## 1.Step by Step procedure to Install java:

1. Go to Folder where you have downloaded jdk setup.
2. Now Double Click on the setup file. (jdk-7u2-window-x64 is jdk file in my case)
3. Following Setup window will gets opened.
4. Now Click on “Next” button
5. It will prompt you optional features to be installed. Let it be default. Now click on “next”.
6. Now Keep Path as default.
7. From the desktop, right click the **Computer** icon.
8. Choose **Properties** from the context menu.
9. Click the **Advanced system settings** link.
10. Click **Environment Variables**. In the section **System Variables**, find the **PATH** environment variable and select it. Click **Edit**. If the **PATH** environment variable does not exist, click **New**.
11. In the **Edit System Variable** (or **New System Variable**) window, specify the value of the **PATH** environment variable. Click **OK**. Close all remaining windows by clicking **OK**.

2.Steps to install eclipse

Downloading Eclipse You can download eclipse from <http://www.eclipse.org/downloads/>

Installing Eclipse

Launching Eclipse

On the windows platform, if you extracted the contents of the zip file to c:\, then you can start eclipse by using c:\eclipse\eclipse.exe

When eclipse starts up for the first time it prompts you for the location of the workspace folder. All your data will be stored in the workspace folder. You can accept the default or choose a new location.

If you select "Use this as the default and do not ask again", this dialog box will not come up again. You can change this preference using the Workspaces Preference Page.

1. **Steps to create project**

**using the New Java Project wizard**

1. Enter the Project Name.
2. Select the Java Runtime Environment (JRE) or leave it at the default.
3. Select the Project Layout which determines whether there would be a separate folder for the source codes and class files.
4. create .java file/class?

**Using the New Java Class Wizard**

1. Ensure the source folder and package are correct.
2. Enter the class name.
3. Select the appropriate class modifier.
4. Enter the super class name or click on the Browse button to search for an existing class.
5. Click on the Add button to select the interfaces implemented by this class.
6. Examine and modify the check boxes related to method stubs and comments.
7. Click the Finish button.
8. how to create packages and what is best way to give name

A **Package** can be defined as a grouping of related types (classes, interfaces, enumerations and annotations ) providing access protection and namespace management.

Some of the existing packages in Java are −

* **java.lang** − bundles the fundamental classes
* **java.io** − classes for input , output functions are bundled in this package

## Creating a Package

While creating a package, you should choose a name for the package and include a **package** statement along with that name at the top of every source file that contains the classes, interfaces, enumerations, and annotation types that you want to include in the package.

The package statement should be the first line in the source file. There can be only one package statement in each source file, and it applies to all types in the file.

If a package statement is not used then the class, interfaces, enumerations, and annotation types will be placed in the current default package.

To compile the Java programs with package statements, you have to use -d option as shown below.

javac -d Destination\_folder file\_name.java

Then a folder with the given package name is created in the specified destination, and the compiled class files will be placed in that folder.

6. Why Main method is declared?

Why Main method is declared public static void in Java is one of Classical interview questions in Java. Every Java Programmer knows that main method is entry point in Java program but only few knows the secret behind signature and syntax of main method in Java

7. Properties is a subclass of Hashtable. It is used to maintain lists of values in which the key is a String and the value is also a String.

The Properties class is used by many other Java classes. For example, it is the type of object returned by System.getProperties( ) when obtaining environmental values.

8. What is a data type in Java?

The platform independent feature of **Java** is achieved through bytecode. The eight primitive **data types** are: byte, short, int, long, float, double, boolean, and char. The **java**.lang.String class represents character strings

There are two data types available in Java −

* Primitive Data Types
* Reference/Object Data Types

9.What is variable?

Each **variable** in Java has a specific type, which determines the size and layout of the **variable's** memory; the range of values that can be stored within that memory; and the set of operations that can be applied to the **variable**. You must declare all **variables** before they can be used.

10.what is method?

A Java method is a collection of statements that are grouped together to perform an operation. When you call the System.out.**println()** method, for example, the system actually executes several statements in order to display a message on the console.

