1. **Software required to Develop the Java program.**
2. **Download and Install Java Software.**
3. **Environment Variables.**
4. **Java Program Structure.**
5. **Steps to develop & compile the java program.**
6. **Java Compiler Activities.**
7. **Sample Program.**
8. **Naming Conventions in java**

**1. Software required to Develop the Java program:** we require 3 types of software.

1.1.Editor:- The editor/Editor is used to write program and save it.

EX:- notepad, notepad++, eclipse, netBeans, ..etc.

1.2. JDK: we are going to use jdk 18 version.

1.3. Command Prompt:-Command prompt is for running compiler software to compile our program. Command prompt is for running JVM to execute the java program.

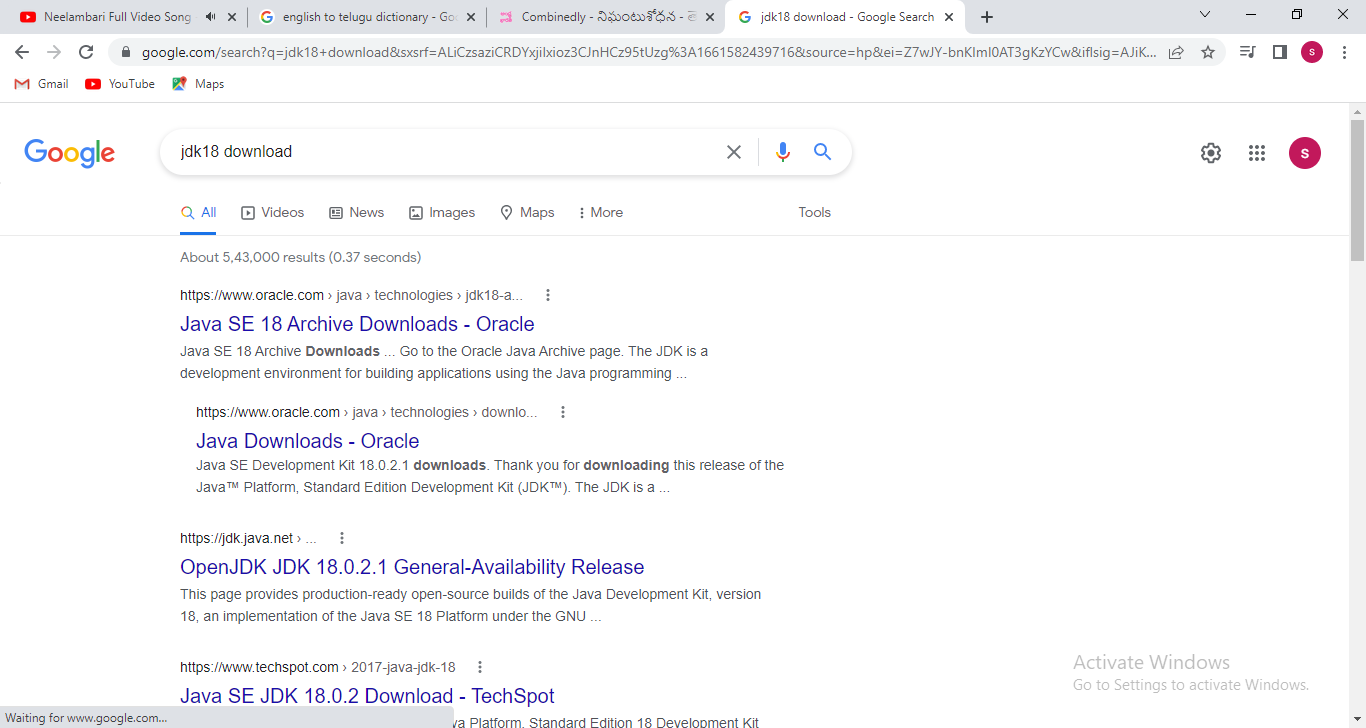
**2. Download And Install Java software:**

The java software name is JDK.

