructor);   Rectangle rectangle3=**new**Rectangle(2.0f);   **float**areaInThirdConstructor=rectangle3.third();   System.out.println(" The area of a rectangle in first    constructor is :  " + areaInThirdConstructor);   Rectangle rectangle4=**new**Rectangle(3.0f,2.0f);   **float**areaInFourthConstructor=rectangle4.fourth();   System.out.println(" The area of a rectangle in first    constructor is :  " + areaInFourthConstructor);   } }  **class**Rectangle{   **int**l, b;   **float**p, q;   **public**Rectangle(**int**x, **int**y){   l = x;   b = y;   }   **public int**first(){    **return**(l \* b);   }   **public**Rectangle(**int**x){   l = x;   b = x;   }   **public int**second(){   **return**(l \* b);   }   **public**Rectangle(**float**x){   p = x;   q = x;   }   **public float**third(){   **return**(p \* q);   }   **public**Rectangle(**float**x, **float**y){   p = x;   q = y;   }   **public float**fourth(){   **return**(p \* q);   } } |

**Output of the program is given below:**

|  |
| --- |
| **C:\java>java ConstructorOverloading The area of a rectangle in first constructor is : 8 The area of a rectangle in first constructor is : 25 The area of a rectangle in first constructor is : 4.0 The area of a rectangle in first constructor is : 6.0** |

[**Download this program**](http://www.roseindia.net/java/beginners/ConstructorOverloading.java)

# How to use "this" keyword in java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/constructoroverloading.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Threads.shtml)**

The keyword **this**is useful when you need to refer to instance of the class from its method. The keyword helps us to avoid name conflicts. As we can see in the program that we have declare the name of instance variable and local variables same. Now to avoid the confliction between them we use **this** keyword. Here, this section provides you an example with the complete code of the program for the illustration of how to what is **this** keyword and how to use it.

In the example, **this.length** and **this.breadth** refers to the instance variable length and breadth while length and breadth refers to the arguments passed in the method. We have made a program over **this**. After going through it you can better understand.

**Here is the code of the program:**

|  |
| --- |
| **class**Rectangle{   **int**length,breadth;   **void**show(**int**length,int breadth){   **this**.length=length;   **this**.breadth=breadth;   }   **int**calculate(){   **return**(length\*breadth);   } } **public class**UseOfThisOperator{   **public static void**main(String[] args){   Rectangle rectangle=**new**Rectangle();   rectangle.show(5,6);   **int**area = rectangle.calculate();   System.out.println("The area of a Rectangle is  :  " + area);   } } |

**Output of the program is given below:**

|  |
| --- |
| **C:\java>java UseOfThisOperator The area of a Rectangle is : 30** |

# Java - Threads in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/useofthisoperator.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/OccurancesInArray.shtml)**

Thread is the feature of mostly languages including Java. Threads allow the program to perform multiple tasks simultaneously. Process speed can be increased by using threads because the thread can stop or suspend a specific running process and start or resume the suspended processes. Multitasking or multiprogramming is delivered through the running of multiple threads concurrently. If your computer does not have multi-processors then the multi-threads really do not run concurrently.

This example illustrates how to create a thread and how to implement the thread. In this example we will see that the program prints numbers from 1 to 10 line by line after 5 seconds which has been declared in the sleep function of the thread class. Sleep function contains the sleeping time in millisecond and in this program sleep function has contained 5000 millisecond mean 5 second time. There is sleep function must caught by the InterruptedException. So, this program used theInterruptedException which tells something the user if thread is failed or interrupted.

**Here is the code of the program :**

|  |
| --- |
| **public class**Threads{   **public static void**main(String[] args){   Thread th = **new**Thread();   System.out.println("Numbers are printing line by line after 5 seconds : ");   **try**{   **for**(**int**i = 1;i <= 10;i++)     {   System.out.println(i);   th.sleep(5000);   }   }   **catch**(InterruptedException e){     System.out.println("Thread interrupted!");   e.printStackTrace();   }   } } |

[**Download this example.**](http://www.roseindia.net/java/beginners/Threads.java)

# Find in Array - Calculate Occurrences of Elements in Array

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/Threads.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/entervaluesfromkeyboard.shtml)**

This section illustrates you how to calculate occurrences of elements in an array. Occurrences means, the how many times an element occurs in the array. In this section, you will get more and more help from the given following program with the complete code.

**Program Description:**

This program first asks for the length of the array whatever you want to fix and then it takes some inputs for the elements of the array how more you have mentioned for the length of the array. And finally it show the occurrences of each and every element of the array uniquely.

**Here is the code of the program:**

|  |
| --- |
| **import**java.io.\*;  **public class**OccurancesInArray{   **public static void**main(String[] args) **throws**IOException{   BufferedReader in = **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.print("How many elements you want to enter in the array: ");   **int**num=0;   **try**{   num = Integer.parseInt(in.readLine());   }   **catch**(NumberFormatException ne){   System.out.println(ne.getMessage() + " is not a number!");   System.exit(0);   }   String[] elements = **new**String[num];   **int**a;   **int**k;   **for**(**int**i = 0; i < num; i++){   elements[i] = in.readLine();   }   **for**(**int**i = 0; i < elements.length; i++){   a = 0;   k = 1;   **for**(**int**j = 0; j < elements.length; j++){   **if**(j >= i){   **if**(elements[i].equals(elements[j]) && j != i){   k++;   }   }   **else if**(elements[i].equals(elements[j])){   a = 1;   }   }   **if**(a != 1){   System.out.println("Occurance of \'" + elements[i] + "\' : " + k);   }   }   } } |

**Output of the program :**

|  |
| --- |
| **C:\chandan>javac OccurancesInArray.java  C:\chandan>java OccurancesInArray How many elements you want to enter in the array: 5 123 chand 453 aaa 123 Occurance of '123' : 2 Occurance of 'chand' : 1 Occurance of '453' : 1 Occurance of 'aaa' : 1  C:\work\chandan>\_** |

[**Download this example.**](http://www.roseindia.net/java/beginners/OccurancesInArray.java)

# Change the user input to integer

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/OccurancesInArray.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/StringTokenizing.shtml)**

Isn't it a very tedious job to enter the values each time before the compilation in the method itself. Now if we want to enter an integer value after the compilation of a program and force the JVM to ask for an input, then we should use Integer.parseInt(string str).

As we know that JVM reads each data type in a String format. Now consider a case where we to enter a value in  int primitive type, then we have to use **int x = Integer.parseInt("any String")** , because the value we have entered is in integer but it will be interpreted as a String so, we have to change the string into integer. Here **Integer** is a name of a class in **java.lang**package and **parseInt()**is a method of**Integer** class which converts String to integer. If we want to enter a integer in a method or class using keyboard, then we have to use a method **parseInt()**.

In this program we are going to calculate a area of a rectangle by using two classes named Rectangle and EnterValuesFromKeyboard. In the first class we have used two methods, ***show(int x, int y)*** and***calculate()***. First method *show()* is taking two variables as input and second method *calculate()*calculates the area of a rectangle. In the second class which is also our main class we declare a will declare our main method. Inside this method we will create a object of a Rectangle class. Now we ask the user to input two values and stored those values in the variables. These entered values will be changed into integer by using *parseInt()* method. Now these variables are passed in the method *show()*and the area will be calculated by *calculate()* method. These methods will be called by the instance of Rectangle class because it is the method of Rectangle class.

**Here is the code of the program**

|  |
| --- |
| **class**Rectangle{   **int**length, breadth;   **void**show(**int**x, **int**y){   length = x;   breadth = y;   }   **int**calculate(){   **return**(length \* breadth);   } } **public class**EnterValuesFromKeyboard{   **public static void**main(String[] args) {   Rectangle rectangle = **new**Rectangle();   **int**a = Integer.parseInt(args[0]);   **int**b = Integer.parseInt(args[1]);   rectangle.show(a, b);   System.out.println(   " you have entered these values : " +  a  + " and " +  b);   **int**area = rectangle.calculate();   System.out.println(" area of a rectange is  : " + area);   } } |

**Output of the program is given below:**

|  |
| --- |
| **C:\java>java EnterValuesFromKeyboard 4 5 you have entered these values : 4 and 5 area of a rectange is : 20** |

# Making Tokens of a String

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/entervaluesfromkeyboard.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/array_list_demo.shtml)**

Tokens can be used where we want to break an application into tokens. We have to break a String  into tokens as well as we will know how many tokens has been generated. That's what we are trying to do in this program. In the program a string is passed into a constructor of **StringTokenizer** class. **StringTokenizer** is a class in java.util.package. We are using while loop to generate tokens. The following methods have been used in this program.

**countTokens():** It gives the number of tokens remaining in the string.

**hasMoreTokens():** It gives true if more tokens are available, else false.

**nextToken():** It gives the next token available in the string.

To break a string into tokens what we need first is to create a class named **StringTokenizing.** Inside this class we will declare our main method. Make an object of class **StringTokenizer** and pass one string inside the constructor which you want to break into tokens. By using the instance of**StringTokenizer** call the method **countTokens()** which gives the number of tokens remaining in the string. It is a method of  **StringTokenizer** class. If the object have more tokens available then it will call method **hasMoreTokens()** and print the tokens by using **nextToken()**.

**The code of the program is given below:**

|  |
| --- |
| **import**java.util.\*;  **public class**StringTokenizing{   **public static void**main(String[] args) {   StringTokenizer stringTokenizer = **new**    StringTokenizer("You are tokenizing a string");   System.out.println("The total no. of tokens     generated :  " + stringTokenizer.countTokens());   **while**(stringTokenizer.hasMoreTokens()){   System.out.println(stringTokenizer.nextToken());   }   } } |

Output of this program is given below:

|  |
| --- |
| **C:\java>java StringTokenizing The total no. of tokens generated : 5 You are tokenizing a string** |

[**Download this program:**](http://www.roseindia.net/java/beginners/StringTokenizing.java)

# Array List Example in java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/StringTokenizing.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/construct_file_name_path.shtml)**

In this example we are going to show the use of **java.util.ArrayList.**We will be creatiing an object of ArrayList class and performs various operations like adding removing the objects.

**Arraylist**provides methods to manipulate the size of the array that is used internally to store the list.**ArrayList**extends **AbstractList** and implements ***List, Cloneable, Serializable***.  **ArrayList** capacity .grows automatically. The **ArrayList** is not synchronized. It permits all elements including null.

In this program we are inserting a value. We are using three methods of **ArrayList**class.

**add(Object o):** Appends the specified element to the end of this list. It returns a boolean value.

**size():** Returns the number of elements in this list.

**remove(int index):** Removes the element at the specified position in this list. It returns the element that was removed from the list. It throws **IndexOutOfBoundsException :**if index is out of range.

**Code of a program is given below:**

|  |
| --- |
| **import**java.util.\*;  **public class**ArrayListDemo{   **public static void**main(String[] args) {   ArrayList<Object> arl=**new**ArrayList<Object>();   Integer i1=**new**Integer(10);   Integer i2=**new**Integer(20);   Integer i3=**new**Integer(30);   Integer i4=**new**Integer(40);   String s1="tapan";   System.out.println("The content of arraylist is: " + arl);   System.out.println("The size of an arraylist is: " + arl.size());   arl.add(i1);   arl.add(i2);   arl.add(s1);   System.out.println("The content of arraylist is: " + arl);   System.out.println("The size of an arraylist is: " + arl.size());   arl.add(i1);   arl.add(i2);   arl.add(i3);   arl.add(i4);   Integer i5=**new**Integer(50);   arl.add(i5);   System.out.println("The content of arraylist is: " + arl);   System.out.println("The size of an arraylist is: " + arl.size());   arl.remove(3);   Object a=arl.clone();   System.out.println("The clone is: " + a);    System.out.println("The content of arraylist is: " + arl);   System.out.println("The size of an arraylist is: " + arl.size());   } } |

**The output of program will be like this:**

|  |
| --- |
| **C:\Java Tutorial>javac ArrayListDemo.java  C:\Java Tutorial>java ArrayListDemo The content of arraylist is: [] The size of an arraylist is: 0 The content of arraylist is: [10, 20, tapan] The size of an arraylist is: 3 The content of arraylist is: [10, 20, tapan, 10, 20, 30, 40, 50] The size of an arraylist is: 8 The clone is: [10, 20, tapan, 20, 30, 40, 50] The content of arraylist is: [10, 20, tapan, 20, 30, 40, 50] The size of an arraylist is: 7  C:\Java Tutorial>** |

# Converting a Filename to a URL

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/array_list_demo.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/createtemporaryfile.shtml)**

A file object is used to a give a filename. Creating the **File** object doesn't mean that a file exists. It may be that the does not exist. Suppose if the file exists, first of all we need to convert the file object in URL, for this we use a method **toURL()**. It returns a**URL** object and throws **MalformedException**. After this we will convert this **URL** to a file object by using **getFile()** method. We will read this file by using **BufferedReader** object.

**toURL() :** It is used to convert the file name into the URL.

**getFile() :** This is the method of the **URL** class, is used to get the file name from the URL.

**Here is the code of the program:**

|  |
| --- |
| **import**java.io.\*; **import**java.net.\*;  **public class**ConstructFileNamePath{   **public static void**main(String[] args){   File file=**new**File("C:/work/chandan/deepak.txt");   URL url=**null**;   **try**{   //The file may or may not exist   url=file.toURL(); //file:/C:/work/chandan/deepak.txt   System.out.println("The url is" + url);    // change the URL to a file object   file=**new**File(url.getFile());  // c:/work/chandan/deepak.txt   System.out.println("The file name is " + file);   **int**i;      //opens an input stream   InputStream is=url.openStream();   BufferedReader br=**new**BufferedReader(**new**InputStreamReader(is));   **do**{   i=br.read();   System.out.println((**char**)i);   }**while**(i!=-1);   is.close();   }   **catch**(MalformedURLException e){   System.out.println("Don't worry,exception has been caught" + e);   }   **catch**(IOException e){   System.out.println(e.getMessage());   }     } } |

**The output of this program is given below:**

|  |
| --- |
| **C:\ConstructFileNamePath>java ConstructFileNamePath The url isfile:/C:/ConstructFileNamePath/ConstructFileNamePath/ConstructFileNamePath.txt The file name is C:\ConstructFileNamePath\ConstructFileNamePath\ConstructFileNamePath.txt C:\ConstructFileNamePath\ConstructFileNamePath\ConstructFileNamePath.txt (The system cannot find the path specified)** |

# Java Temporary File - Temporary File Creation in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/construct_file_name_path.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/date.shtml)**

In this program we are going to make a temporary file, which will be deleted automatically by the garbage collector when the program ends.

In this program we are using following methods:

**createTempFile(String prefix, String suffix):** This is a static method of file class. This creates an empty file and we have used two parameters to generate name.

**write():** It will write in String.

**close():** When we don't need to write further, we should close the program with() that will close the string.

**deleteOnExit():** It will delete the existing file when you close the java program or your JVM.

**Code of this program is given below:**

|  |
| --- |
| **import**java.io.\*;  **public class**CreateTemporaryFile{   CreateTemporaryFile(){   }   **public static void**CreateTempFile(){   **try**{   // Create a temporary file object   File tempFile = File.createTempFile("prefix", "suffix");   System.out.println("\nTemporary file file has  been created : " + tempFile + "\n");   // Write to temporary file   BufferedWriter out = **new**BufferedWriter(**new**  FileWriter(tempFile));   out.write("You are writing on temporary file  which will delete on exit : " + tempFile);   out.close();  // Delete temp file when program exits   tempFile.deleteOnExit();   }    **catch**(IOException e){   System.out.println("Exception is" + e);   }   }   **public static void**main(String[] args){   CreateTempFile();   }  } |

**The output of this program is given below:**

|  |
| --- |
| C:\ CreateTemporaryFile>java CreateTemporaryFile  Temporary file file has been created : C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\prefix3 2276suffix   C:\tapan> |

[**Download this example**](http://www.roseindia.net/java/beginners/CreateTemporaryFile.java)

# Example of Date class

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/createtemporaryfile.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/file-name-refer-same-file.shtml)**

We can use Date class to use current date. In this program we are creating a **Date**object and by using it we are going to display the current date and time. It implements ***Serializable, Cloneable, Comparable***interfaces. Its known subclasses are **Date, Time, Timestamp**.

**Code of a program is given below:**

|  |
| --- |
| **import**java.util.\*;  **public class**DateDemo{   **public static void**main(String[] args) {   Date d=**new**Date();   System.out.println("Today date is "+ d);   } } |

**Output of this program is given below:**

Today date is Tue Dec 19 22:14:46 GMT+05:30 2006

# Determining if two Filename paths refer to the same file.

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/date.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/file-directory.shtml)**

This program is going to determine that the file name refers to the same file or not. What you need to do is to create two **File**object and pass the file name in the constructor of both the **File** class. Now match both the file instances. It will return false if the file paths are not equal. To send the true value we first have to normalize both the paths, only after that  it will return true.