**Syntax**

public static int methodName(int a, int b) {

// body

}

EX;

public class ExampleVoid {

public static void main(String [] args) {

methodRankPoints(255.7);

}

public static void methodRankPoints(double points) {

if (points >= 202.5) {

System.out.println("Rank1");

}else if (points >= 122.4) {

System.out.println("Rank2");

}else {

System.out.println("Rank3");

}

}

}

1. creating variable, we can create variables inside method

Declaring a variable in the main method will make it available only in the main. Declaring a variable outside will make it available to all the methods of the class, including the main method.

public class Foo {

private String varOne = "Test";

public void testOne() {

System.out.println(varOne);

System.out.println(varTwo);

}

public void testTwo() {

String varTwo = "Bar";

System.out.println(varOne);

System.out.println(varTwo); }

}

1. creating method with return data type, int/string/double/float/date etc

The data type of the return value must match the method's declared return type; you can't return an integer value from a method declared to return a boolean.

The getArea() method in the Rectangle [Rectangle](https://docs.oracle.com/javase/tutorial/java/javaOO/examples/Rectangle.java" \t "_blank) class that was discussed in the sections on objects returns an integer:

// a method for computing the area of the rectangle

public int getArea() {

return width \* height;

}

This method returns the integer that the expression width\*height evaluates to.

The getArea method returns a primitive type. A method can also return a reference type. For example, in a program to manipulate Bicycle objects, we might have a method like this:

public int getArea() {

return width \* height;

}

public Bicycle seeWhosFastest(Bicycle myBike, Bicycle yourBike,

Environment env) {

Bicycle fastest;

return fastest;

}

|  |
| --- |
|  |

14.hard coding?

First off, let’s clarify that hard coding is a term that applies to all programming languages and not just Java. So, what is hard coding? Well, it refers to the practice of using actual data values in the source code when instead what should be used are variables that accept any input value. There are definitely some situations when hard coding values may be necessary, but generally it is looked down upon as a negative practice that should be avoided whenever possible.

1. Create default/paramterzied constructors (overloaded constructor)

[**constructors**](http://beginnersbook.com/2013/03/constructors-in-java/) can also be overloaded. We will see constructor [**overloading**](http://beginnersbook.com/2013/05/method-overloading/) with the help of an example using this() and parameterized constructor.

package beginnersbook.com;

public class StudentData

{

private int stuID;

private String stuName;

private int stuAge;

StudentData()

{

//Default constructor

stuID = 100;

stuName = "New Student";

stuAge = 18;

}

StudentData(int num1, String str, int num2)

{

//Parameterized constructor

stuID = num1;

stuName = str;

stuAge = num2;

}

//Getter and setter methods

public int getStuID() {

return stuID;

}

public void setStuID(int stuID) {

this.stuID = stuID;

}

public String getStuName() {

return stuName;

}

public void setStuName(String stuName) {

this.stuName = stuName;

}

public int getStuAge() {

return stuAge;

}

public void setStuAge(int stuAge) {

this.stuAge = stuAge;

}

}

class TestOverloading

{

public static void main(String args[])

{

//This object creation would call the default constructor

StudentData myobj = new StudentData();

System.out.println("Student Name is: "+myobj.getStuName());

System.out.println("Student Age is: "+myobj.getStuAge());

System.out.println("Student ID is: "+myobj.getStuID());

/\*This object creation would call the parameterized

\* constructor StudentData(int, String, int)\*/

StudentData myobj2 = new StudentData(555, "Chaitanya", 25);

System.out.println("Student Name is: "+myobj2.getStuName());

System.out.println("Student Age is: "+myobj2.getStuAge());

System.out.println("Student ID is: "+myobj2.getStuID());

}

}

1. The return value for a property can be set in the same was as for a method, using Returns(). You can also just use plain old property setters for read/write properties; they’ll behave just the way you expect them to.

calculator.Mode.Returns("DEC");

Assert.AreEqual(calculator.Mode, "DEC");

calculator.Mode = "HEX";

Assert.AreEqual(calculator.Mode, "HEX");

1. In **Java**, **parameters** sent to methods are passed-by-value: Definition clarification: What is passed "to" a method is referred to as an "argument". The "type" of data that a method can receive is referred to as a "**parameter**".

public Polygon polygonFrom(Point[] corners) {

// method body goes here

}

16. **Java static method**

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

class StaticDemo

{

public static void copyArg(String str1, String str2)

