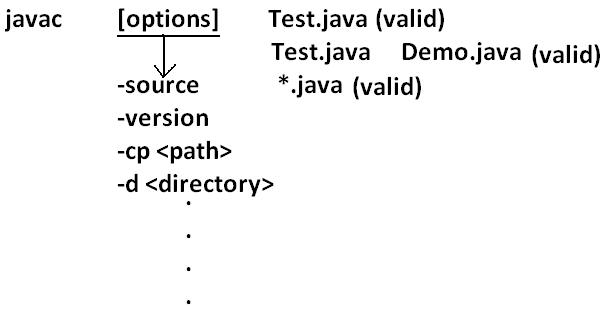
**Development**

**Javac:** we can use Javac to compile a single or group of “.java files”.

**Syntax**:



**Java:** we can use java command to run a single “.class file”.

**Syntax:**



**Classpath:** Class path describes the location where the required “.class files” are available. We can set the class path in the following 3 ways.

1. Permanently by using environment variable “classpath”. This class path will be preserved after system restart also.
2. Temporary for a particular command prompt level by using “set” command.

**Example:**



* Once if you close the command prompt automatically this class path will be lost.

1. We can set the class path for a particular command level by using “–cp” (or) “–class path”. This class path is applicable only for that command execution. After executing the command this classpath will be lost.

* Among the 3 ways of setting the class path the most common way is setting class path at command level by using “–cp”.

**Example 1:**

class Rain

{

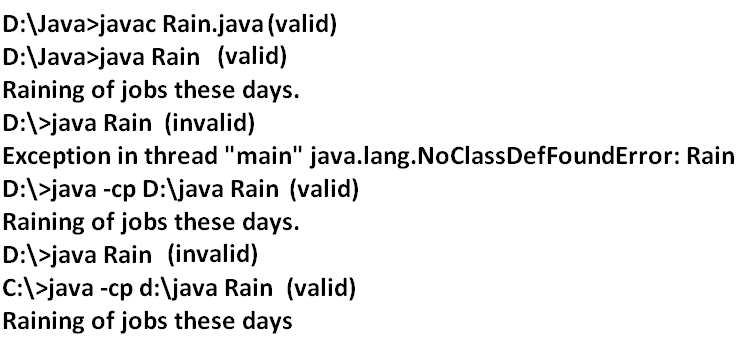
public static void main(String args[]){

System.out.println("Raining of jobs these days");