In this program we have used following methods:

**equals():**This is the method of **String** class. It checks whether the pathname are equal or not.

**getCanonicalFile():**It denotes the same file or directory its abstract pathname. It throws **IOException** and**SecurityException.**

**Code of the program is given below:**

import java.io.\*;  
  
public class FileNameReferToSameFile{  
public static void main(String args[]){  
File file1 = new File("./filename.txt");  
File file2 = new File("filename.txt");  
System.out.println("Actual fileName1 = " + file1 + "\n");  
System.out.println("Actual fileName2 = " + file2 + "\n");  
  
// It returns false if Filename paths are not equal   
boolean b = file1.equals(file2);   
System.out.println("It checks whether the file name   
  
paths are equal or not" + b);  
  
  
  
// Normalize the paths  
try {  
file1 = file1.getCanonicalFile(); // c:\tapan\filename  
file2 = file2.getCanonicalFile(); // c:\tapan\filename  
System.out.println("Actual path of filName1 = " + file1 + "\n");  
System.out.println("Actual path of fileName2 = " + file2 + "\n");  
} catch (IOException e) {  
System.out.println("IOException is "+ e);  
}  
  
// It returns true if Filename paths are equal  
b = file1.equals(file2);   
System.out.println("the file name are now equal" + b);  
System.out.println("Actual path of fileName1 = " + file1 + "\n");  
System.out.println("Actual path of fileName2 = " + file2 + "\n");  
}  
}

**Output of this program is given below:**

|  |
| --- |
| **C:\FileNameReferToSameFile>java FileNameReferToSameFile Actual Filename1 = .\FileNameReferToSameFile.txt  Actual Filename2 = FileNameReferToSameFile.txt  It checks whether the file name are equal or notfalse Actual Filename1 = C:\FileNameReferToSameFile\FileNameReferToSameFile.txt  Actual Filename2 = C:\FileNameReferToSameFile\FileNameReferToSameFile.txt  the file name are now equaltrue Actual Filename1 = C:\FileNameReferToSameFile\FileNameReferToSameFile.txt  Actual Filename2 = C:\FileNameReferToSameFile\FileNameReferToSameFile.txt** |

[**Download this example:**](http://www.roseindia.net/java/beginners/FileNameReferToSameFile.java)

# Determining if a Filename path is a file or a directory

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/file-name-refer-same-file.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/file-directory-exists.shtml)**

You can solve this problem very easily. What you need to do, just create a **File**object and pass some file name or a directory in it.

We are using a following methods to solve this problem.

**isDirectory():** It checks whether the file is a directory or not.

**getAbsolutePath():**It returns the absolute path of a file or directory.

**Code of the program is given below:**

|  |
| --- |
| **import**java.io.\*;  **public class**FileOrDirectory{   **public static void**main(String args[]){   File directory = **new**File("Enter any    directory name or file name");   **boolean**isDirectory = directory.isDirectory();   **if**(isDirectory) {   // It returns true if directory is a directory.   System.out.println("the name you have entered     is a directory  : "  +  directory);     //It returns the absolutepath of a directory.   System.out.println("the path is "  +      directory.getAbsolutePath());   } **else**{   // It returns false if directory is a file.   System.out.println("the name you have    entered is a file  : " + directory);   //It returns the absolute path of a file.   System.out.println("the path is "  +       directory.getAbsolutePath());   }   } } |

**The output of this example is given below:**

|  |
| --- |
| **C:\FileOrDirectory>java FileOrDirectory the name you have entered is a file : FileOrDirectory the path is C:\FileOrDirectory\FileOrDirectory** |

[**Download this example**](http://www.roseindia.net/java/beginners/FileOrDirectory.java)**:**

# Reading a File into a Byte Array

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/list-system-roots.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/serializable-singleton.shtml)**

In this program we will read a file in the form of a byte array. We can do it very easily. Firstly we will create an object of **File** class and pass the name of the file which we want to read, in the constructor of **File**class. We will use **DataInputStream** to read primitive data types.

We are using the following methods:

**length():** It gives the length of the file.

**Getting the file size code:**[**ReadInByteArray.java**](http://www.roseindia.net/java/beginners/ReadInByteArray.java)

|  |
| --- |
| **import**java.io.\*;  **public class**ReadInByteArray  { **public static void**main(String[] args)   {  **try**  {  File file = **new**File("tapan.txt");  **int**size = (**int**)file.length();//It gives the length of the file.     System.out.println("Size of the file: " + size);    }   **catch**(Exception e)   {   System.out.println("Exception has been thrown :" + e);   }   } } |

[**Download this program**](http://www.roseindia.net/java/beginners/ReadInByteArray.java)

The following code helps you in getting the file size in byte array.

**Here is the Byte Array code:**[**ByteArrayExample.java**](http://www.roseindia.net/java/beginners/ByteArrayExample.java)

|  |
| --- |
| **import**java.io.\*;  **public class**ByteArrayExample{ **public static void**main(String[] args) {   **try**{ BufferedReader bf = **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.println("Enter File name: ");   String str = bf.readLine();   File file = **new**File(str);    //File length   **int**size = (**int**)file.length();    **if**(size > Integer.MAX\_VALUE){   System.out.println("File is to larger");   }   **byte**[] bytes = **new byte**[size];  DataInputStream dis = **new**DataInputStream(**new**FileInputStream(file));    **int**read = 0;   **int**numRead = 0;   **while**(read < bytes.length && (numRead=dis.read(bytes, read,   bytes.length-read)) >= 0) {   read = read + numRead;   }   System.out.println("File size: " + read);   // Ensure all the bytes have been read in   **if**(read < bytes.length) { System.out.println("Could not completely read: "+file.getName());   }   }   **catch**(Exception e){   e.getMessage();   }   } } |

# Implementing a Serializable Singleton

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/read-byte-array.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/stack-demo.shtml)**

In Singeton classes only one instance will be created. We are going to serialize the class. This can be done very easily. What we need to do is to implement a***java.io.Serializable*** interface. We will use a method of ***Serializable*** interface that is **readResolve().**

**readResolve():** It returns **Object**and throw **ObjectStreamException**

**Code of the program is given below:**

|  |
| --- |
| **import**java.io.\*;  **public class**SerializableSingleton **implements**java.io.Serializable {     **static**SerializableSingleton singleton;  **protected**SerializableSingleton()    {   // Exists only to thwart instantiation.   }   **private**Object readResolve()    {  **return**instancd;   }   **public static void**main(String args[])   {   singleton = **new**SerializableSingleton();   singleton.readResolve();   } } |

[**Download this progr**](http://www.roseindia.net/java/beginners/SerializableSingleton.java)

# Calculating the Checksum of the file/CRC32

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ChecksumAdler32.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/CompressingFile.shtml)**

A **Checksum** is used for error checking while transferring a file. Data flow across the network in the form of packets. So, checksum is a computed value that is dependent on the contents of  a file. For each packet the computed checksum will  be different. This computed value is transmitted with the packet when it is transmitted. The receiving system checks the checksum and on the basis of checksum it receives and rejects the packet. It is mainly used where it becomes necessary to check the packets before accepting it.

**CRC:**  CRC means **Cyclic Redundancy Check**. It is a error checking technique used to check the accuracy of data while transmitting it to the other end. Errors are checked while the data is transferring. CRC performs some type of calculation before transmitting the data and send the result to the other end. The other end repeats the same operation before accepting the data. If both the devices get the result same, it means the transmission is error free.

To make a program on Checksum first of all make a class named CheckSumCRC32. Inside the class declare one method checkSum(String file) which will take a value of a file which has to pass at run time. Now make a object of **FileInputStream**, **CheckedInputStream** and **CRC32** and pass the instance of**FileInputStream,** **CRC32** into the constructor of **CheckedInputStream**class. To calculate the size of the file call the method length() of File class. We have define a array of type byte, the size of the array is 100, i.e. the size of each packet. The Checksum for each packet will be generated randomly by the CheckedInputStream class. It returns the long data type. Now define a main method inside which we will call checkSum() method which will give us the checksum, size of the file and name of the file.  
To achieve the desired result we have used the following classes and methods.

**FileInputStream:** It is a class of java.io package. It extends InputStream class. It is used for reading in byte form.

**CheckedInputStream**: It is a input stream that keeps the checksum. Its constructor use two parameter, first is InputStream and second is Checksum.

**CRC32:** It is a class of java.util.zip package. It implements ***Checksum***. This class is used to calculate the CRC-32 of the stream.

**length():**It is a method of File class. It returns the length of the file.

**read():** It reads a byte.

**getChecksum():** It returns the Checksum.

**getValue():** It is a method of Checksum interface. It returns the checksum value.

**The code of the program is given below.**

|  |
| --- |
| **import**java.io.\*; **import**java.util.zip.CheckedInputStream; **import**java.util.zip.CRC32;  **public class**ChecksumCRC32 {   **public void**checkSum(String file){   **try**{   FileInputStream fis = **null**;   CheckedInputStream cis = **null**;   CRC32 crc = **null**;   **long**sizeOfFile = 0;   **try**{   fis = **new**FileInputStream(file);     crc = **new**CRC32();   cis = **new**CheckedInputStream(fis, crc);   sizeOfFile = **new**File(file).length();  }   **catch**(Exception e){   System.out.println("File  Not found ");   System.exit(1);   }   **byte**[] buffer = **new byte**[100];   **while**(cis.read(buffer)>=0){   **long**checksum = cis.getChecksum().getValue();   System.out.println(checksum + " " + sizeOfFile + " " + file);   }   }   **catch**(IOException e){   System.out.println("the exception has been thrown" + e);   System.exit(1);   }   }   **public static void**main(String[] args) {   ChecksumCRC32 crc = **new**ChecksumCRC32();   **if**(args.length!= 1){   System.out.println("Please enter the valid file name : " );   }   **else**{   crc.checkSum(args[0]);   }   } } |

The output of the program is given below.

In the output we can see that first we have created a file javalearner.txt in the java folder and pass some information in it. We can see that for each packet a checksum is calculated. The size of each packet is 100 bytes. It can be changed. The output will show 100 checksum value as 10 packets are generated. In the output firstly we are displaying the checksum of the packet, secondly the size of the file and lastly the name of the file.

|  |
| --- |
| **C:\java>java ChecksumCRC32  javalearner.txt 566064419   958  javalearner.txt 3839930655   958  javalearner.txt 3892475745   958  javalearner.txt 3357861592   958  javalearner.txt 4227549807   958  javalearner.txt 459090179   958  javalearner.txt 3273102972   958  javalearner.txt 1119235310   958  javalearner.txt 2705113445   958  javalearner.txt 695211703   958  javalearner.txt** |

# Constructing a File Name path

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CompressingFile.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/firstarrayprogram.shtml)**

This Java programming tutorial will teach you how you can construct a file name path. By using the constructing filename path it is possible to set dynamic path, which is helpful for mapping local  file name with the actual path of the file.

Java API has provided us many packages, one of them is **java.io**  package. This package contains a**File** class  In this example we are using one **static final** variable of File class i.e.. **separatorChar**. The value of this separator is system dependent.  If we are using Windows platform then the value of this separator is  ' \ ' .

**The code of the program is given below:**

|  |
| --- |
| **import**java.io.\*;  **public class**ConstructingFileNamePath {   **public static void**main(String[] args){     String filepath = File.separatorChar  + "java" + File.separatorChar + "example";   System.out.println("The path of the file is  :  " + filepath);      } } |

Output of this program is given below:

|  |
| --- |
| **C:\java>java ConstructingFileNamePath The path of the file is : \java\example** |

# Use of Array in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/constructingfilenamepath.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/howtoaccessstaticmethod.shtml)**

This Java programming example will teach you the the concepts of arrays so that you can write a program on array by yourself. An array works a container object which can hold a values of same type. The length of an array is created when the array is going to be created. **Arrays** can be created in following ways. One way to create a array is by using **new** operator. e.g.. : **a = new int[5];**. It will create an array of type int which can store 5 integer values. The next way to define array is just same as we have defined an array in the program below.  Array should be used,  where you have to store multiple values of the same type. For example if we have to store months of the year then the better idea is to make an array of it.

In the program below we have defined a class named **FirstArrayProgram**. Now declare a method in which you have to define a array of String type which can hold only string values. In this array we are storing a months of the year. Now we want to retrieve the values of an array, so use for loop to access it.

**Code of the program is given below**

|  |
| --- |
| **public class**FirstArrayProgram{   **public static void**main(String[] args){   String a[]={"January","Febuary","March","April","May","June","July",    "August","September","October","November","December"};   System.out.println("Name of months in a year : ");  **for**(**int**i=0;i<a.length;i++){   System.out.println(a[i]);   }   } } |

The output of this program is given below:

|  |
| --- |
| **C:\java>java FirstArrayProgram Name of months in a year  :**  **January Febuary March April May June July August September October November December** |

[**Download this program**](http://www.roseindia.net/java/beginners/FirstArrayProgram.java)

**Example of static method**Posted on: June 8, 2007 at 12:00 AM

**In this section, you will learn how to define a static method.**

# Example of static method

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/firstarrayprogram.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/JavaUncompress.shtml)**

This Java programming example will teach you the way to define a **static** methods.In java we have two types of methods,**instance methods** and **static methods**. Static methods can't use any instance variables. The this keyword can't be used in a static methods. You can find it difficult to understand when to use a static method and when not to use. If you have a better understanding of the instance methods and static methods then you can know where to use instance method and static method.

A static method can be accessed without creating an instance of the class. If you try to use a non-static method and variable defined in this class then the compiler will say that non-static variable or method cannot be referenced from a static context. Static method can call only other static methods and  static variables defined in the class.

The concept of static method will get more clear after this program. First of all create a class**HowToAccessStaticMethod**. Now define two variables in it, one is instance variable and other is class variable. Make one static method named **staticMethod()** and second named as **nonStaticMethod()**. Now try to call both the method without constructing a object of the class. You will find that only static method can be called this way.

**The code of the program is given below:**

|  |
| --- |
| **public class**HowToAccessStaticMethod{   **int**i;   **static int**j;   **public static void**staticMethod(){   System.out.println("you can access a static method this way");   }   **public void**nonStaticMethod(){   i=100;   j=1000;   System.out.println("Don't try to access a non static method");   }   **public static void**main(String[] args) {   //i=100;         j=1000;   //nonStaticMethod();   staticMethod();   } } |

Output of the program is given below:

|  |
| --- |
| **C:\java>java HowToAccessStaticMethod you can access a static method this way** |

# Uncompressing the file in the GZIP format.

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/howtoaccessstaticmethod.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/nonstaticvariable.shtml)**

This example will help you to understand the concepts of **GZIP**.  Sometimes it happens that when you download a file or any PDF  file you get it in compressed mode. This means that the file is compressed, but can't read it in compressed form so, it needs to be uncompressed. There are various uncompress ion utility program which can be found very easily found on internet. If you are uncompressing a PDF with the extension .gz, then its resultant file will have a **.PDF** extension. Its main advantage is that it will uncompress the PDF file so that you can be able to read it. This utility is a part of **java.util.zip**package which provides classes and methods to compress and decompress the file.

To make a program on uncompressing a file, firstly you need to import java.util.zip package so, that all classes and method of this package can be used, also import java.io package for the input/output operation. Then create a class named **JavaUncompress**. Declare a main method inside the class which will take one String argument i.e. file name. If the user forgets to type the name of the input file which has to be uncompressed, then ask the user to enter the file name. After getting the file  name, we need to uncompress that particular file. For that we need to open the gzip file. Print a message so that you can understand where gzip file is opening. To open the input file firstly create a object of**FileInputStream** class which and pass the instance of **FileInputStream** class into the constructor of**GZIPInputStream** class, which will be created by using new operator. After opening the input file now we need to open the output file so that the data can be transferred from the input file to the output file. Print the message so that you can understand where the output is being written. For that use make a object of **FileOutputStream** class and pass the name of the resultant file in its constructor. For transferring the bytes from the compressed file to the output file declare a array of type byte with the size of 1024 bytes. It can be changed according to the needs of a programmer, now declare one variable of type int which will read the byte from the object of **GZIPInputStream** class. At last close the file and stream.

**GZIPInputStream:** This class is a part of **java.util.zip** package. It extends **DeflaterInputStream**class. This class reads compressed data in the GZIP format.

**FileInputStream:** This class extends **InputStream.** It reads bytes from a file .

FileOutputStream: This class extends OutputStream class. It is a output stream for writing data to a file.

**read():** It returns int. It reads the buffer.

**write(byte[] buf,  int off, int len i):** It is a method of **OutputStream**, which takes three arguments. It is used for writing a array of bytes to the uncompressing  file.

**close():** It closes all the resources occupied by the **GZIPInputStream**

**close():** It closes all the resources occupied by the **OutputStream.**

**The code of the program is given below:**

|  |
| --- |
| **import**java.util.zip.GZIPInputStream; **import**java.io.OutputStream; **import**java.io.FileOutputStream; **import**java.io.FileInputStream; **import**java.io.FileNotFoundException; **import**java.io.IOException;  **public class**JavaUncompress{   **public static void**main(String args[]){   **try**{   //To Uncompress GZip File Contents we need to open the gzip file.....   **if**(args.length<=0){   System.out.println("Please enter the valid file name");   }   **else**{   String inFilename = args[0];   System.out.println("Opening the gzip file........  .................. :  opened");    GZIPInputStream gzipInputStream = **null**;   FileInputStream fileInputStream = **null**;   gzipInputStream = **new**GZIPInputStream(**new**  FileInputStream(inFilename));   System.out.println("Opening the output file..  ........... : opened");   String outFilename = inFilename +".pdf";   OutputStream out = **new**FileOutputStream(outFilename);   System.out.println("Transferring bytes from the  compressed file to the output file........:      Transfer successful");   **byte**[] buf = **new byte**[1024];  //size can be  changed according to programmer's need.   **int**len;   **while**((len = gzipInputStream.read(buf)) > 0) {   out.write(buf, 0, len);   }   System.out.println("The file and stream is ..  ....closing.......... : closed");    gzipInputStream.close();   out.close();   }   }   **catch**(IOException e){   System.out.println("Exception has been thrown" + e);   }   } } |

The output of the program is given below.