{

//copies argument 2 to arg1

str2 = str1;

System.out.println("First String arg is: "+str1);

System.out.println("Second String arg is: "+str2);

}

public static void main(String agrs[])

{

//StaticDemo.copyArg("XYZ", "ABC");

copyArg("XYZ", "ABC");

}

}

\*signup

|  |  |
| --- | --- |
|  | import java.util.Scanner; |
|  |  |
|  | public class Signup { |
|  |  |
|  | public static void main(String[] args) { |
|  | String Firstname; |
|  | String lastname; |
|  | int phno; |
|  | int dob; |
|  |  |
|  | Scanner b = new Scanner(System.in); |
|  | System.out.println("Firstname"); |
|  | Firstname=b.next(); |
|  | System.out.println("lirstname"); |
|  | lastname=b.next(); |
|  | System.out.println("phno"); |
|  | phno=b.nextInt(); |
|  | System.out.println("dob"); |
|  | dob=b.nextInt(); |
|  | System.out.println("signup is successfully"); |
|  | Login.main(null); |
|  |  |
|  |  |
|  |  |
|  | } |
|  | } |

\*[ReverseString.java](https://github.com/bantu2205/Java/blob/master/ReverseString.java)

|  |  |
| --- | --- |
|  |  |
|  | import java.util.Scanner; |
|  |  |
|  | public class Addition { |
|  |  |
|  | private static Scanner s; |
|  | public static void main(String[] args) { |
|  | s = new Scanner(System.in); |
|  | System.out.print("Enter a value a:"); |
|  | int a = s.nextInt(); |
|  | System.out.print("Enter a value b:"); |
|  | int b = s.nextInt(); |
|  | int sum =a+b; |
|  | int sub =a\*b; |
|  | float div =a/b; |
|  | System.out.println("Total of sum:"+sum); |
|  | System.out.println("Total of sub:"+sub); |
|  | System.out.println("Total of div:"+div); |
|  |  |
|  | } |
|  |  |
|  | } |

\*[PrimeNumber.java](https://github.com/bantu2205/Java/blob/master/PrimeNumber.java)

|  |
| --- |
| import java.util.Scanner; |
|  |  |
|  | public class PrimeNumber { |
|  | public static Scanner f; |
|  | public static void main(String[] args) { |
|  | f = new Scanner(System.in); |
|  | System.out.println("Enter number:"); |
|  | int i = f.nextInt(); |
|  | //Logic for Prime list of Numbers |
|  | for(i=2;i<=100;i++){ |
|  | for(int j=2;j<=i;j++){ |
|  | if(j==i) |
|  | { |
|  | System.out.println(i); |
|  | } |
|  | if(i%j==0) |
|  | { |
|  | break; |
|  | } |
|  | } |
|  | } |
|  | } |
|  |  |
|  | } |

\*[Practice.java](https://github.com/bantu2205/Java/blob/master/Practice.java)

|  |
| --- |
| import java.util.Scanner; |
|  |  |
|  | public class Practice { |
|  |  |
|  | private static Scanner f; |
|  |  |
|  | public static void main(String[] args) { |
|  | int i; |
|  | System.out.println("Enter number:"); |
|  |  |
|  | f = new Scanner(System.in); |
|  | i = f.nextInt(); |
|  | if(i % 2==0) |
|  | System.out.println(i+ "Even number "); |
|  | else |
|  | System.out.println(i+ " Odd number "); |
|  |  |
|  |  |
|  |  |
|  | } |
|  |  |
|  | } |

\***Powers.java**

|  |
| --- |
| import java.util.Scanner; |
|  |  |
|  | public class Powers { |
|  |  |
|  | private static Scanner f; |
|  |  |
|  | public static void main(String[] args) { |
|  | int c; |
|  | f = new Scanner(System.in); |
|  | System.out.println("Enter number"); |
|  | int i = f.nextInt(); |
|  | //Given number is power of number |
|  | if(i<100){ |
|  |  |
|  | c=i\*i; |
|  |  |
|  | System.out.println("value is:" +c); |
|  |  |
|  | } |
|  |  |
|  |  |
|  |  |
|  | } |
|  |  |
|  | } |