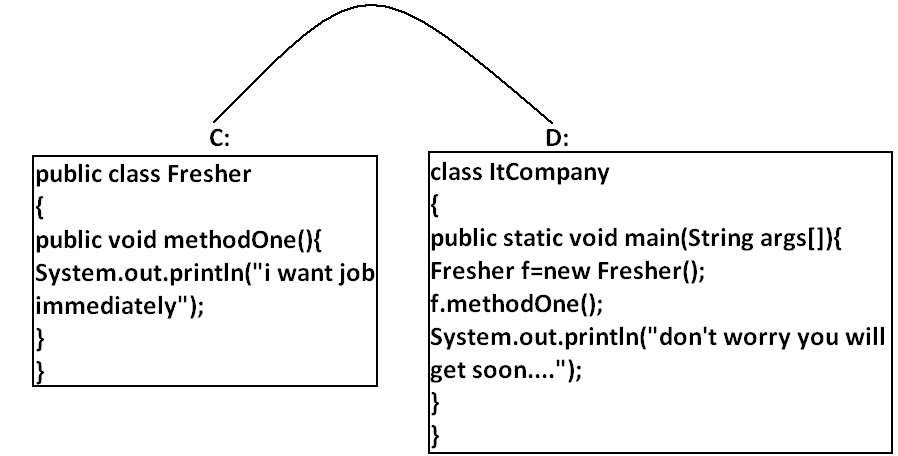
}

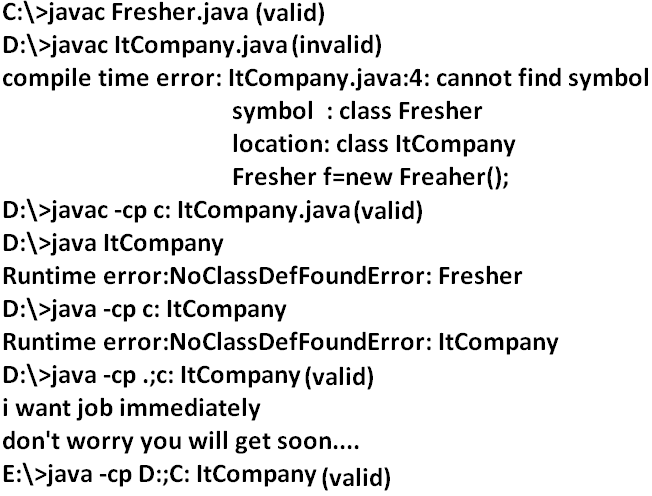
}

**Analysis:**

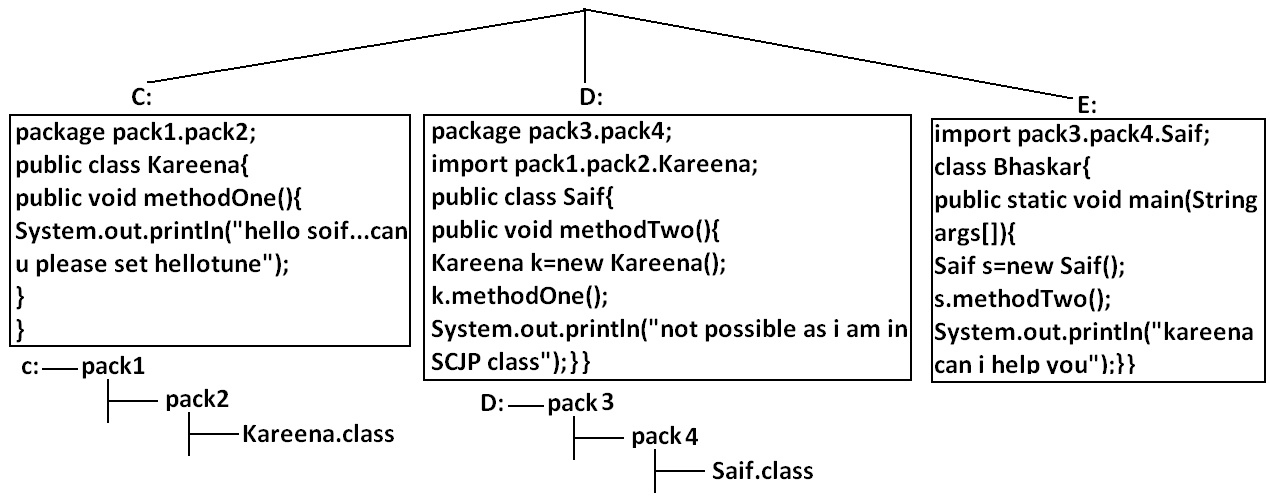


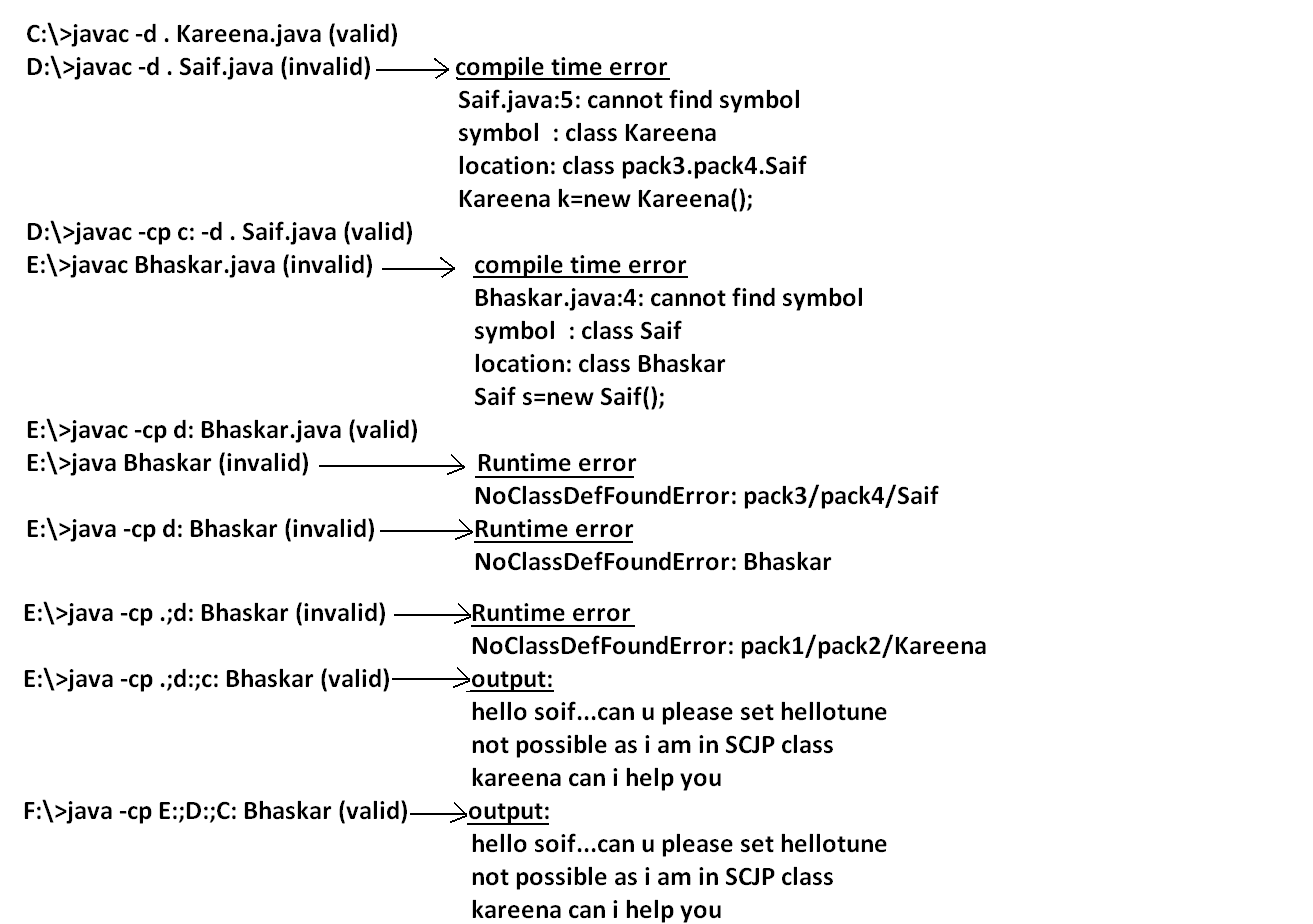
**Example 2:**

**Analysis:**



**Example 3:**

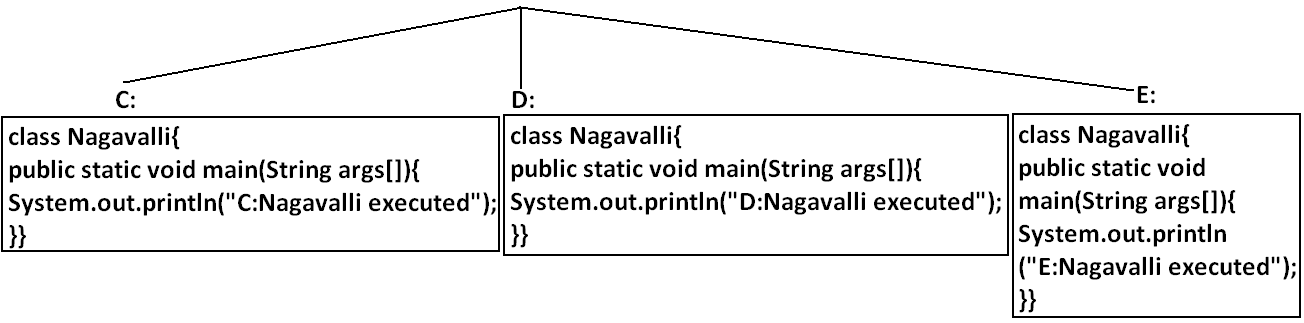
**Analysis:**

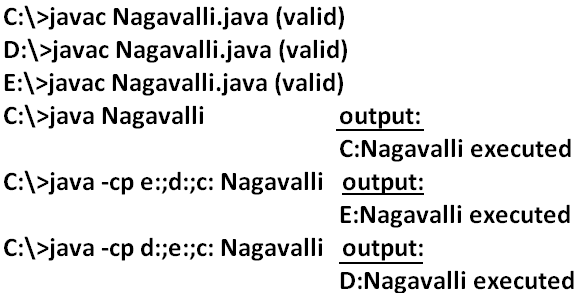


**Note:** if any folder structure created because of package statement. It should be resolved by import statement only and the location of base package should be make it available in class path.