In the output we can see that firstly we have given the name of the compressed file. The file name should ends with .gz extension, while opening the compressed gzip file we are printing that the file is opening. After opening the file we are opening a file to which we have to write the data from the compressed file and that also is printed in the output. For transferring the bytes from the compressed file to the output file we are printing the message. Before closing the file and stream we are again printing the message.

|  |
| --- |
| **C:\java>java JavaUncompress C:\javaExample\unCompress\jsp.pdf.gz Opening the gzip file.......................... : opened Opening the output file............. : opened Transferring bytes from the compressed file to the output file........  : Transfer successful The file and stream is ......closing.......... : closed  C:\java>** |

# Example of a instance variable

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/JavaUncompress.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/staticvariable.shtml)**

When a number of objects are created from the same class,  the same copy of  instance variable is provided to all. Remember, each time you call the instance the same old value is provided to you, not the updated one . In this program we are showing that how a instance variable is called in each instance, but the copy remains same irrespective the counter we have used in the constructor of a class. We can't call the non-static variable in a main method. If we try to call the non-static method in the main method then the compiler will prompt you that **non-static variable cannot be referenced from a static context**. We can call the non-static variable by the instance of the class.

The example will show you how you can use a non-static variables. First create a class named NonStaticVariable. Declare one global variable and call it in the constructor overloaded method in which you will have to increment the value of the variable by the counter. To access a  non-static variable we will have to make a object of NonStaticVariable by using new operator. Now call the instance variable. Now again make a new object of the class and again call the instance variable. Now you can realize that the value of the instance variable in both the object is same.

**Code of this program is given below:**

|  |
| --- |
| **public class**NonStaticVariable{   **int**noOfInstances;   NonStaticVariable(){   noOfInstances++;   }   **public static void**main(String[] args){   NonStaticVariable st1 = **new**NonStaticVariable();   System.out.println("No. of instances  for st1  : "    + st1.noOfInstances);    NonStaticVariable st2 = **new**NonStaticVariable();   System.out.println("No. of instances  for st1  : "    + st1.noOfInstances);   System.out.println("No. of instances  for st2  : "    + st2.noOfInstances);    NonStaticVariable st3 = **new**NonStaticVariable();   System.out.println("No. of instances  for st1  : "    + st1.noOfInstances);   System.out.println("No. of instances  for st2  : "     + st2.noOfInstances);   System.out.println("No. of instances  for st3  : "    + st3.noOfInstances);    } } |

The output of this variable is given below :

As we can see in the output the same copy of the instance variable is provided to all the objects, no matter how many objects we create.

|  |
| --- |
| **C:\java>java NonStaticVariable No. of instances for st1 : 1 No. of instances for st1 : 1 No. of instances for st2 : 1 No. of instances for st1 : 1 No. of instances for st2 : 1 No. of instances for st3 : 1** |

# Example of a class variable (static variable)

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/nonstaticvariable.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/swapping.shtml)**

This Java programming example will teach you how you can define the static class variable in a class. When a number of objects are created from the same class, each instance has its own copy of class variables. But this is not the case when it is declared as static **static**.

**static**  method or a variable is not attached to a particular object, but rather to the class as a whole. They are allocated when theclass isloaded.Remember, each time you call the instance the new value of the variable is provided to you. For example in the class StaticVariable each instance has different copy of a class variable. It will be updated each time the instance has been called. We can call class variable directly inside the main method.

To see the use of a static variable first of all create a class **StaticVariable**. Define one static variable in the class. Now make a  constructor  in which you will increment the value of the static variable. Now make a object of **StaticVariable** class and call the static variable of the class. In the same way now make a second object of the class and again repeats the process. Each time you call the static variable you will get a new value.

**Code of this example is given below:**

|  |
| --- |
| **public class**StaticVariable{   **static int**noOfInstances;   StaticVariable(){   noOfInstances++;   } **public static void**main(String[] args){  StaticVariable sv1 = **new**StaticVariable();  System.out.println("No. of instances for sv1 : " + sv1.noOfInstances);   StaticVariable sv2 = **new**StaticVariable();  System.out.println("No. of instances for sv1 : "  + sv1.noOfInstances);  System.out.println("No. of instances for st2 : "  + sv2.noOfInstances);   StaticVariable sv3 = **new**StaticVariable();  System.out.println("No. of instances for sv1 : "  + sv1.noOfInstances);  System.out.println("No. of instances for sv2 : "  + sv2.noOfInstances);  System.out.println("No. of instances for sv3 : "  + sv3.noOfInstances);  } } |

Output of the program is given below:

As we can see in this example each object has its own copy of class variable.

|  |
| --- |
| **C:\java>java StaticVariable No. of instances for sv1 : 1 No. of instances for sv1 : 2 No. of instances for st2 :  2 No. of instances for sv1 : 3 No. of instances for sv2 : 3 No. of instances for sv3 : 3** |

# Swapping of two numbers

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/staticvariable.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/Swap.shtml)**

This Java programming tutorial will teach you the methods for writing program to calculate swap of two numbers. Swapping is used where  you want to interchange the values. This program will help you to increase your programming ability.

In this program we will see how we can swap two numbers. We can do this by using a temporary variable which is used to store variable so that we can swap the numbers. To swap two numbers first we have to declare a class Swapping. Inside a class declare one static method swap(int i, int j) having two arguments, the value of these arguments will be swapped. Now declare one local variable temp which will help us to swap the values. At last call the main method inside of which you will call the swap method and the result will be displayed to you.

**Code of the program is given below**

|  |
| --- |
| **public class**Swapping{   **static void**swap(**int**i,int j){   **int**temp=i;   i=j;   j=temp;   System.out.println("After swapping i = " + i + " j = " + j);   }   **public static void**main(String[] args){   **int**i=1;   **int**j=2;      System.out.prinln("Before swapping i="+i+" j="+j);   swap(i,j);      } } |

Output of this program is given below:

# Swap two any numbers (from Keyboard)

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/swapping.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/ChecksumByteArray.shtml)**

This is a simple Java Oriented  language program. If  you are newbie in Java programming then our tutorial and example are helpful for understanding Java programming in the simplest way.  Here we will learn how to swap or exchange the number to each other. First of all we have to create a class "Swap". Now, we use the **Integer.parseInt(args[o])** and**Integer.parseInt(args[1])**methods for getting the integer type values in command line. Use a temporary variable z of type integer that will help us to swap the numbers  Assign the value of x to z variable, now assign the value of y to the variable x. Lastly assign the value of z to variable y. Now we get the values has been interchanged. To display the values on the command prompt use **println()**method and the swapped values will be displayed.

**Here is the code of this program**

|  |
| --- |
| **public class**Swap {   **public static void**main(String[] args) {      **int**x= Integer.parseInt(args[0]);   **int**y= Integer.parseInt(args[1]);   System.out.println("Initial value: "+ x +" and " + y);   **int**z = x;   x = y;   y = z;   System.out.println("Swapped value: " + x +" and " + y);   } } |

[**Download this Example.**](http://www.roseindia.net/java/beginners/Swap.java)

# Comparing the File Dates

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ChecksumByteArrayCRC.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/GetAbsolutePath.shtml)**

This java example will help you to compare the dates of files. In java we have the option to compare two or more files. This is useful when we have to compare the files. Mostly it happens that after creating a file or a directory we forgets when it was made or last modified, and sometimes it becomes essential to know the exact date and time of creation. By this example you will be able to understand it.

To  make a program on this example firstly we have to make a class **ComparingFileDates**. Inside the class we have defined three static variables, which will be used for comparing the files. Declare a method comparing which will check when the  file is last modified  and which file is older or a new one. Now call the main method. Inside this method make a two objects of a **File** class which will take file name as its input in the constructor of **File** class. Now call the method lastModified by the instances of file classes which will return the time of the file creation. Now call the comparing method inside the main method and the result will be displayed to you.

In this program we have used the following classes and methods.

**File**: This class extends class **Object**and implements ***Comparable*** and ***Serializable*** interface. It takes the file as its input.

**lastModified():** It is a method of a **File** class. It returns the time when the file was last modified.

|  |
| --- |
| **import**java.io.\*;  **public class**ComparingFileDates {  **static**String compare;  **static long**modify1;  **static long**modify2;  **static void**comparing(){   //String compare;   **if**(modify1 == modify2)   compare = " are created at the same time along with";   **else if**(modify1 < modify2)   compare = " file is older than ";   **else**   compare = "file is newer than";    }   **public static void**main(String[] args) {   File file1 = **new**File(args[0]);   File file2 = **new**File(args[1]);      modify1 = file1.lastModified();   modify2 = file2.lastModified();   comparing();   System.out.println(file1 +  compare + ' ' + file2);   } } |

**Output of the program is given below.**

|  |
| --- |
| C:\java>java ComparingFileDates   file1.txt   file2.txt file1.txt   file is older than   file2.txt  C:\java> |

# Getting a absolute path

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ComparingFileDates.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/tokenizingjavasourcecode.shtml)**

If you are new in Java programming then our tutorials and examples will be helpful in understanding Java programming in the most simplest way. Here after reading this lesson, you will be able to find the absolute path of a file or directory.

To find a file or directory it is very necessary to know the path of the file or directory so that you can access it. If you know the path then it is very easy to work on it. Suppose a situation where a problem comes in front you where you don't know the path of the file, then what will you do. Then this problem can be solved by using a method *getAbsolutePath()*. The method *getAbsolutePath()* should be used where we don't know the exact path of the file.

To find an absolute path of a file, Firstly we have to make a class **GetAbsolutePath**. Inside this class define the main method. Inside this method  define a File class of java.io package. Inside the constructor of a File class pass the name of the file whose absolute path you want to know. Now call the method *getAbsolutePath()* of the **File**class by the reference of  **File** class and store it in a String variable. Now print the string, you will get a absolute path of the file.  
In this class we have make use of the following things by which this problem can be solved.

**File**: It is class in java.io package. It implements ***Comparable*** and ***Serializable*** interface.

*getAbsolutePath()*:  It returns the absolute path name in  the form of string.

**Code of the program is given below:**

|  |
| --- |
| **import**java.io.\*;  **public class**GetAbsolutePath{   **public static void**main(String[] args){   String str = args[0];   File file = **new**File(str);   String absolutePathOfFirstFile = file.getAbsolutePath();   System.out.println(" The absolute path in first form is "      + absolutePathOfFirstFile);   file = **new**File( "java" + File.separatorChar+ str);   String absolutePathOfSecondFile = file.getAbsolutePath();   System.out.println(" The absolute path is " + absolutePathOfSecondFile);   file = **new**File("java" + File.separator + ".." + File.separator + str);   String absolutePathOfThirdFile = file.getAbsolutePath();   System.out.println(" The absolute path is " + absolutePathOfThirdFile);    } } |

Output of the program is given below.

|  |
| --- |
| **C:\java>java GetAbsolutePath java The absolute path in first form is C:\java\java The absolute path is C:\java\java\java The absolute path is C:\java\java\..\java** |

# Making Tokens of a Java Source Code

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/GetAbsolutePath.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/AddTwoBigNumbers.shtml)**

If you are new in Java programming then our tutorials and examples will be helpful in understanding Java programming in the most simplest way. Here after reading this lesson, you will be able to break java program into tokens.

The use of  token is just like as we take tokens for getting a milk or get into a metro, so that the one token can be read at a time.   
Suppose if we have to make a token of a java file then we will do. Firstly we have to make a class**TokenizingJavaSourceCode**. Declare a main method in it . Now make a object of **BufferedReader**class and pass the **InputStream**class object into it which will read the name of the java file at run time. Now we need to read the file so we use **FileReader** class ,after reading a file we need to generate a tokens out of that file so, we use class **StreamTokenizer** and pass reference of a **Reader** class in its constructor. In this program we are using one method of **StreamTokenizer**class.

To solve this problem we have make use of the following classes and methods.

**InputStream:**It is a abstract class and the parent class of all the **Stream** classes.

**BufferedReader**: This class reads the text from the input stream. This class extends **Reade**r class.

**File:**It is a class implementing ***Serializable*** and ***Comparable*** interface.

**FileReader**:  This class  reads the file in the character format. It extends **InputStreamReader**.

**StreamTokenizer**: This class takes a input stream and break it into tokens so that one token can be read at a time. It takes **FileReader** instance as a input in its constructor.

**readLine():** It is a method of **BufferedReader**class which reads a line.

**exists():** It will check whether the file exists or not.

**nextToken():** It gives the next token.

We have also used one **final static** variable **TT\_EOF** which tells the reader that the end of the file has been read.

**The code of this program is given below:**

|  |
| --- |
| **import**java.io.\*;  **import**java.util.\*;  **public class**TokenizingJavaSourceCode{   //public static final int TT\_EOL;    **public static void**main(String[] args) **throws**IOException{   BufferedReader in = **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.print("Please enter a java file name: ");   String filename = in.readLine();   **if**(!filename.endsWith(".java")){   System.out.println("This is not a java file.");   System.exit(0);   }   File javaFile = **new**File(filename);   **if**(javaFile.exists()){   FileReader file = **new**FileReader(filename);   StreamTokenizer streamTokenizer = **new**StreamTokenizer(file);   // It will go through the file and    //gives the number of tokens in the file   **int**i=0;   **int**numberOfTokensGenerated = 0;   **while**(i != StreamTokenizer.TT\_EOF){   i = streamTokenizer.nextToken();   numberOfTokensGenerated++;   }   System.out.println("Number of tokens = " + numberOfTokensGenerated);   }   **else**{   System.out.println("File does not exist!");   System.exit(0);   }   } } |

**Output of this program is given below:**

|  |
| --- |
| **C:\Java Tutorial>java TokenizingJavaSourceCode Please enter a java file name: TokenizingJavaSourceCode.java Number of tokens = 158  C:\Java Tutorial>** |

# Add two big numbers

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/tokenizingjavasourcecode.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/AppendInsert.shtml)**

In this section, you will learn how to add two big numbers. For adding two numbers implement two big decimal numbers then apply the Sum() method that takes both numbers. Now, you will have to define it and use the **add()**method to add both numbers. Finally, the result will be displayed on the command prompt (DOS prompt).

**BigDecimal():**This is the constructor of **BigDecimal** class. It translates the data into BigDecimal.

**BigDecimal.add(BigDecimal big):**This method is used to add the two BigDecimal numbers.

**Here is the code of program:**

|  |
| --- |
| **import**java.math.BigDecimal;  **public class**AddTwoBigNumbers{   **public static void**main(String[] args) {   BigDecimal num1, num2;   num1 = **new**BigDecimal(50.00035);   num2 = **new**BigDecimal(100.0025);   Sum(num1, num2);   }    **public static void**Sum(BigDecimal val1, BigDecimal val2){   BigDecimal sum = val1.add(val2);   System.out.println("Sum of two BigDecimal numbers: "+ sum);   } } |

[**Download this example.**](http://www.roseindia.net/java/beginners/AddTwoBigNumbers.java)

**Output of program:**

|  |
| --- |
| C:\vinod\Math\_package>javac AddTwoBigNumbers.java  C:\vinod\Math\_package>java AddTwoBigNumbers Sum of two BigDecimal numbers: 150.002849999999995134203345514833927154 |

# String Buffer insert and append example

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/AddTwoBigNumbers.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/CharComparation.shtml)**

In this section, you will learn how to work with **StringBuffer** and append data to it. The StringBuffer is a class that implements multiple sequence of characters. The following program construct a string buffer that has '**0**' capacity. This program inserts the data in the string buffer using the**insert()** method that provides the facility for adding the data in any positions. But the **append()**method also add data in string buffer in ending position. This method inserts always last or end position.

**Description of code:**

**StringBuffer(int len):**This is the constructor of **StringBuffer** class that implements multiple sequence of characters. This constructor constructs a string buffer that have no characters. It takes initial capacity that have to initialized.

**insert(int pos, String str):**This method inserts the string into the object of **StringBuffer**. It takes following parameters:

**pos:**This is the position that have to be used for inserting the data.  
  **str:**This is the string that have to inserted.

**append(String str):**This is the method that inserts the string into string buffer. It always inserts a string at last position.

**Here is the code of program**

|  |
| --- |
| **import**java.lang.\*;  **public class**AppendInsert{   **public static void**main(String[] args) {   System.out.println("StringBuffer insert and append example!");   StringBuffer sb = **new**StringBuffer(0);   //First position   System.out.println(sb.insert(0, "vinod"));   **int**len = sb.length();   //last position   System.out.println(sb.insert(len, "Deepak"));   //Six position   System.out.println(sb.insert(6, "Raj"));   //Always last   System.out.println(sb.append("Mohit"));   } } |

[**Download this example.**](http://www.roseindia.net/java/beginners/AppendInsert.java)

**Output of program:**

|  |
| --- |
| C:\vinod\Math\_package>javac AppendInsert.java  C:\vinod\Math\_package>java AppendInsert StringBuffer insert and append example! vinod vinodDeepak vinodDRajeepak vinodDRajeepakMohit |

# Character Comparison Example

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/AppendInsert.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/CharCompIgnoreCase.shtml)**

In this section, you will learn how to compare characters in Java. The ***java.lang***package provides a method for comparing two case sensitive  strings. The **compareTo()** method compares two strings on the basis of Unicode of each character of the given strings. This method returns integer type value. If it will return '**0**' it means the given string is equal otherwise not equal.

**Description of program:**

This program helps you to compare characters with cases and sequence. For comparing the string you will need those string that have to be compared. Here the **CharComp()** method is applied for comparing the string. At the time of defining, you will use the **compareTo()** method that compares both strings and shows an appropriate given message.