\***Palindrom.java**

|  |
| --- |
| package Facebook; |
|  |  |
|  | import java.util.Scanner; |
|  |  |
|  | public class Palindrom { |
|  |  |
|  | private static Scanner h; |
|  |  |
|  | public static void main(String arg[]){ |
|  |  |
|  | int r,sum=0,temp; |
|  |  |
|  | //It is the number variable to be checked for palindrome |
|  | h = new Scanner (System.in); |
|  | System.out.println("Enter number"); |
|  | int n =h.nextInt(); |
|  | temp=n; |
|  | while(n>0){ |
|  | r=n%10; //getting remainder |
|  | sum=(sum\*10)+r; |
|  | n=n/10; |
|  | } |
|  | if(temp==sum) |
|  | System.out.println("palindrom number "); |
|  | else |
|  | System.out.println("Not palindrom number"); |
|  |  |
|  | } |
|  | } |

\***Matrix.java**

|  |
| --- |
| mport java.util.Scanner; |
|  |  |
|  | public class Matrix { |
|  | private static Scanner input; |
|  | public static void main(String[] args) { |
|  | int n; |
|  | input = new Scanner (System.in); |
|  | System.out.println("Enter Matrix Base:"); |
|  | n = input.nextInt(); |
|  | int a[][]= new int[n][n]; |
|  | int b[][]= new int[n][n]; |
|  | int c[][]= new int[n][n]; |
|  |  |
|  | System.out.println("Enter First Matrix Number"); |
|  | for(int i=0;i<n;i++){ |
|  | for (int j=0;j<n;j++){ |
|  |  |
|  | a[i][j] = input.nextInt(); |
|  | } |
|  |  |
|  | } |
|  | System.out.println("Enter Second Matrix Number"); |
|  | for(int i=0;i<n;i++){ |
|  | for (int j=0;j<n;j++){ |
|  |  |
|  | b[i][j] = input.nextInt(); |
|  | } |
|  |  |
|  | } |
|  | System.out.println("Enter Multiplication Matrix Number"); |
|  | for(int i=0;i<n;i++){ |
|  | for (int j=0;j<n;j++){ |
|  | for (int k=0;k<n;k++){ |
|  |  |
|  | c[i][j] = c[i][j]+a[i][k]\*b[k][j]; |
|  | } |
|  |  |
|  | } |
|  | } |
|  | System.out.println("The product is:"); |
|  |  |
|  | for (int i = 0; i < n; i++) |
|  |  |
|  | { |
|  |  |
|  | for (int j = 0; j < n; j++) |
|  |  |
|  | { |
|  |  |
|  | System.out.print(c[i][j] + " "); |
|  |  |
|  | } |
|  |  |
|  | System.out.println(); |
|  |  |
|  | } |
|  |  |
|  | input.close(); |
|  |  |
|  | } |
|  |  |
|  |  |
|  | }  \***Maths.java**   |  | | --- | | import java.util.Scanner; | |  |  | |  | public class Maths { | |  |  | |  |  | |  |  | |  | /\*Math Magic results 13837\*age\*73= 262626 | |  | \* | |  | age =26 \*/ | |  | public static void main(String[] args) { | |  | int a =13837; | |  | int b =73; | |  | int Results; | |  | Scanner d = new Scanner (System.in); | |  | System.out.println("Enter your age:"); | |  | int age = d.nextInt(); | |  | Results =a\*age\*b; | |  | System.out.println("Maths Magic:"+Results); | |  |  | |  |  | |  | } | |  |  | |  | } |   \*login   |  | | --- | | import java.util.Scanner; | |  |  | |  | public class Login { | |  |  | |  |  | |  |  | |  | private static Scanner s; | |  |  | |  | public static void main(String[] args) { | |  |  | |  | String username; | |  | String password; | |  |  | |  | s = new Scanner(System.in); | |  | System.out.println("Enter username "); | |  | username = s.next(); | |  | System.out.println("Enter username "+username); | |  | System.out.println("Enter password "); | |  | password = s.next(); | |  | System.out.println("Enter password "+password); | |  | System.out.println("login successfully"); | |  | //Signup.main(null); | |  |  | |  |  | |  |  | |  | } | |  |  | |  | } | |