**Note:** in classpath the order of locations is very important and it should be from left to right.

**Example 4:**

**Analysis:**

****

**Jar file:** If several dependent classes present then it is never recommended to set the classpath individual for every component. We have to group all these “.class files” into a single jar file and we have to make that jar file available to the classpath.

**Example:** All required classes to develop a Servlet are grouped into a single jar file (Servlet-api.jar) hence while compiling Servlet classes we have to make this jar file available in the classpath.

What is the difference between Jar, War and Ear?

**Jar (java archive):** Represents a group of “.class files”.

**War (web archive):** Represents a web application which may contains Servlets, JSP, HTML pages, JavaScript files etc.

**Ear (Enterprise archive):** it represents an enterprise application which may contain Servlets, JSP, EJB’S, JMS component etc.

* In generally an ear file consists of a group of war files and jar files.

Ear=war+ jar

**Various Commands:**

**To create a jar file:**

D:\Enum>jar -cvf bhaskar.jar Beer.class Test.class X.class

D:\Enum>jar -cvf bhaskar.jar \*.class

**To extract a jar file:**

D:\Enum>jar -xvf bhaskar.jar

**To display table of contents of a jar file:**

D:\Enum>jar -tvf bhaskar.jar

**Example 5:**

public class BhaskarColorFulCalc{

public static int add(int x,int y){

return x\*y;

}

public static int multiply(int x,int y){

return 2\*x\*y;

}}

**Analysis:**

C:\>javac BhaskarColorFulCalc.java

C:\>jar -cvf bhaskar.jar BhaskarColorFulCalc.class

**Example 6:**

class Client{

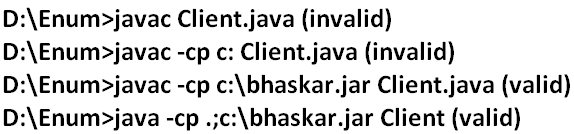
public static void main(String args[]){

System.out.println(BhaskarColorFulCalc.add(10,20));

System.out.println(BhaskarColorFulCalc.multiply(10,20));

}}

**Analysis:**



**Note:** Whenever we are placing jar file in the classpath compulsory we have to specify the name of the jar file also and just location is not enough.

**System properties:**

* For every system some persistence information is available in the form of system properties. These may include name of the os, java version, vendor of jvm etc.
* We can get system properties by using getProperties() method of system class. The following program displays all the system properties.

**Example 7:**

import java.util.\*;

class Test{

public static void main(String args[]){

//Properties is a class in util package.

//here getPropertes() method returns the Properties object.

Properties p=System.getProperties();

p.list(System.out);

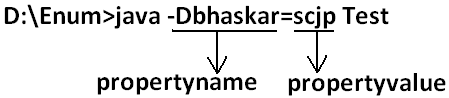
}

}

**How to set system property from the command prompt:**

* We can set system property from the command prompt by using –D option.

**Command:**



**What is the difference between path and classpath?**

**Path:** We can use “path variable” to specify the location where required binary executables are available.

* If we are not setting path then “java” and “Javac” commands won’t work.

**Classpath:** We can use “classpath variable” to describe location where required class files are available.

* If we are not setting classpath then our program won’t compile and run.

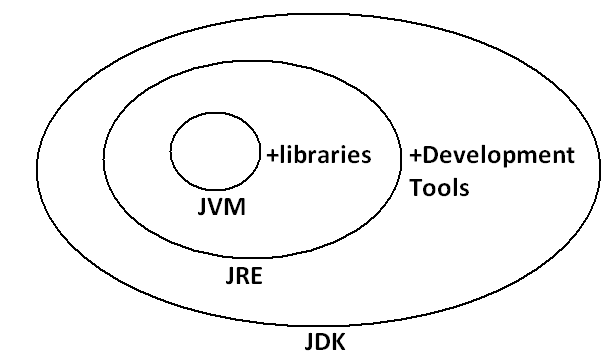
**What is the difference between JDK, JRE and JVM?**

**JDK (java development kit):** To develop and run java applications the required environment is JDK.

**JRE (java runtime environment):** To run java application the required environment is JRE.

**JVM (java virtual machine):** To execute java application the required virtual machine is JVM.

**Diagram:**



* JDK=JRE+Development Tools.
* JRE=JVM+Libraries.
* JRE is the part of JDK.
* Jvm is the part of JRE.

Note: At client side JRE is required and at developers side JDK is required.

**Shortcut way to place a jar files:**

* If we are placing jar file in the following location then it is not required to set classpath explicitly.

**Diagram:**