**Description of code:**

**compareTo(String str):**Above method is used to compare two strings with its cases. If it returns '0', an argument of the given string is equal otherwise not. It takes string type parameter as following:

**str:** This is the string that have to be compared.

**Here is the code of program:**

|  |
| --- |
| **import**java.lang.\*;  **public class**CharComparation{   **public static void**main(String[] args) {   System.out.println("Character comparation example!");   String str1 = "Vinod";   String str2 = "Vinod";   String str3 = "vinod";   CharComp(str1, str2, str3);   }   **public static void**CharComp(String str1, String str2, String str3){   System.out.println("String1 = " + str1);   System.out.println("String2 = " + str2);   System.out.println("String2 = " + str3);   **if**(str1.compareTo(str2) == 0){   System.out.println("String1 and String2 are equal!");   }   **else**{   System.out.println("String1 and String2 are not equal!");   }   **if**(str1.compareTo(str3) == 0){   System.out.println("String1 and String3 are equal!");   }   **else**{   System.out.println("String1 and String3 are not equal!");   }   **if**(str2.compareTo(str3) == 0){   System.out.println("String2 and String3 are equal!");   }   **else**{   System.out.println("String2 and String3 are not equal!");   }   } } |

[**Download this example.**](http://www.roseindia.net/java/beginners/CharComparation.java)

**Output of program:**

|  |
| --- |
| C:\vinod\Math\_package>javac CharComparation.java  C:\vinod\Math\_package>java CharComparation Character comparation example! String1 = Vinod String2 = Vinod String2 = vinod String1 and String2 are equal! String1 and String3 are not equal! String2 and String3 are not equal! |

# Combine String example

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CharCompIgnoreCase.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/CompString.shtml)**

In this section, you will learn how to combine or merge two strings in java. The***java.lang***package provides the method that helps you to combine two strings and making  into single string. The following program is used for concatenating two string through using the **concat()**method that concatenates the specified string to the end of string and finally, you will get the combined string.

**Description of code:**

**concat(String str):**This is the method that concatenates the two specified strings.

**Here is the code of program:**

|  |
| --- |
| **import**java.io.\*;  **public class**CombinString{   **public static void**main(String[] args) **throws**IOException{   BufferedReader bf =  **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.println("Enter First String:");   String str1 = bf.readLine();   System.out.println("Enter Second String:");   String str2 = bf.readLine();   System.out.println("Combin string example!");   String com = str1.concat(str2);   System.out.println("Combined string: " + com);   } } |

[**Download this example.**](http://www.roseindia.net/java/beginners/CombinString.java)

**Output of program:**

|  |
| --- |
| C:\vinod\Math\_package>javac CombinString.java  C:\vinod\Math\_package>java CombinString Enter First String: RoseIndia Enter Second String: NewstrackIndia Combin string example! Combined string: RoseIndiaNewstrackIndia |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Compare strings example **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CombinString.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/replace.shtml)**  In this section, you will learn how to**compare two strings** in java. The**java lang** package provides a method to compare two with their case either upper and lower. The **equals()**method provides the facility of comparing the two strings. The following program uses the equals() method and helps you to compare the two strings. If both strings are equal, it will display a message "**The given strings are equal**" otherwise it will show "**The given string are not equal**".  **Description of code:**  **equals():** This is the method that compares an object values and returns Boolean type value either '**true**' or '**false**'. If it returns 'true' for the both objects, it will be equal otherwise not.  **Here is the code of program:**   |  | | --- | | **import**java.lang.\*; **import**java.io.\*;  **public class**CompString{   **public static void**main(String[] args) **throws**IOException{   System.out.println("String equals or not example!");   BufferedReader bf =  **new**BufferedReader(**new**InputStreamReader(System.in));   System.out.println("Please enter first string:");   String str1 = bf.readLine();   System.out.println("Please enter second string:");   String str2 = bf.readLine();   **if**(str1.equals(str2)){   System.out.println("The given string is equals");   }   **else**{   System.out.println("The given string is not equals");   }   } } |   [**Download this example.**](http://www.roseindia.net/java/beginners/CompString.java)  **Output of program:**  **Both are equals:**   |  | | --- | | C:\vinod\Math\_package>javac CompString.java  C:\vinod\Math\_package>java CompString String equals or not example! Please enter first string: Rose Please enter second string: Rose The given string is equals |   **Both are not equals:**   |  | | --- | | C:\vinod\Math\_package>java CompString String equals or not example! Please enter first string: Rose Please enter second string: rose The given string is not equals |  Replace Character in String **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CompString.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/ConvertStringToFloat.shtml)**  This example replaces a character with a specified character in a given string.  To replace a character with the given character in sting first convert the string into char array. Use **getChars(int scrStart, int scrEnd, char[] destChar, int destStart)** method for converting a string into an array of characters. Then use while loop for comparing the "oldChar" ( character name for replace) to be change with the new character of the array. If any match find then replace the "oldChar" with newChar (character name to replace) and set flag =1. To convert charArray into string, pass "charArray" into **String** class.  **getChars(int scrStart, int scrEnd, char[] destChar, int destStart):**This method returns an array of characters from a string. We are passing four parameter into this method. First parameter **scrStart**is the starting point while second parameter **scrEnd**is the end point of the source string to convert the string into a char array**.**The**destChar**is the destined array which stores all the characters**.**The**destStart**is starting index to store the characters.     **The code of the program is given below:**   |  | | --- | | **public class**Replace{   **public static void**main(String s[]){   String string="rajesh raju raja rahul ray rani ram";   **char**oldChar='r';   **char**newChar='g';   **int**numChar=string.length();   **char**[] charArray=**new char**[numChar];   string.getChars(0, numChar, charArray,0);   **int**i=0,flag=0;     System.out.println("Your String before  repalce\n"+string);   **while**(i<charArray.length)   {   **if**(charArray[i]==oldChar)   {   charArray[i]=newChar;   flag=1;   }   i++;   }   **if**(flag==1)   {   System.out.println("\nYour String after  repalceing 'h' with 'j'");   String newString=**new**String(charArray);   System.out.println(newString+"\n\nYour  char has been replaced");   }   **if**(flag==0)   {   System.out.println("\nThe char not found");   }     } } |   **The output of the program is given below:**   |  | | --- | | C:\replace>javac Replace.java  C:\replace>java Replace  Your String before repalce  rajesh raju raja rahul ray rani ram  Your String after repalceing 'h' with 'j'  gajesh gaju gaja gahul gay gani gam  Your char has been replaced |  Trim String Example **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/StringLength1.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/StrStartWith.shtml)**  In this section, you will learn how to remove the blank spaces. For removing  the blank spaces use **trim()** method that removes the blank spaces and shows only string.  **Description of code:**  **trim():** This method removes the blank spaces from both ends of the given string (Front and End).  **Here is the code of program:**         |  | | --- | | **import**java.lang.\*;  **public class**StringTrim{   **public static void**main(String[] args) {   System.out.println("String trim example!");   String str = " RoseIndia";   System.out.println("Given String :" + str);   System.out.println("After trim :" +str.trim());   } } |   [**Download this example.**](http://www.roseindia.net/java/beginners/StringTrim.java)  **Output of program:**   |  | | --- | | C:\vinod\Math\_package>javac StringTrim.java  C:\vinod\Math\_package>java StringTrim String trim example! Given String :  RoseIndia After trim :RoseIndia |  String Start with Example **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/StringTrim.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/StrEndWith.shtml)**  In this section, you will learn how to check the given string that start from the specified character in java. The following program checks the started string with "**Wel**". If  you  use **startsWith()**function it will return 'true' and display a message "***The given string is start with Wel***" otherwise it will show "***The given string is not start with Wel***".  **Description of code:**  **startsWith(String start):** This is a Boolean type method that returns either 'true' or 'false'. It checks the given string that begins with specified string in the beginning. It takes a string type parameter such as:  **start**: This is the string that starts.  **Here is the code of program:**   |  | | --- | | **import**java.lang.\*;  **public class**StrStartWith{   **public static void**main(String[] args) {   System.out.println("String start with example!");   String str = "Welcome to RoseIndia";   String start = "Wel";   System.out.println("Given String : " + str);   System.out.println("Start with : " + start);   **if**(str.startsWith(start)){   System.out.println("The given string is start with Wel");   }   **else**{   System.out.println("The given string is not start with Wel");   }   } } |   [**Download this example.**](http://www.roseindia.net/java/beginners/StrStartWith.java)  **Output of program:**   |  | | --- | | C:\vinod\Math\_package>javac StrStartWith.java  C:\vinod\Math\_package>java StrStartWith String start with example! Given String : Welcome to RoseIndia Start with : Wel The given string is start with Wel |  Passing Command Line Arguments **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/StrEndWith.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DecimalFormatExample.shtml)**  In this example we are going to learn how we can pass values on command line argument .  We are taking a sequence of character from command line at run time. Store these strings into an array string. Show these strings on console as output. If user forget enter string at run time on command line argument then the message  "No values has been entered at the command line" will displayed on screen. To check the user enter string or not at run time we just compare length of argument values .If value is greater than one then display the values on screen else so message  "No values has been entered at the command line".  **The code of the program is given below:**   |  | | --- | | **public class**ArgumentPassingExample{   **public static void**main(String[] args){   **int**num=args.length;   String s[]=**new**String[num];    **if**(num>0){   System.out.println("The values enter at  argument command line are:");    **for**(**int**i = 0; i <num ; i++)   {   System.out.println("Argument " + (i + 1) +  " = " + args[i]);   }   }   **else**{   System.out.println("No values has been  entered at the command line.");   }     } } |   **The output of the program is given below:**   |  | | --- | | C:\convert\rajesh\completed>javac ArgumentPassingExample.java  C:\convert\rajesh\completed>java ArgumentPassingExample rajesh kumar rahul  The values enter at argument command line are:  Argument 1 = rajesh  Argument 2 = kumar  Argument 3 = rahul |  Decimal Format Example **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ArgumentPassingExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DelayExample.shtml)**  In this example we are going to format a decimal value.  In this example we are talking a double value. We are creating an object of**DecimalFormat(String format\_type)**class .In **DecimalFormat** class we are passing **"0.000"** .This is used to create an decimal format which display three digits after decimal point. We are using **NumberFormat** to store the reference of  object of **DecimalFormat**with name of formatter. To apply the format on double value we are using **format()**method**.**  **The code of the program is given below:**   |  | | --- | | **import**java.text.DecimalFormat; **import**java.text.NumberFormat;   **public class**DecimalFormatExample {   **public static void**main(String args[])   {   **double**amount = 2192.015;   NumberFormat formatter = **new**DecimalFormat("#0.000");   System.out.println("The Decimal Value is:"+formatter.format(amount));   } } |   **The output of the program is given below:**   |  | | --- | | C:\convert\rajesh\completed>javac DecimalFormatExample.java  C:\convert\rajesh\completed>java DecimalFormatExample  The Decimal Value is:2192.015 |  Line Number Reader Example **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/DelayExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/OSInformation.shtml)**  In this example we are reading  number of lines.  In this example we are reading a file from system .Then we are reading number of lines in this file. **FileReader**class provides**readLine()** method.The  **readLine()** method is used to read the data line by line. **LineNumberReader**class is provide **getLineNumber()** method to get the number of line. We are using **File(String**file\_name**)**constructor to make an object of file. We are reading file by use of **FileReader(File file)**constructor. We are using **LineNumberReader(FileReader** filereader**)**constructor to create an object of**LineNumberReader** class.  **The methods used in this example:**  **getLineNumber():** This method is used to get the number of line.  **readLine():** This method is used to read the data line by line. Then returns the data of a line as a single string.  **The code of the program is given below:**   |  | | --- | | **import**java.io.\*; **public class**LineNumberReaderExample{   **public static void**main(String[] args) **throws**Exception{   File file = **null**;   FileReader freader = **null**;   LineNumberReader lnreader = **null**;   **try**{   file = **new**File("LineNumberReaderExample.java");   freader = **new**FileReader(file);   lnreader = **new**LineNumberReader(freader);   String line = "";   **while**((line = lnreader.readLine()) != **null**){   System.out.println("Line:  " + lnreader. getLineNumber() + ": " + line);   }   }   **finally**{   freader.close();   lnreader.close();   }   } } |   **The output of the program is given below:**   |  | | --- | | C:\convert\rajesh\completed>javac LineNumberReaderExample.java  C:\convert\rajesh\completed>java LineNumberReaderExample  Line: 1: import java.io.\*;  Line: 2: public class LineNumberReaderExample{  Line: 3: public static void main(String[] args)  throws Exception{  Line: 4: File file = null;  Line: 5: FileReader freader = null;  Line: 6: LineNumberReader lnreader = null;  Line: 7: try{  Line: 8: file = new File("  LineNumberReaderExample.java");  Line: 9: freader = new FileReader(file);  Line: 10: lnreader = new  LineNumberReader(freader);  Line: 11: String line = "";  Line: 12: while ((line =  lnreader.readLine()) != null){  Line: 13: System.out.  println("Line: " + lnreader.  getLineNumber() + ": " + line);  Line: 14: }  Line: 15: }  Line: 16: finally{  Line: 17: freader.close();  Line: 18: lnreader.close();  Line: 19: }  Line: 20: }  Line: 21: } |  Operating System Information **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/LineNumberReaderExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/SubstringExample.shtml)**  In this example we are learn how we can get information about our operation system. In this example we are getting the OS name, its version and architecture of OS. We are using **getProperty(String key)** to get the property of the OS. To get OS name we have key value *os.name*,to get version we have*os.version* and to get architecture we have *os.orch*.  The **System** class contains several useful class fields and methods. It cannot be instantiated.  **The method and its keys:**  **String getProperty(String key):** This method is used to get the system property.   |  |  |  | | --- | --- | --- | | **S/No.** | **Key** | **Description** | | 1 | java.version | The version of Java Runtime Environment. | | 2 | java.vendor | The name of Java Runtime Environment vendor | | 3 | java.vendor.url | The URL of Java vendor | | 4 | java.home | The directory of Java installation | | 5 | java.vm.specification.version | The specificationversion of Java Virtual Machine | | 6 | java.vm.specification.vendor | The name of specification vendor of Java Virtual Machine | | 7 | java.vm.specification.name | Java Virtual Machine specification name | | 8 | java.vm.version | JVM implementation version | | 9 | java.vm.vendor | JVM implementation vendor | | 10 | java.vm.name | JVM  implementation name | | 11 | java.specification.version | The name of specification version Java Runtime Environment | | 12 | java.specification.vendor | JRE specification vendor | | 13 | java.specification.name | JREspecification name | | 14 | java.class.version | Java class format version number | | 15 | ava.class.path | Path of java class | | 16 | java.library.path | List of paths to search when loading libraries | | 17 | java.io.tmpdir | The path of temp file | | 18 | java.compiler | The Name of JIT compiler to use | | 19 | java.ext.dirs | The path of extension directory or directories | | 20 | os.name | The name of OS name | | 21 | os.arch | The OS architecture | | 22 | os.version | The version of OS | | 23 | file.separator | The File separator | | 24 | path.separator | The path separator | | 25 | line.separator | The line separator | | 26 | user.name | The name of account name user | | 27 | user.home | The home directory of user | | 28 | user.dir | The current working directory of the user |   **The code of the program is given below:**   |  | | --- | | **public class**OpertingSystemInfo  {   **public static void**main(String[] args)   {   String nameOS = "os.name";     String versionOS = "os.version";     String architectureOS = "os.arch";   System.out.println("\n  The information about OS");   System.out.println("\nName of the OS: " +  System.getProperty(nameOS));   System.out.println("Version of the OS: " +  System.getProperty(versionOS));   System.out.println("Architecture of THe OS: " +  System.getProperty(architectureOS));   } } |   **The output of the program is given below:**   |  | | --- | | C:\convert\rajesh\completed>javac OpertingSystemInfo.java  C:\convert\rajesh\completed>java OpertingSystemInfo  The information about OS  Name of the OS: Windows 2000  Version of the OS: 5.0  Architecture of The OS: x86 |  Taking  Substring **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/OSInformation.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/TempDirExample.shtml)**  In this example we are taking a sub string from a given string.  In this example we are creating an string object .We initialize this string object as "Rajesh Kumar". We are taking sub string by use of **substring()**  method.  **The methods used: substring(int i):** This method is used to find all sub string after index i.  **substring(int start,int end):** This is used to find the substring between start and end point.  **The code of the program is given below:**   |  | | --- | | **public class**SubstringExample1{   **public static void**main(String[] args){   String string = "Rajesh kumar";    System.out.println("String : " + string);   String substring = string.substring(3);   System.out.println("String after 3rd index:  " + substring);    substring = string.substring(1, 2);   System.out.println("Substring (1,2): " +  substring);   } } |   **The output of the program is given below:**   |  | | --- | | C:\convert\rajesh\completed>javac SubstringExample1.java  C:\convert\rajesh\completed>java SubstringExample1  String : Rajesh kumar  String after 3rd index: esh kumar  Substring (1,2): a |  Find Current Temp Directory **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/SubstringExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/UserHomeExample.shtml)**  In this example we are find the current Temp directory.  We are using **getProperty(String** key**)**method to find the current temp directory. The **getProperty(String** key**)**is defined into **System**class. **System**class extends directly **Object**class. **System**class is defined final so any class never extends **System** class. **System** class allow us to get or set system information. In this example we are pasing "*java.io.tmpdir*" as key to get the current temp directory into **getProperty()** method.  **The code of the program is given below:**   |  | | --- | | **public class**TempDirExample  {   **public static void**main(String[] args)   {      System.out.println("OS current temporary directory is "  + System.getProperty("java.io.tmpdir"));   } } |   **The output of the program is given below:**   |  | | --- | | C:\convert\rajesh\completed>javac TempDirExample.java  C:\convert\rajesh\completed>java TempDirExample  OS current temporary directory is C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\ |  Find User Home Directory **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/TempDirExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/WriteTextFileExample.shtml)**  In this example we are find user home directory.  We are using **System**class  to get the information about system. **System** class extends **Object** class. **System** class allow us to get or set system information.  **The method used into this example:**  **getProperty(String**Key**):** This method is used to get the property of system for passing Key values.  In this example we are passing "*user.home*" as key to get the user define home directory.  **The code of the program is given below:**   |  | | --- | | **public class**UserHomeExample  {   **public static void**main(String[] args)   {     System.out.println("User Home Path: "+ System.getProperty("user.home"));   } } |   **The output of the program is given below:**   |  | | --- | | C:\convert\rajesh\completed>javac UserHomeExample.java  C:\convert\rajesh\completed>java UserHomeExample  User Home Path: C:\Documents and Settings\Administrator | |

# Write Text into File

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/UserHomeExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/WordCountExample.shtml)**

In this example we are writing text into file.In this example we are initialize string to write into file. We are creating file in which we are writing string by use of **write()**method.

We need  **java.io.\***  package import first. The create   a .txt file with name "write.txt". We are using **FileWriter** class to read file for modification.**BufferedWriter**class is used for buffering the file which will store into an object of  **Writer**class.Then we are using **write()** method to write text into file. At last close output file using **close()** method.

**The code of the program is given below:**

|  |
| --- |
| **import**java.io.\*;  **public class**WriteTextFileExample{   **public static void**main(String[] args)**throws**IOException{   Writer output = **null**;   String text = "Rajesh Kumar";   File file = **new**File("write.txt");   output = **new**BufferedWriter(**new**FileWriter(file));   output.write(text);   output.close();   System.out.println("Your file has been written");     } } |

**The output of the program is given below:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Word Count **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/WriteTextFileExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/StringReverseUsingStringUtils.shtml)**  This example counts the number of occurrences of  a specific word in a string. Here we are counting the occurrences of word "you" in a string. To count it we are using **countMatches()**method.  The **org.apache.commons.lang.StringUtils** class extends the **Object** class and defines certain words related to **String**handling such as null for null,**""**for a **zero-length**string,**' '** for space characters, **Charecter.isWhitespace(char)** for **whitespace** and **String.trim()**for **trim**. The **StringUtils**class handles null input strings.  The method used:  **countMatches(String str,String sub):**This method counts how many times the string **sub** appears in the String **str**. This method returns zero if  **StringUtils.countMatches(null, \*)**,**StringUtils.countMatches("", \*)**,**StringUtils.countMatches("abba", null)**,**StringUtils.countMatches("abba", "")** , and **StringUtils.countMatches("abba", "x")**. The parameters used  as "str" is String to be checked and "sub" is substring to be count.   **The code of the program is given below:**   |  | | --- | | **import**org.apache.commons.lang.StringUtils;   **public class**WordCountExample  {     **public static void**main(String[] args)   {   String string = "How r you?R you fine?Where are you going?";     System.out.println(StringUtils.countMatches(string,"you") + " occurrences of the word 'you' in '" + string +   "' is found.");   } } |   **The output of the program is given below:**   |  | | --- | | C:\rajesh\kodejava>javac WordCountExample.java  C:\rajesh\kodejava>java WordCountExample  3 occurrences of the word 'you' in 'How r you?R you fine?  Where are you going?' is found. |  String Reverse Using StringUtils **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/WordCountExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/StringReverseExample.shtml)**  In this example we are going to reverse a given string using **StringUtils** api. In this example we are reversing a string and the reversed string  is delimited by a specific character.  The **org.apache.commons.lang.StringUtils** class extends **Object** class and defines certain words related to String handling such as null for null,**""**for a **zero-length**string,**' '** for space characters,**Charecter.isWhitespace(char)** for **whitespace** and **String.trim()**for **trim**.  The methods used:  **reverse(String str):**This method is used to reverse the order of a buffered string. It returns a null value if no string is passed i.e **StringUtils.reverse(null)** = null or **StringUtils.reverse("")** = "" otherwise reverse order will returns e.g. **StringUtils.reverse("bat")** = "tab".  **reverseDelimited(String str,char separatorChar):**This method is used to reverse a String that is delimited by a specific character like "". The strings between the delimiters will not reverse. In this method "str" is string to reverse and "separatorChar" is the separator character.   **The code of the program is given below:**   |  | | --- | | **import**org.apache.commons.lang.StringUtils;   **public class**StringReverseUsingStringUtils  {     **public static void**main(String[] args)   {   String string = "Hi, How R YOU?";    String reverse = StringUtils.reverse(string);   String delimitedReverse = StringUtils. reverseDelimited(string, ' ');    System.out.println("\nThe original String: " + string);   System.out.println("The reversed string: " + reverse);   System.out.println("The delimited Reverse string: "  + delimitedReverse);   } } |   **The output of the program is given below:**   |  | | --- | | C:\rajesh\kodejava>javac StringReverseUsingStringUtils.java  C:\rajesh\kodejava>java StringReverseUsingStringUtils  The original String: Hi, How R YOU?  The reversed string: ?UOY R woH ,iH  The delimited Reverse string: YOU? R How Hi, |  Singleton Design Pattern **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/StringReverseExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/RuntimeExec.shtml)**  This example explores how to implement a**SingletonPattern** on a class in java.  The Singleton design pattern ensures that only one instance of a class is created, it provides a global point of access to the object and allow multiple instances in the future without affecting a singleton class's clients. To ensure that only one instance of a class is created we make SingletonPattern as static. **getInstance()** method returns a single instance of the class. We create a private default constructor to avoid outside modification.  This example gets instances of SingletonPattern two times but the method **getInstance()** will return the same object without creating a new one.   **The code of the program is given below:**   |  | | --- | | **public class**SingletonPattern{   **private static**SingletonPattern instance;   **private**SingletonPattern(){}  **public static synchronized**SingletonPattern getInstance(){   **if**(instance == **null**)   {   instance = **new**SingletonPattern();   }   **return**instance;   } **public static void**main(String arg[]){   System.out.println("The output of two instance:");   SingletonPattern sp=**new**SingletonPattern();   System.out.println("First Instance: "+sp.getInstance());   sp=**new**SingletonPattern();   System.out.println("Second Instance:"+sp.getInstance());   } } |   **The output of the program is given below:**   |  | | --- | | C:\rajesh\kodejava>javac SingletonPattern.java  C:\rajesh\kodejava>java SingletonPattern  The output of two instance:  First Instance: SingletonPattern@3e25a5  Second Instance:SingletonPattern@3e25a5 |  Runtime Exec Example **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/SingletonPattern.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/EnvironmentInformation.shtml)**  In this example we are going to execute an application using java program.   This example uses the **Process** class that is contained in the lang package and extends **Object** class. The method **Runtime.exec()** creates a native process and returns an instance of a subclass of the Process class. This instance is then used to control the process and to obtain the information about the process. The Process class provides methods for performing input ,output , wait, checking the exit status , and destroying (killing) the process.  The **Runtime.exec()**method may not work well for windowing processes, daemon processes, Win16/DOS processes on Microsoft Windows, or shell scripts.  The class **Runtime** contained in java.lang package and extends the **Object**class. Every Java application has a single instance of class Runtime. The current runtime can be obtained by using the**getRuntime()** method. Any application cannot create its own instance of runtime class.  The method used: **exec(String command):**This method is used to execute the command. Here command is a specified system command. It can also be used to execute any .exe file.     **The code of the program is given below:**   |  | | --- | | **import**java.io.IOException;  **public class**RuntimeExec{   **public static void**main(String[] args){   **try**   {   Process process = Runtime.getRuntime() .exec("notepad.exe");   }**catch**(IOException e)   {  e.printStackTrace();   }   } } |   **The output of the program is given below:**  http://www.roseindia.net/java/beginners/exeimage.bmp Find Day of Month **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/EnvironmentInformation.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DayOfYearToDayOfWeek.shtml)**  This example explores how to find the day of a month and the day of a week  This example sets the year as 2007 and the day as 181. The example  finds the day of a month and the day of a week by using**get(field\_name)** method.  The **Calendar** class extends **Object** class. It is an abstract base class and converts a **Date** object into a set of integer fields. **Calendar** class provides a **getInstance()**  method that returns a **Calendar**object whose time fields are initialized with the current date and time.  The methods used: **getTime():**This method is used to get current time from calendar. **getInstance():**This method gets a calendar using the default time zone , locale and current time.  The fields used: **WEEK\_OF\_MONTH:** This field is used to get and set the week indicating the week number within the current month. **DAY\_OF\_WEEK:**This field gets and set the day indicating the day of the week. This field takes values SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, and SATURDAY.  **The code of the program is given below:**   |  | | --- | | **import**java.util.Calendar;   **public class**DayYearToDayMonth  {   **public static void**main(String[] args)   {    Calendar calendar = Calendar.getInstance();   calendar.set(Calendar.YEAR, 2007);   calendar.set(Calendar.DAY\_OF\_YEAR, 181);   System.out.println("\nThe date of Calendar is: " +  calendar.getTime());   **int**dayOfMonth = calendar.get(Calendar.DAY\_OF\_MONTH);     System.out.println("The day of month: " +  dayOfMonth);     **int**dayOfWeek = calendar.get(Calendar.DAY\_OF\_WEEK);   System.out.println("The day of week: " +  dayOfWeek);   } } |   **The output of the program is given below:**   |  | | --- | | C:\rajesh\kodejava>javac DayYearToDayMonth.java  C:\rajesh\kodejava>java DayYearToDayMonth  The date of Calendar is: Sat Jun 30 17:03:01 GMT+05:30 2007  The day of month: 30  The day of week: 7 |  Find the Day of the Week **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/DayYearToDayMonth.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DateDifferent.shtml)**  This example finds the specified date of  an year and a day of a week from calendar.  The **Calendar** class is an abstract base class that extends **Object** class and converts a **Date** object into a set of integer fields. **Calendar** class provides a **getInstance()**  method  for returning a **Calendar** object whose time fields have been initialized with the current date and time.  **The methods used:** **getTime():**This method gets the current time from the calendar.  **setTimeInMillis(long millis):**This method sets the current time in a calendar object.  **getInstance():**This method gets a calendar object containing the default time zone, locale and current time.  The fields used:  **DAY\_OF\_WEEK:**This field is used to  get and set a day indicating the day of the week. This field takes values SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, and SATURDAY.  **DAY\_OF\_YEAR:**This field gets and sets the day to indicate the day number within a current year. e.g. The first day of the year has value 1.  **YEAR:**This field is used to get and set the year for indicating the year.  **The code of the program is given below:**   |  | | --- | | **import**java.util.Calendar;  **public class**DayOfYearToDayOfWeek{   **public static void**main(String[] args){   Calendar calendar = Calendar.getInstance();   **for**(**int**i=2000;i<2005;i++){   calendar.set(Calendar.YEAR, i);   calendar.set(Calendar.DAY\_OF\_YEAR,0);   System.out.println(calendar.getTime().toString());   **int**weekday = calendar.get(Calendar.DAY\_OF\_WEEK);   System.out.println("Weekday: " + weekday);   System.out.println();   }   } } |   **The output of the program is given below:**   |  | | --- | | C:\rajesh\kodejava>javac DayOfYearToDayOfWeek  C:\rajesh\kodejava>java DayOfYearToDayOfWeek  Fri Dec 31 16:21:22 GMT+05:30 1999  Weakday: 6  Sun Dec 31 16:21:22 GMT+05:30 2000  Weakday: 1  Mon Dec 31 16:21:55 GMT+05:30 2001  Weekday: 2  Tue Dec 31 16:21:55 GMT+05:30 2002  Weekday: 3  Wed Dec 31 16:21:55 GMT+05:30 2003  Weekday: 4 |  Calendar Example **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/DateDifferent.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DateComparison.shtml)**  In this section, we are discussing the entire functionality of java.util.Calender class. In the following code segment we are performing various operations on Date object using its methods. The method**getTime()** of the calendar class returns a Date object that is then passed to the println() method just to print today's date. The arithmetic function of the date class adds specific amount of time to the given time field following the calendar's rule. This example subtracts two years from the current date of the calendar and also add 3 days to the current date of the calendar.  **Description of program:**  In the following code segment, first we are getting the current date and time into the date object by using the **getInstance()** and **getTime()** methods of the calendar class in a method**CalendarTimemethod()** and printing this current date and time on the console.  Then we are taking another method **SimpleDateFormatmethod()**, this method is getting an instance of the calendar class into the date object. Now we are taking an object of **SimpleDateFormat** class. Now we are getting the current date and time from the date object by calling **getTime()** method on this object. Now we are passing this current date and time into the **format()** method of the **dateformatter**object (SimpleDateFormat class) just to get the current date and time into the simple date format, after that we are printing current date and time on the console.  Now we are taking another method **Adddates()** that adds two different dates. In this method we are taking an instance **date**of the calendar class into a reference variable **cldr**and an object**dateformatter**of the **SimpleDateFormat** class. Now we are taking the clone of the calendar class and passing the reference of this clone into **cldr**. Now we are getting the dates of two years ago and five years after, by using the **add()** method and printing these values on the console.  Now we are taking another method that is **DateDifference().**In this method we are taking date and time from an object of the GregorianCalendar class and passing it into an object **startDate1**of the Date class and also taking an another object **endDate1** of the date class. Now we are taking the difference of times of these two objects and printing it on the console.  Now in the last **Getcalendermethods()** we are displaying values to demonstrate the various method of the calendar class.  **Here is the code of program:**   |  | | --- | |  |  |  | | --- | | **import**java.util.Date; **import**java.util.Calendar; **import**java.text.SimpleDateFormat; **import**java.util.\*;  **public class**CalendarExample {      **private static void**CalendarTimemethod() {   Date date = Calendar.getInstance().getTime();   System.out.println("Current date and time is: " + date);   System.out.println();   }    **private static void**SimpleDateFormatmethod() {   Calendar date = Calendar.getInstance();   SimpleDateFormat dateformatter = **new**SimpleDateFormat   ("E yyyy.MM.dd 'at' hh:mm:ss a zzz");   System.out.println("Current date and  time in simple date format: "    + dateformatter.format(date.getTime()));   System.out.println();   }    **private static void**Adddates() {    System.out.println("Performing  operations on calendar dates.");    // Get today's date   Calendar date = Calendar.getInstance();   Calendar cldr;   SimpleDateFormat dateformatter = **new**SimpleDateFormat   ("E yyyy.MM.dd 'at' hh:mm:ss a zzz");    cldr = (Calendar) date.clone();   cldr.add(Calendar.DAY\_OF\_YEAR, - (365 \* 2));   System.out.println("Before two years it was: "   + dateformatter.format(cldr.getTime()));    cldr = (Calendar) date.clone();   cldr.add(Calendar.DAY\_OF\_YEAR, + 5);   System.out.println("After five years it will be: "   + dateformatter.format(cldr.getTime()));    System.out.println();   }    **private static void**DateDifference() {      System.out.println("Difference between two dates");   Date startDate1 = **new**GregorianCalendar(2005, 02,  25, 14, 00).getTime();   Date endDate1 = **new**Date();;    **long**diff = endDate1.getTime() - startDate1.getTime();    System.out.println("  Difference between " + endDate1);   System.out.println("  and " + startDate1 + " is "     + (diff /  (1000L\*60L\*60L\*24L)) + " days.");   System.out.println();   }    **private static void**Getcalendermethods() {    System.out.println("Various get methods  of the calendar class:");   Calendar calender = Calendar.getInstance();    System.out.println("Year : "  + calender.get(Calendar.YEAR));   System.out.println("Month  : "  + calender.get(Calendar.MONTH));   System.out.println("Day of Month  : "  + calender.get(Calendar.DAY\_OF\_MONTH));   System.out.println("Day of Week  : "  + calender.get(Calendar.DAY\_OF\_WEEK));   System.out.println("Day of Year  : "  + calender.get(Calendar.DAY\_OF\_YEAR));   System.out.println("Week of Year  : "  + calender.get(Calendar.WEEK\_OF\_YEAR));   System.out.println("Week of Month  : "  + calender.get(Calendar.WEEK\_OF\_MONTH));   System.out.println    ("Day of the Week in Month : "  + calender.get(Calendar.DAY\_OF\_WEEK\_IN\_MONTH));   System.out.println("Hour   : " + calender.get(Calendar.HOUR));   System.out.println("AM PM   : " + calender.get(Calendar.AM\_PM));   System.out.println("Hour of the Day   : " + calender.get(Calendar.HOUR\_OF\_DAY));   System.out.println("Minute    : " + calender.get(Calendar.MINUTE));   System.out.println("Second   : " + calender.get(Calendar.SECOND));   System.out.println();   }    **public static void**main(String[] args) {   System.out.println();   CalendarTimemethod();   SimpleDateFormatmethod();   Adddates();   DateDifference();   Getcalendermethods();   } } |   **Here is the output:**   |  | | --- | | C:\Examples>java CalendarExample  Current date and time is: Mon Dec 10 18:37:06 GMT+05:30 2007  Current date and time in simple date format: Mon 2007.12.10 at 06:37:07 PM GMT+05:30  Performing operations on calendar dates. Before two years it was: Sat 2005.12.10 at 06:37:07 PM GMT+05:30 After five years it will be: Sat 2007.12.15 at 06:37:07 PM GMT+05:30  Difference between two dates Difference between Mon Dec 10 18:37:07 GMT+05:30 2007 and Fri Mar 25 14:00:00 GMT+05:30 2005 is 990 days.  Various get methods of the calendar class: Year : 2007 Month : 11 Day of Month : 10 Day of Week : 2 Day of Year : 344 Week of Year : 50 Week of Month : 3 Day of the Week in Month : 2 Hour : 6 AM PM : 1 Hour of the Day : 18 Minute : 37 |  Date Comparison **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CalendarExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/DateExample.shtml)**  In this section we are discussing the comparison of two dates, whether they are equal or not, the first date comes before the other or not, the first date comes after the other or not, by using the equals(), before() and after() methods of the date class.  **Description of program:**  In this program first we are write the method **DateComparison()** in which we create an object**comparetodaydate**of the Date class that have current date and time and a clone of this object. We also create one more object **comparedate1970**of the Date class that contains a date of 1970. Now we are comparing these two dates whether they are equal or not by using the **equals()** method, comparison of the date contained in the object **comparetodaydate** comes before the date contained in the object **comparedate1970**or not,anddate contained in the object **comparetodaydate** comes after the date contained in the object **comparedate1970**or not and then printing all the values on the console.  **Here is the code  of this program:**      |  | | --- | |  |   import java.util.Date; import java.text.ParseException; import java.text.SimpleDateFormat;  public class DateComparison {   private static void DateComparison() {      Date comparetodaydate  = new Date();     Date comparedate  = (Date) comparetodaydate.clone();     Date comparedate1970  = new Date(24L\*60L\*60L\*1000L);      System.out.println("Comparison of two dates by using the equals() method:");     System.out.println();     System.out.println("Today's date: " + comparetodaydate);     System.out.println("Comparing date: " + comparedate);      if (comparetodaydate.equals(comparedate)) {             System.out.println("The two dates are equal");     }      else {             System.out.println("The two dates are not equal");     }      System.out.println();     System.out.println("Comparison of two dates by using the equals() method:");     System.out.println();     System.out.println("Today's date: " + comparetodaydate);     System.out.println("Compairing date: " + comparedate1970);      if (comparetodaydate.equals(comparedate1970)) {       System.out.println("The two dates are equal");     }      else {             System.out.println("The two dates are not equal");     }          System.out.println();     System.out.println("Comparison of two dates by using the before() method:");     System.out.println();     System.out.println("Today's date: " + comparetodaydate);     System.out.println("Compairing date: " + comparedate1970);      if (comparetodaydate.before(comparedate1970)) {             System.out.println("Today's date comes before the compairing date");     }      else {             System.out.println("Today's date does not come before the comparing date");     }              System.out.println();     System.out.println("Comparison of two dates by using the after() method::");     System.out.println();     System.out.println("Today's date: " + comparetodaydate);     System.out.println("Comparing date: " + comparedate1970);              if (comparetodaydate.after(comparedate1970)) {             System.out.println("Today's date comes after the comparing date");     }      else {             System.out.println("Today's date does not come after the comparing date");             System.out.println();     }     System.out.println();      }      public static void main(String[] args) {         System.out.println();         DateComparison();     } }    **Here is the output of above program:**   |  | | --- | | C:\Examples>java DateComparison  Comparison of two dates by using the equals() method:  Today's date: Mon Dec 10 18:24:59 GMT+05:30 2007 Comparing date: Mon Dec 10 18:24:59 GMT+05:30 2007 The two dates are equal  Comparison of two dates by using the equals() method:  Today's date: Mon Dec 10 18:24:59 GMT+05:30 2007 Comparing date: Fri Jan 02 05:30:00 GMT+05:30 1970 The two dates are not equal  Comparison of two dates by using the before() method:  Today's date: Mon Dec 10 18:24:59 GMT+05:30 2007 Comparing date: Fri Jan 02 05:30:00 GMT+05:30 1970 Today's date does not come before the comparing date  Comparison of two dates by using the after() method::  Today's date: Mon Dec 10 18:24:59 GMT+05:30 2007 Comparing date: Fri Jan 02 05:30:00 GMT+05:30 1970 Today's date comes after the comparing date |  Calendar **[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/DateForms.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/AutoIncrementAndDecrement.shtml)**  In this section we are displaying the days of  a specific month and the calendar format of the days for this month. We are also checking whether the year is leap or not.  **Description of program:**  In the given example first we are taking a string array **monthcalendar** that contains the name of the months of an year. After that we are creating an integer type array that contains number of days in the months in their respective order. Now we create a method **displayMonth().** In this method we are leaving the number of days blank at the start of the month, throwing an illegalArgumentException object showing a message if the month value does not lie in between 0 and 11. Now we are creating an object**cldr**of the Gregoriancalendar class by passing the values of month and year and then printing the days of the week on the console. Now we are getting the number of days to be left blank at the start of a month by using the **get()** method of the **Gregoriancalendar** class and also checking whether the year is leap or not and printing the days of that month on the console.   |  | | --- | |  |  |  | | --- | | **import**java.util.\*; **import**java.text.\*;  **public class**MonthCalender {      **public final static**String[] monthcalender = {   "January", "February", "March", "April", "May", "June", "July",   "August", "September", "October", "November", "December"};      **public final static int**daysinmonths[] = {31, 28, 31, 30, 31,      30, 31, 31, 30, 31, 30, 31 };      **private void**displayMonth(**int**month, **int**year) {      // The number of days to leave blank at   // the start of this month.      **int**blankdays = 0;   System.out.println("  " + monthcalender[month] + " " + year);      **if**(month < 0 || month > 11) {   **throw new**IllegalArgumentException(   "Month " + month + " is not valid and must lie in  between 0 and 11");   }      GregorianCalendar cldr = **new**GregorianCalendar(year,  month, 1);   System.out.println("Sunday Monday Tuesday Wednesday  Thirsday Friday Saturday");      // Compute how much to leave before before the first  day of the month.   // getDay() returns 0 for Sunday.      blankdays = cldr.get(Calendar.DAY\_OF\_WEEK)-1;   **int**daysInMonth = daysinmonths[month];      **if**(cldr.isLeapYear(cldr.get(Calendar.YEAR))  && month == 1) {      ++daysInMonth;   }      // Blank out the labels before 1st day of the month   **for**(**int**i = 0; i < blankdays; i++) {   System.out.print(" ");   }      **for**(**int**i = 1; i <= daysInMonth; i++) {      // This "if" statement is simpler than  messing with NumberFormat   **if**(i<=9) {   System.out.print(" ");   }   System.out.print(i);    **if**((blankdays + i) % 7 == 0) { // Wrap if EOL   System.out.println();   }   **else**{   System.out.print(" ");   }   }   }      /\*\*  \* Sole entry point to the class and application.  \* @param args Array of String arguments.  \*/   **public static void**main(String[] args) {      **int**mon, yr;   MonthCalender moncldr = **new**MonthCalender();    **if**(args.length == 2) {   moncldr.displayMonth(Integer.parseInt(args[0])-1,   Integer.parseInt(args[1]));   }   **else**{   Calendar todaycldr = Calendar.getInstance();   moncldr.displayMonth(todaycldr.get(Calendar.MONTH)  , todaycldr.get(Calendar.YEAR));   }   } } |   **Here is the output:**   |  | | --- | | C:\Examples>java MonthCalender December 2007 Sunday Monday Tuesday Wednesday Thursday Friday Saturday 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 |   C:\convert\rajesh\completed>javac WriteTextFileExample.java  C:\convert\rajesh\completed>java WriteTextFileExample  Your file has been written |

# Garbage Collection

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/ReverseString.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/CheckEmptyStringExample.shtml)**

In the example we are describing how the garbage collector works.

**Description of program:**

In the example given below, first we are creating three objects to give the garbage collector something to do, then we are assigning some values to these variables and then appending values to the **StringBuffer** variable **strbuf**and setting the null values to these objects so that they can be garbage collected. Now we are taking the system's current time in milliseconds into a long type variable then calling the garbage collector method **gc()**of the System class and then system's current time after the garbage collection operation completes and subtracts the previous time from the last one to determine the time taken by the garbage collector to complete its operation and prints this time on the console.

**Here is the code of this program:**

|  |
| --- |
|  |

|  |
| --- |
| **import**java.util.Vector;  **public class**GarbageCollection {     **public static void**main(String[] args) {    **int**size = 5000;   StringBuffer strbuf;   Vector vctr;      **for**(**int**i = 0; i < size; i++) {   strbuf = **new**StringBuffer(70);   vctr = **new**Vector(50);   strbuf.append(i).append(i+1).append(i+2).append(i+3);   }    strbuf= **null**;   vctr = **null**;   System.out.println("Staring explicit Garbage Collection.");   **long**time = System.currentTimeMillis();   System.gc();    System.out.println("Garbage Collection took " +    (System.currentTimeMillis()-time) + " ms");    **int**[] arr = **new int**[size\*10];   // null out the variable so that the array can be garbage collected   time = System.currentTimeMillis();   arr = **null**;   System.out.println("Staring explicit Garbage Collection.");   System.gc();    System.out.println("Garbage Collection took " +      (System.currentTimeMillis()-time) + " ms");     } } |

**Here is the output:**

|  |
| --- |
| C:\Examples>java GarbageCollection Staring explicit Garbage Collection. Garbage Collection took 984 ms Staring explicit Garbage Collection. Garbage Collection took 16 ms |

[**Download of this program.**](http://www.roseindia.net/java/beginners/GarbageCollection.java)

# Check Empty String

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/GarbageCollection.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/static-acess-object.shtml)**

In this example we are checking a sting object containing empty or null value. Apache has provided two methods**isBlank()** and **isNotBlank()** for checking the strings..

The class**org.apache.commons.lang.StringUtils** extends **Object** class and  provides methods that operate on null safe string.

The methods used: **isBlank(String str):**This method is used to check whether a string is empty ("") , null or whitespace.

**isNotBlank(String str):**This method is used to check whether a string is not empty (""), not null or not a whitespace .

There are two more methods which can be used:  
**isNotEmpty(String str):** Use this method for checking a string whether it is not empty("") or not null.

**isEmpty(String str):**Use this method to check a string for its empty (" ) or null value.  
  
 **The code of the program is given below:**

|  |
| --- |
| **import**org.apache.commons.lang.StringUtils;   **public class**CheckEmptyStringExample  {     **public static void**main(String[] args)   {   String string1 = "";   String string2 = "\t\r\n";   String string3 = " ";   String string4 = **null**;   String string5 = "Hi";    System.out.println("\nString one is empty? " +  StringUtils.isBlank(string1));   System.out.println("String one is not empty? " +  StringUtils.isNotBlank(string1));   System.out.println("\nString two is empty? " +  StringUtils.isBlank(string2));   System.out.println("String two is not empty?" +  StringUtils.isNotBlank(string2));   System.out.println("\nString three is empty?" +  StringUtils.isBlank(string3));   System.out.println("String three is not empty?" + StringUtils.isNotBlank(string3));   System.out.println("\nString four is empty?" +  StringUtils.isBlank(string4));   System.out.println("String four is not empty?" +  StringUtils.isNotBlank(string4));   System.out.println("\nString five is empty?" +  StringUtils.isBlank(string5));   System.out.println("String five is not empty?" +  StringUtils.isNotBlank(string5));    } } |

**The output of the program is given below:**

|  |
| --- |
| C:\rajesh\kodejava>javac CheckEmptyStringExample.java  C:\rajesh\kodejava>java CheckEmptyStringExample  String one is empty? true  String one is not empty? false  String two is empty? true  String two is not empty?false  String three is empty?true  String three is not empty?false  String four is empty?true  String four is not empty?false  String five is empty?false  String five is not empty?true |

[**Download this example.**](http://www.roseindia.net/java/beginners/check-empty-string-example.java)

**Access Static Member Of The Class Through Object**Posted on: June 27, 2008 at 12:00 AM

**Static methods are special type of methods that work without any object of the class.**

# Access Static Member Of The Class Through Object

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CheckEmptyStringExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/cmnd-line-arguments.shtml)**

Static methods are special type of methods that work without any object of the class. Static methods are limited to calling other static methods in the class and to using only static variables. The following example shows both ways to access static method (through simply class name and through object).

**StaticMemberClass.java**

|  |
| --- |
| /\*\*    \* **Access static member of the class through object.**    \*/    import java.io.\*;     class StaticMemberClass {   // Declare a static method.   public static void staticDisplay() {    System.out.println("This is static method.");    }    // Declare a non static method.    public void nonStaticDisplay() {    System.out.println("This is non static method.");    }   }     class StaticAccessByObject {     public static void main(String[] args) throws IOException {     // call a static member only by class name.    StaticMemberClass.staticDisplay();    // Create object of StaticMemberClass class.    StaticMemberClass obj = new StaticMemberClass();    // call a static member only by object.    obj.staticDisplay();    // accessing non static method through object.    obj.nonStaticDisplay();   }    } |
| **Output of the program:** |
| This is static method.  This is static method.  This is non static method. |

**[Download source code](http://www.roseindia.net/java/beginners/StaticMemberClass.java)**

# Read a string and reverse it and then print in alphabetical order.

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/cmnd-line-arguments.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/ArraysToCollectionExample.shtml)**

A string is a contiguous sequence of symbols or values, such as a character string (a sequence of characters) or a binary digit string (a sequence of binary values). In the example given below takes string value from user and reverse that string by using String reverse(String str) method and set in alphabetical order by String alphaOrder(String str) method. Note that passed string by user must be in upper case or in lower case but mixed cases are not permitted for this example.

Program asks the user to enter the input string. Once it is entered by the user, the program performs the reverse operations.

**Program: Read a string reverse it and arrange in alphabetical order.**

ReverseAlphabetical.java

|  |
| --- |
| /\*\*   \* read a string and reverse it and then write in alphabetical order.    \*/    import java.io.\*;    import java.util.\*;     class ReverseAlphabetical {      String reverse(String str) {    String rStr = new StringBuffer(str).reverse().toString();    return rStr;    }     String alphaOrder(String str){    char[] charArray = str.toCharArray();    Arrays.sort(charArray);    String aString = new String(charArray);   return aString ;    }     public static void main(String[] args) throws IOException {   System.out.print("Enter the String : ");    BufferedReader br =new BufferedReader(new InputStreamReader(System.in));   String inputString = br.readLine();   System.out.println("String before reverse : " + inputString);   ReverseAlphabetical obj = new ReverseAlphabetical();   String reverseString = obj.reverse(inputString);   String alphaString = obj.alphaOrder(inputString);   System.out.println("String after reverse : " + reverseString);   System.out.println("String in alphabetical order : " + alphaString);    }   } |

**Output of the program:**

|  |
| --- |
| Enter the String : mahendra String before reverse : mahendra String after reverse : ardneham String in alphabetical order : aadehmnr |

# Array to Collection

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/CheckEmptyStringExample.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/copy-file.shtml)**

In this example we are converting values of an array into collection.

**List**  interface is a member of the Java**Collection** Framework and extends**Collection** interface. **List** interface is an ordered collection which follows insertion order, typically allow duplicate elements. The List interface provides a special iterator, called ListIterator for insertion, replacement and  bi-directional access.   
The **Arrays** class contains various methods for manipulating arrays (e.g. sorting and searching etc.) and extends the **Object**class. The **Iterator**class allows access to the elements of a collection.

Methods used: **hasNext():**This method is used to get more elements. If the element at the next position is found then returns true otherwise throws an exception.

**next():** This method returns the next element in the iteration.

**asList(Object[] a):**This method acts as a bridge between array and collection. This method converts array into Arrays by internally using  **Colletion.toArray()**method.

**The code of the program is given below:**

|  |
| --- |
| **import**java.util.Arrays; **import**java.util.List; **import**java.util.Iterator;   **public class**ArraysToCollection{   **public static void**main(String[] args){   String[] array = {"Java", "Struts", "JSP", "J2EE"};   List list = Arrays.asList(array);   Iterator iterator = list.iterator();   System.out.println("\nThe values into Array");   **while**(iterator.hasNext()){    System.out.println((String) iterator.next());  }  } } |

**The output of the program is given below:**

|  |
| --- |
| C:\rajesh\kodejava>javac ArraysToCollection.java  C:\rajesh\kodejava>java ArraysToCollection  The values into Array  Java  Struts  JSP  J2EE |

# Introduction to Java Arrays

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/structure_of_java_arrays.shtml)**

In this section you will be introduced to the concept of Arrays in Java Programming language. You will learn how the Array class in java  helps the programmer to organize the same type of data into easily manageable format.

Program data is stored in the variables and takes the  memory spaces, randomly. However, when we need the data of the same type  to store in the contiguous memory allocations we use the data structures like arrays. To meet this feature java has provided an Array class which abstracts the array data-structure.

The java array enables the user to store the values of the same type in contiguous memory allocations. Arrays are always a fixed length abstracted data structure  which can not be altered when required.

The **Array** class implicitly extends **java.lang.Object**so an array is an instance of Object.

**Advantages of Java Array:**

1. An array can hold primitive types data.
2. An array has its size that is known as array length.
3. An array knows only its type that it contains. Array type is checked at the compile-time.

**Disadvantages of Java Array:**

1. An array has fixed size.
2. An array holds only one type of data (including primitive types).

# Structure of Java Arrays

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/introduction_to_java_arrays.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/java_array_declaration.shtml)**

Now lets study the structure of Arrays in java. Array is the most widely used data structure in java. It can contain multiple values of the same type. Moreover, arrays are always of fixed length i.e. the length of an array cannot be increased or decreased.

Lets have a close look over the structure of Array. Array contains the values which are implicitly  referenced through the index values. So to access the stored values in an array we use indexes. Suppose an array contains "n"  integers. The first element of this array will  be indexed with the "0" value and the last integer will be referenced by "n-1" indexed value.

Presume an array  that contains 12 elements as shown  in the figure. Each element is holding a distinct value. Here the first element is refrenced by a[0] i.e. the first  index value. We have filled the 12 distinct values in the array each referenced as:

a[0]=1

a[1]=2

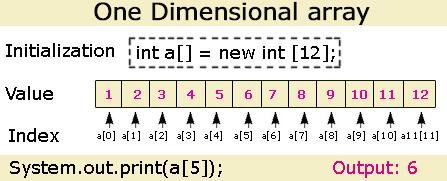
...

a[n-1]=n

...

a[11]=12

The figure below shows the structure of an Array more precisely.



# Java Array Declaration

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/structure_of_java_arrays.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/java_array_initialization.shtml)**

As we declare a variable in Java, An Array variable is declared the same way. Array variable has a type and a valid Java identifier i.e. the array's type and the array's name. By type we mean the type of elements contained in an  array. To represent the variable as an Array, we use [] notation. These two brackets are used to hold the array of a variable.

By array's name, we mean that we can give any name to the array, however it should follow the predefined conventions. Below are the examples which show how to declare an array :-

|  |
| --- |
| int[] array\_name;   //declares an array of integers String[] names; int[][] matrix;  //this is an array of arrays |

It is essential to assign memory to an array when we declare it. Memory is assigned to set the size of the declared array. for example:

|  |
| --- |
| int[] array\_name = new int[5]; |

Here is an example that creates an array that has 5 elements.

|  |
| --- |
| public class Array {  public static void main(String[] args)  {   int[] a = new int[5];  } } |

# ava Array Initialization

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/java_array_declaration.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/java_array_usage.shtml)**

After declaring an array variable, memory is allocated to it. The "new" operator is used for the allocation of  memory to the array object. The correct way to use the "new" operator is

 String names[];

 names = new String[10];

Here, the new operator is followed by the type of variable and the number of elements to be allocated. In this example [] operator has been used to place the number of elements to be allocated.

Lets see a simple example of an array,

|  |
| --- |
| **public class**Sum  {   **public static void**main(String[] args)    {   **int**[] x = **new int**[101];   **for**(**int**i = 0; i<x.length; i++ )   x[i] = i;   **int**sum = 0;   **for**(**int**i = 0; i<x.length; i++)   sum += x[i];   System.out.println(sum);   } } |

In this example, a variable 'x' is declared which  has a type array of int, that is, int[]. The variablex is initialized to reference a newly created array object. The expression 'int[] = new int[50]' specifies that the array should have 50 components. To know the length of the Array, we use field length, as shown.

**Output for the given program:**

|  |
| --- |
| **C:\tamana>javac Sum.java  C:\tamana>java Sum 5050  C:\tamana>** |

# Java Array Usage

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/java_array_initialization.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/copying_arrays.shtml)**

We have already discussed that to refer an element within an array, we use the [] operator. The [] operator takes an "int" operand and returns the element at that index. We also know that the array indices start with zero, so the first element will be held by the 0 index. For Example :-  
int month = months[4];  //get the 5th month (May)

Most of the times it is not known in the program that which elements are of interest in an array. To find the elements of interest in the program, it is required that the program must run a loop through the array. For this purpose "for" loop is used to examine each element in an array. For example :-

|  |
| --- |
| String months[] =  {"Jan", "Feb", "Mar", "Apr", "May", "Jun",  "July", "Aug", "Sep", "Oct", "Nov", "Dec"};  //use the length attribute to get the number  //of elements in an array  for (int i = 0; i < months.length; i++ ) {  System.out.println("month: " + month[i]); |

Here, we have taken an array of months which is,

   String months[] =   
 {"Jan", "Feb", "Mar", "Apr", "May", "Jun",   
  "July", "Aug", "Sep", "Oct", "Nov", "Dec"};

Now, we run a for loop to print each element individually starting from the month "Jan".

  for (int i = 0; i < months.length; i++ )

In this loop **int i = 0;** indicates that the loop starts from the 0th position of an array and goes upto the last position which is length-1, **i < months.length;**indicates the length of the array and **i++** is used for the increment in the value of i which is i = i+1.

# Copying Arrays

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/java_array_usage.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/two.shtml)**

After learning all about arrays, there is still one interesting thing left to learn i.e. copying arrays. It means to copy data from one array to another. The precise way to copy data from one array to another is

|  |
| --- |
| public static void arraycopy(Object source,  int srcIndex,  Object dest,  int destIndex,  int length) |

Thus apply system's **arraycopy** method for copying arrays.The parameters being used are :-

|  |
| --- |
| src  the source array srcIndex start position (first cell to copy) in the source array dest  the destination array destIndex start position in the destination array length the number of array elements to be copied |

The following program, ArrayCopyDemo(in a .java source file), uses **arraycopy** to copy some elements from the copyFrom array to the copyTo array.

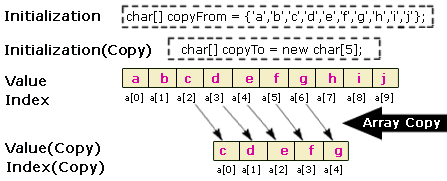
|  |
| --- |
| **public class**ArrayCopyDemo{   **public static void**main(String[] args){   **char**[] copyFrom = {'a','b','c','d','e','f','g','h','i','j'};   **char**[] copyTo = **new char**[5];   System.arraycopy(copyFrom, 2, copyTo, 0, 5);   System.out.println(**new**String (copyTo));   } } |

**Output of the program:**

|  |
| --- |
| **C:\tamana>javac ArrayCopyDemo.java**  **C:\tamana>java ArrayCopyDemo**  **cdefg**  **C:\tamana>** |

In this example the array method call begins the copy of elements from element number 2. Thus the copy begins at the array element 'c'. Now, the **arraycopy** method takes the copied element and puts it into the destination array. The destination array begins at the first element (element 0) which is the destination array **copyTo**. The copyTo copies 5 elements : 'c', 'd', 'e', 'f', 'g'. This method will take "cdefg" out of "abcdefghij", like this :

Following image illustrates the procedure of copying array from one to another.



# Two-Dimensional Arrays

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/copying_arrays.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/multi.shtml)**

Two-dimensional arrays are defined as "an array of arrays". Since an array type is a first-class Java type, we can have an array of ints, an array of Strings, or an array of Objects. For example, an array of ints will have the type int[]. Similarly we can have int[][], which represents an "array of arrays of  ints". Such an array is said to be a two-dimensional array.   
The command

  int[][] A = new int[3][4];

declares a variable, A, of type int[][], and it initializes that variable to refer to a newly created object. That object is an array of arrays of ints. Here, the notation int[3][4] indicates that there are 3 arrays of ints in the array A, and that there are 4 ints in each of those arrays.  
To process a two-dimensional array, we use **nested for loops**. We already know about for loop. A loop in a loop is called a **Nested loop**. That means we can run another loop in a loop.

Notice in the following example how the rows are handled as separate objects.

|  |
| --- |
| Code: Java  int[][] a2 = new int[10][5];  // print array in rectangular form  for (int i=0; i<a2.length; i++) {  for (int j=0; j<a2[i].length; j++) {  System.out.print(" " + a2[i][j]);  }  System.out.println("");  } |

In this example, "int[][] a2 = new int[10][5];" notation shows a two-dimensional array. It declares a variable a2 of type int[][],and it initializes that variable to refer to a newly created object. The notation int[10][5] indicates that there are 10 arrays of ints in the array a2, and that there are 5 ints in each of those arrays.

**Here is the complete code of the example:**

|  |
| --- |
| **public class**twoDimension{   **public static void**main(String[] args) {   **int**[][] a2 = **new int**[10][5];   **for**(**int**i=0; i<a2.length; i++) {   **for**(**int**j=0; j<a2[i].length; j++) {   a2[i][j] = i;   System.out.print(" " + a2[i][j]);   }   System.out.println("");   }   } } |

**Here is the output for the program:**

|  |
| --- |
| **C:\tamana>javac twoDimension.java**  **C:\tamana>java twoDimension**  **0 0 0 0 0**  **1 1 1 1 1**  **2 2 2 2 2**  **3 3 3 3 3**  **4 4 4 4 4**  **5 5 5 5 5**  **6 6 6 6 6**  **7 7 7 7 7**  **8 8 8 8 8**  **9 9 9 9 9**  **C:\tamana>\_** |

# Multi-dimensional arrays

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/two.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/bubbleSort.shtml)**

So far we have studied about the one-dimensional and two-dimensional arrays. To store data in more dimensions a multi-dimensional array is used. A multi-dimensional array of dimension n is a collection of items. These items are accessed via **n** subscript expressions. For example, in a language that supports it, the element of the two-dimensional array x is denoted by x[i,j].   
The Java programming language does not really support multi-dimensional arrays. It does, however, supports an array of arrays. In Java, a two-dimensional array 'x' is an array of one-dimensional array. For instance :-

  int[][] x = new int[3][5];

The expression x[i] is used to select the one-dimensional array; the expression x[i][j] is ued to select the element from that array. The first element of this array will  be indexed with the "0" value and the last integer will be referenced by "length-1" indexed value. There is no array assignment operator.

# Bubble Sorting in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/multi.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/BidirectionalBubbleSort.shtml)**

**Introduction**

In this example we are going to sort integer values of an array using bubble sort.  
  
Bubble sort is also known as exchange sort. Bubble sort is a simplest sorting algorithm. In bubble sort algorithm array is traversed from 0 to the length-1 index of the array and compared one element to the next element and swap values in between if the next element is less than the previous element. In other words, bubble sorting algorithm compare two values and put the largest value at largest index. The algorithm follow the same steps repeatedly until the values of array is sorted. In worst-case the complexity of bubble sort is O(n2) and in  best-case the complexity of bubble sort is Ω(n).  
     
**Code description:**

Bubble Sorting is an algorithm in which we are comparing first two values and put the larger one at higher index. Then we take next two values compare these values and place larger value at higher index. This process do iteratively until the largest value is not reached at last index. Then start again  from zero index up to n-1 index. The algorithm follows the same steps iteratively unlit elements are not sorted.   
  
**Working of bubble sort** **algorithm:**

Say we have an array unsorted  A[0],A[1],A[2]................ A[n-1] and A[n] as input. Then the following steps are followed by bubble sort algorithm to sort the values of an array.  
 1.Compare A[0] and A[1] .   
 2.If A[0]>A[1] then Swap A[0] and A[1].  
 3.Take next A[1] and A[2].  
 4.Comapre these values.  
 5.If A[1]>A[2]  then swap A[1] and A[2]  
 ...............................................................  
................................................................  
at last compare A[n-1] and A[n]. If A[n-1]>A[n] then swap A[n-1] and A[n]. As we see the highest value is reached at nth position. At next iteration leave nth value. Then apply the same steps repeatedly on A[0],A[1],A[2]................ A[n-1] elements repeatedly until the values of array is sorted.  
  
In our example we are taking the following array values  
12 9 4 99 120 1 3 10  
  
The basic steps followed by algorithm:-  
     
In the first step compare first two values 12 and 9.  
**12** **9** 4 99 120 1 3 10  
  
As 12>9 then we have to swap these values   
Then the new sequence will be   
***9*** ***12*** 4 99 120 1 3 10  
  
In next step take next  two values 12 and 4   
9 **12** **4** 99 120 1 3 10  
  
Compare these two values .As 12>4 then we have to swap these values.   
Then the new sequence will be  
9 ***4 12*** 99 120 1 3 10  
  
We have to follow similar steps up to end of array. e.g.  
9 4 **12** **99** 120 1 3 10  
9 4 12 **99** **120** 1 3 10  
9 4 12 99 **1** **120** 3 10   
9 4 12 99 1 **120** **3** 10  
9 4 12 99 1 3 120 10  
9 4 12 99 1 3 10 120  
  
When we  reached at last index .Then restart same steps unlit the data is not sorted.  
  
The output of this example will be :  
1 3 4 9 10 12 99 120

**The code of the program :**

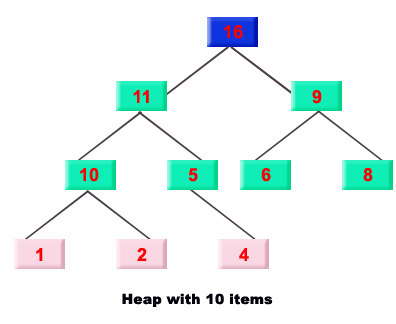
|  |
| --- |
| **public class**bubbleSort{   **public static void**main(String a[]){   **int**i;   **int**array[] = {12,9,4,99,120,1,3,10};   System.out.println("Values Before the sort:\n");   **for**(i = 0; i < array.length; i++)   System.out.print( array[i]+"  ");   System.out.println();   bubble\_srt(array, array.length);   System.out.print("Values after the sort:\n");   **for**(i = 0; i <array.length; i++)   System.out.print(array[i]+"  ");   System.out.println();   System.out.println("PAUSE");   }    **public static void**bubble\_srt( **int**a[], **int**n ){   **int**i, j,t=0;   **for**(i = 0; i < n; i++){   **for**(j = 1; j < (n-i); j++){   **if**(a[j-1] > a[j]){   t = a[j-1];   a[j-1]=a[j];   a[j]=t;   }   }   }   } } |

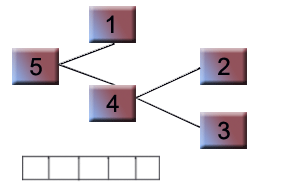
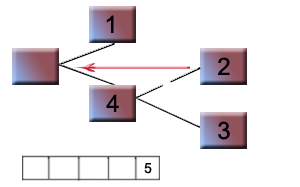
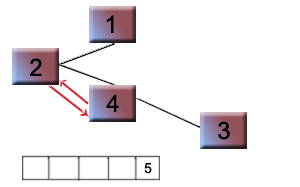
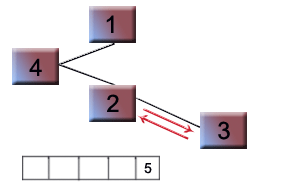
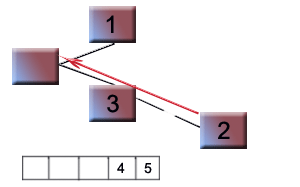
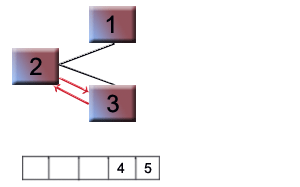
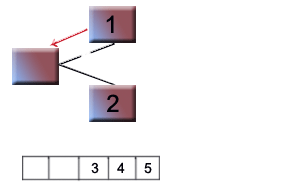
# Heap Sort in Java

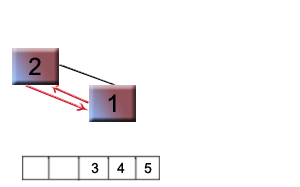
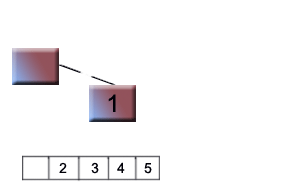
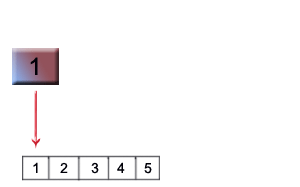
**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/extraStorageMergeSort.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/InsertionSort.shtml)**

**Introduction**

In this example we are going to sort integer values of an array using heap sort.  
  
There are two types of heap. First one is  Max heap and second one is min heap. Max heap is a special type of binary tree .The roots of  the max heap is greater than its child roots. Other heap is min heap it is also a special type of heap which has minimum root than his child. We can sort the array values using heap sorting algorithm. In this algorithm the heap build is used to rebuild the heap.   
In this example we sorting all elements of an array. The complexity   of the heap sort is O(n.log(n)). Heap sort is slowest  but it is better option for large data sets.  
  
The example of Max heap:



**Code description:**   
To sort a heap  Build Heap algorithm is used to build a heap out of the data set .Then remove the root element and replace the last element at the position of root node. Then rearrange the heap. Place the root node in an array. Follow these steps until all elements in heap is not replace into array. The values in array will be in sorted order.   
  
**Steps of heap sort algorithm:**1. Remove the parent root and replace it with the rightmost leaf.   
2.Strore parent root in an array.   
3. Re-establish the heap.   
4. Repeat steps 1 and 3  until values in heap is not zero.   
 **Working of heap sort algorithm:  
Input:1,3,5,4,2  
Step1:**Buid Heap tree and an array of same size. **  
  
  
Step2:**Remove largest root and  add largest root in array. **  
Step3:**Replace last value (eg 2) at at root node position.**  
Step4:**Swap 2 and 4 **  
Step5:**Swap 2 and 3. **  
Step6:**Remove 4 and replace 2 at position of 4 and add 4 in array **   
Step7:**Swap 2 and 3 **  
Step8:**Remove 3 ,add in array and replace 1 at position of 3.

**  
Step9:**Swap 2 and 1. **  
Step10:**Remove 2 and it at root position **   
Step11:**Remove 1 and add in array. **Output:**Sorted array 1,2,3,4,5

**The code of the program :**

|  |
| --- |
| **public class**heap\_Sort{   **public static void**main(String a[]){   **int**i;   **int**arr[] = {1,3,4,5,2};    System.out.println("\n  Heap Sort\n---------------\n");   System.out.println("\n  Unsorted Array\n\n");   **for**(i = 0; i < arr.length; i++)   System.out.print(" "+arr[i]);   **for**(i=arr.length; i>1; i--){   fnSortHeap(arr, i - 1);   }   System.out.println("\n  Sorted array\n---------------\n");   **for**(i = 0; i < arr.length; i++)   System.out.print(" "+arr[i]);   }    **public static void**fnSortHeap(**int**array[], **int**arr\_ubound){   **int**i, o;   **int**lChild, rChild, mChild, root, temp;   root = (arr\_ubound-1)/2;    **for**(o = root; o >= 0; o--){   **for**(i=root;i>=0;i--){   lChild = (2\*i)+1;   rChild = (2\*i)+2;   **if**((lChild <= arr\_ubound) && (rChild <= arr\_ubound)){   **if**(array[rChild] >= array[lChild])   mChild = rChild;   **else**   mChild = lChild;   }   **else**{   **if**(rChild > arr\_ubound)   mChild = lChild;   **else**   mChild = rChild;   }    **if**(array[i] < array[mChild]){   temp = array[i];   array[i] = array[mChild];   array[mChild] = temp;   }   }   }   temp = array[0];   array[0] = array[arr\_ubound];   array[arr\_ubound] = temp;   **return**;   } } |

**Output of the example:**

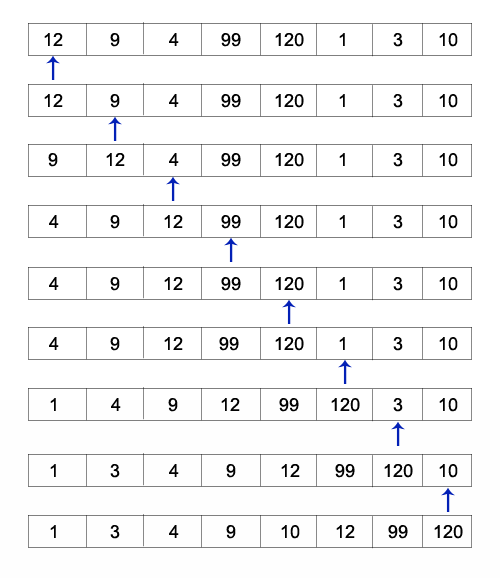
|  |
| --- |
| **C:\array\sorting>Javac heap\_Sort.java**  **C:\array\sorting>java heap\_Sort**  **Heap Sort**  **---------------**  **Unsorted Array**  **1 3 4 5 2**  **Sorted array**  **---------------**  **1 2 3 4 5**  **C:\array\sorting>\_** |

# Insertion Sort In Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/heapSort.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/mergeSort.shtml)**

**Introduction**

In this example we are going to sort integer values of an array using insertion sort.  
  
Insertion sorting algorithm is similar to bubble sort. But insertion sort is more  efficient than bubble sort because in insertion sort the elements comparisons are less as compare to bubble sort. In insertion sorting algorithm compare the value until  all the prior elements are lesser than compared value is not found. This mean that the all previous values are lesser than compared value. This algorithm is more efficient than the bubble sort .Insertion sort is a good choice for small values and for nearly-sorted values. There are more efficient algorithms such as quick sort, heap sort, or merge sort for large values .  
**Positive feature of insertion sorting:**  
1.It is simple to implement   
2.It is efficient on (quite) small data values   
3.It is efficient on data sets which are already nearly sorted.  
  
The complexity of insertion sorting is O(n) at best case of an already sorted array and  O(n2) at worst case .  
     
**Code description:**In insertion sorting take the element form left assign value into a variable. Then compare the  value with  previous values. Put  value so that values must be lesser than the previous values. Then assign  next  value to a variable and follow the same steps relatively until the comparison not reached to end of array.     
**Working of insertion sorting:**

****

**The code of the program :**

|  |
| --- |
| **public class**InsertionSort{   **public static void**main(String a[]){   **int**i;   **int**array[] = {12,9,4,99,120,1,3,10};   System.out.println("\n\n RoseIndia\n\n");   System.out.println(" Selection Sort\n\n");    System.out.println("Values Before the sort:\n");     **for**(i = 0; i < array.length; i++)   System.out.print( array[i]+"  ");   System.out.println();   insertion\_srt(array, array.length);     System.out.print("Values after the sort:\n");     **for**(i = 0; i <array.length; i++)   System.out.print(array[i]+"  ");   System.out.println();    System.out.println("PAUSE");    }    **public static void**insertion\_srt(**int**array[], **int**n){   **for**(**int**i = 1; i < n; i++){   **int**j = i;   **int**B = array[i];   **while**((j > 0) && (array[j-1] > B)){   array[j] = array[j-1];   j--;   }   array[j] = B;   }   } } |

**Output of the example:**

|  |
| --- |
| **C:\array\sorting>javac InsertionSort.java**  **C:\array\sorting>java InsertionSort**  **RoseIndia**  **Selection Sort**  **Values Before the sort:**  **12 9 4 99 120 1 3 10**  **Values after the sort:**  **1 3 4 9 10 12 99 120**  **PAUSE**  **C:\array\sorting>\_** |

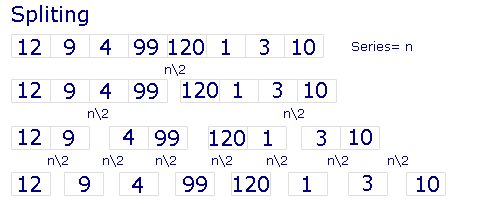
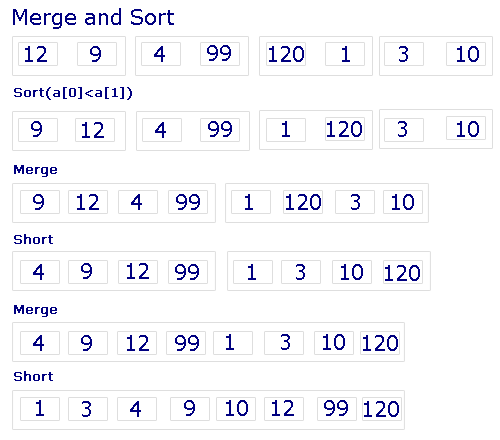
[**Download this example.**](http://www.roseindia.net/java/beginners/arrayexamples/InsertionSort.java)

# Merge Sort in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/InsertionSort.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/extraStorageMergeSort.shtml)**

**Introduction**

In this example we are going to sort integer values of an array using merge sort.  
  
In merge sorting algorithm unsorted values are divided into two equal parts iteratively. Then merge both parts and sort it. Then again merge the next part and sort it. Do it iteratively until  the values are not in sorted order. In merge sorting the number of elements must be even. The merge sorting is invented by John von Neumann in 1945 .  
The complexity of the merge sorting is in worst-case O(n log n) and in average case O(n log n).  
     
**Code description:**In merge sort split the array values in halves recursively until each half has only single  element. Merge the two 1/2 values together and sort the values. Do same steps iteratively until the values are not sorted.   
  
**Working of merge sort** **algorithm:**

Say we have an array unsorted  A[0],A[1],A[2]................ A[n-1] and A[n] as input. Then the following steps are followed by merge sort algorithm to sort the values of an array.  
  
**Step1:**Spliting the values of array  
Divide the values into two equal 1/2  
   A[0],A[1],A[2].........A[n/2-1]   &  A[n/2]....... .................A[n-1], A[n]   
  
Again divide two equal 1/2  
A[0] a[1]A[2]..............A[(n/2-1)/2-1] &  A[(n/2-1)/2]............A[n/2-1],   
A[n/2].............A[(2n-1)/2-1]  & a[(2n-1)/2].............A[n-1],A[n]   
..........................................................................................................................  
..........................................................................................................................  
........................................................................................................................  
A[0] & A[1] & A[2]& A[3],..............................................................A[n-1]& A[n]  
   
**Step2:**Merge two values  and sort the values  
  
A[0],A[1] & A[2],A[3]&..................................................................&A[n-1],A[n]  
If A[1]<A[0],A[]<A[3]........................................................................A[n-1]>A[n]  
then  
A[1]A[0],A[2]A[3],...............................................................................A[n]A[n-1]  
**Step3:**Merge four values and sort the values  
A[2] A[1] A[0] A[3],...................................................................................A[n-1]  
..................................................................................................................  
..................................................................................................................  
.................................................................................................................  
**Step3:**Merge n values and sort the values  
A[2]A[6]......................................................................................................A[n-5]  
  
Where n must be even number.  
  
**Steps of Merge Sort:**  
Say unsorted  an array values are:  
12,9,4,99,120,1,3,10  
  


**The code of the program :**

|  |
| --- |
| **public class**mergeSort{   **public static void**main(String a[]){   **int**i;   **int**array[] = {12,9,4,99,120,1,3,10};   System.out.println("\n\n RoseIndia\n\n");   System.out.println(" Selection Sort\n\n");   System.out.println("Values Before the sort:\n");   **for**(i = 0; i < array.length; i++)   System.out.print( array[i]+"  ");   System.out.println();   mergeSort\_srt(array,0, array.length-1);   System.out.print("Values after the sort:\n");   **for**(i = 0; i <array.length; i++)   System.out.print(array[i]+"  ");   System.out.println();   System.out.println("PAUSE");   }    **public static void**mergeSort\_srt(**int**array[],int lo, **int**n){   **int**low = lo;   **int**high = n;   **if**(low >= high) {   **return**;   }    **int**middle = (low + high) / 2;   mergeSort\_srt(array, low, middle);   mergeSort\_srt(array, middle + 1, high);   **int**end\_low = middle;   **int**start\_high = middle + 1;   **while**((lo <= end\_low) && (start\_high <= high)) {   **if**(array[low] < array[start\_high]) {   low++;   } **else**{   **int**Temp = array[start\_high];   **for**(**int**k = start\_high- 1; k >= low; k--) {   array[k+1] = array[k];   }   array[low] = Temp;   low++;   end\_low++;   start\_high++;   }   }   }   } |

**Output of the example:**

|  |
| --- |
| **C:\array\sorting>javac mergeSort.java**  **C:\array\sorting>java mergeSort**  **RoseIndia**  **Selection Sort**  **Values Before the sort:**  **12 9 4 99 120 1 3 10**  **Values after the sort:**  **1 3 4 9 10 12 99 120**  **PAUSE**  **C:\array\sorting>\_** |

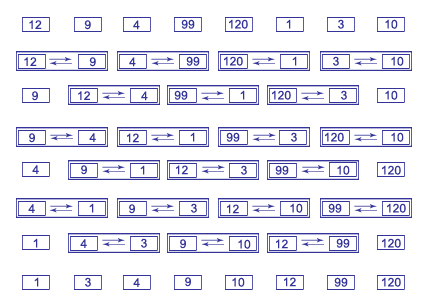
# Odd Even Transposition Sort In Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/extraStorageMergeSort.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/QuickSort.shtml)**

**Introduction**

In this example we are going to sort integer values of an array using odd even transposition sort.  
  
Odd even transposition sort is a parallel sorting algorithm. Odd Even is based on the Bubble Sort technique of comparing two numbers and swapping them and put higher value at larger index .In each parallel computational steps can pair off either the odd or even neighboring pairs. Each number (In Processing Element-PE) would look to it's right neighbor and if it were greater, it would swap them.   
  
**Code description:**The odd even transposition sortis a parallel sorting algorithm. That mean more than one compression can made at one iteration. The comparison is same as bubble sort.

**Working of odd even transposition sort:**

**  
The code of the program :**

|  |
| --- |
| **public class**OddEvenTranspositionSort{   **public static void**main(String a[]){   **int**i;   **int**array[] = {12,9,4,99,120,1,3,10,13};      System.out.println("\n\n RoseIndia\n\n");   System.out.println(" Odd Even Transposition Sort\n\n");   System.out.println("Values Before the sort:\n");   **for**(i = 0; i < array.length; i++)   System.out.print( array[i]+"  ");   System.out.println();   odd\_even\_srt(array,array.length);   System.out.print("Values after the sort:\n");   **for**(i = 0; i <array.length; i++)   System.out.print(array[i]+"  ");   System.out.println();   System.out.println("PAUSE");   }    **public static void**odd\_even\_srt(**int**array[],int n){   **for**(**int**i = 0; i < n/2; i++ ) {   **for**(**int**j = 0; j+1 < n; j += 2)   **if**(array[j] > array[j+1]) {   **int**T = array[j];   array[j] = array[j+1];   array[j+1] = T;   }   **for**(**int**j = 1; j+1 < array.length; j += 2)   **if**(array[j] > array[j+1]) {   **int**T = array[j];   array[j] = array[j+1];   array[j+1] = T;   }   }   } } |

**Output of the example:**

|  |
| --- |
| **C:\array\sorting>javac OddEvenTranspositionSort.java**  **C:\array\sorting>java OddEvenTranspositionSort**  **RoseIndia**  **Odd Even Transposition Sort**  **Values Before the sort:**  **12 9 4 99 120 1 3 10 13**  **Values after the sort:**  **1 3 4 9 10 12 13 99 120**  **PAUSE**  **C:\array\sorting>\_** |

[**Download this example.**](http://www.roseindia.net/java/beginners/arrayexamples/OddEvenTranspositionSort.java)

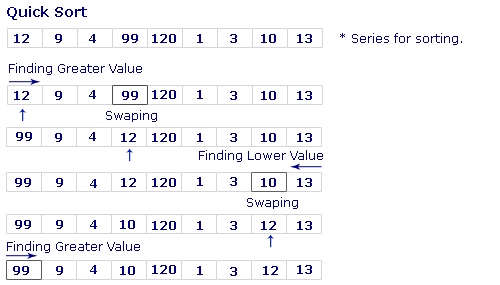
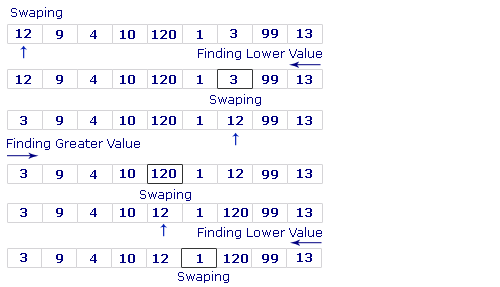
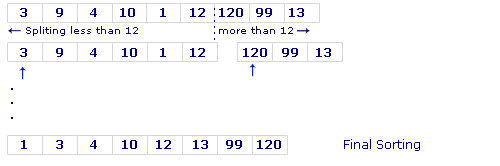
# Quick Sort in Java

**[http://www.roseindia.net/images/previous.gif](http://www.roseindia.net/java/beginners/arrayexamples/OddEvenTranspositionSort.shtml)**    **[http://www.roseindia.net/images/bt_home.gif](http://www.roseindia.net/java/beginners/arrayexamples/index.shtml)**  **[http://www.roseindia.net/images/next.gif](http://www.roseindia.net/java/beginners/arrayexamples/SelectionSort.shtml)**

**Introduction**

In this example we are going to sort integer values of an array using quick sort.  
  
Quick sort algorithm is developed by C. A. R. Hoare. Quick sort is a comparison sort. The working of  quick sort algorithm is depending on a divide-and-conquer strategy. A divide and conquer strategy is  dividing  an array  into two sub-arrays. Quick sort is one of the fastest and simplest sorting algorithm. The complexity of quick sort in the average case is  Θ(n log(n)) and in the  worst case is Θ(n2).

**Code description:**

In quick sort algorithm pick an element from array of elements. This element is called the pivot. Then compare the the values from left to right until a greater element is find then swap the values. Again start comparison from right with pivot. When lesser element is find then swap the values. Follow the same steps until  all elements which are less than the pivot come before the pivot and all elements greater than the pivot come after it. After this partitioning, the pivot is in its last position. This is called the partition operation. Recursively sort the sub-array of lesser elements and the sub-array of greater elements.   
  
**Working of quick sort algorithm:  
Input:**12 9 4 99 120 1 3 10 13  **  
  
  
Output:**1 3 4 10 12 13 99 120

**The code of the program :**

|  |
| --- |
| **public class**QuickSort{   **public static void**main(String a[]){   **int**i;   **int**array[] = {12,9,4,99,120,1,3,10,13};    System.out.println("\n\n RoseIndia\n\n");   System.out.println(" Quick Sort\n\n");   System.out.println("Values Before the sort:\n");   **for**(i = 0; i < array.length; i++)   System.out.print( array[i]+"  ");   System.out.println();   quick\_srt(array,0,array.length-1);   System.out.print("Values after the sort:\n");   **for**(i = 0; i <array.length; i++)   System.out.print(array[i]+"  ");   System.out.println();   System.out.println("PAUSE");   }    **public static void**quick\_srt(**int**array[],int low, **int**n){   **int**lo = low;   **int**hi = n;   **if**(lo >= n) {   **return**;   }   **int**mid = array[(lo + hi) / 2];   **while**(lo < hi) {   **while**(lo<hi && array[lo] < mid) {   lo++;   }   **while**(lo<hi && array[hi] > mid) {   hi--;   }   **if**(lo < hi) {   **int**T = array[lo];   array[lo] = array[hi];   array[hi] = T;   }   }   **if**(hi < lo) {   **int**T = hi;   hi = lo;   lo = T;   }   quick\_srt(array, low, lo);   quick\_srt(array, lo == low ? lo+1 : lo, n);   } } |

**Output of the example:**

|  |
| --- |
| **C:\array\sorting>javac QuickSort.java**  **C:\array\sorting>java QuickSort**  **RoseIndia**  **Quick Sort**  **Values Before the sort:**  **12 9 4 99 120 1 3 10 13**  **Values after the sort:**  **1 3 4 9 10 12 13 99 120**  **PAUSE**  **C:\array\sorting>\_** |