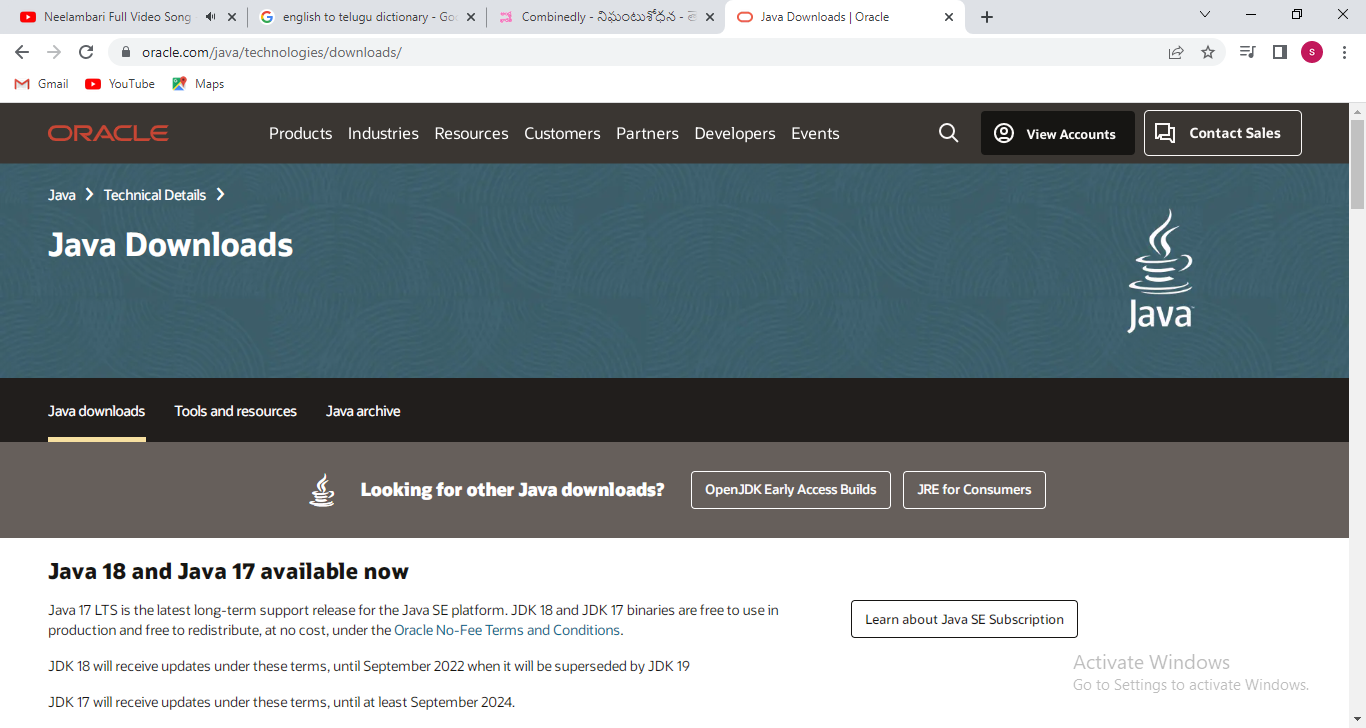
**2.1. Downloading:-**

Step1: open the google chrome.

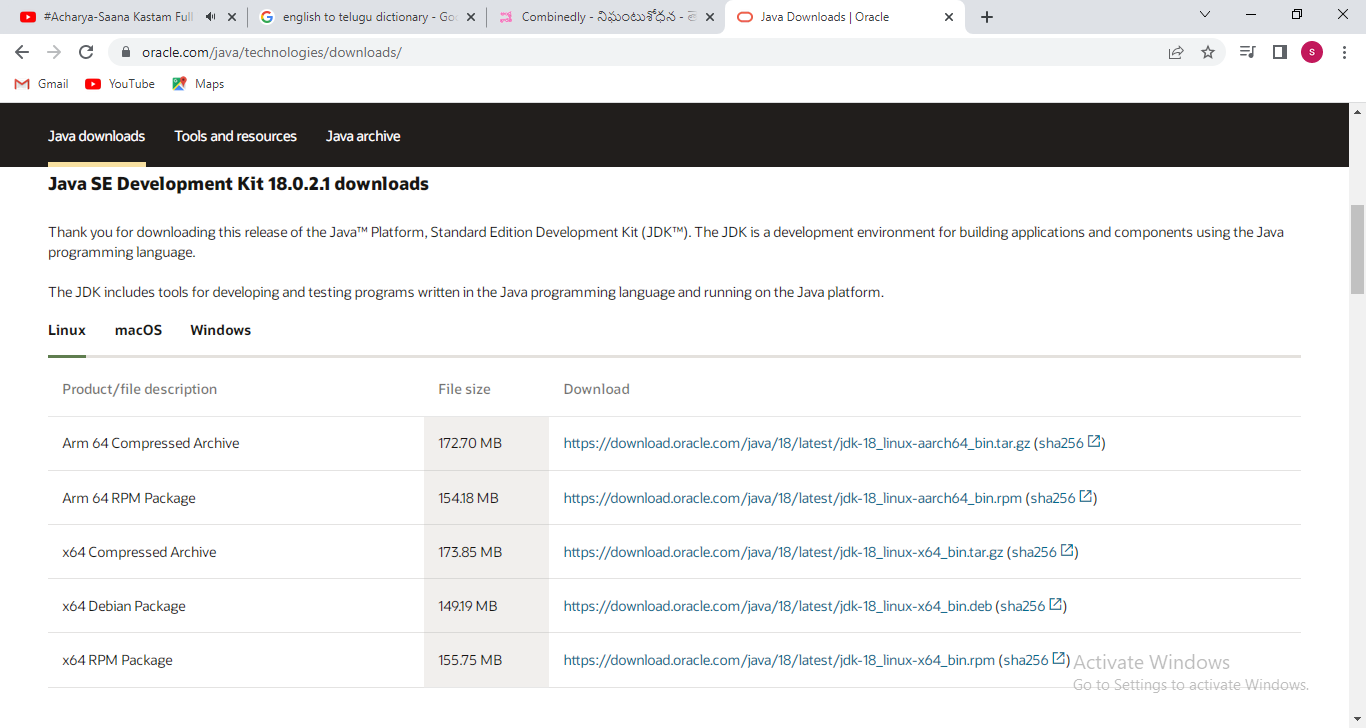
Step 2: Type JDK18 download.



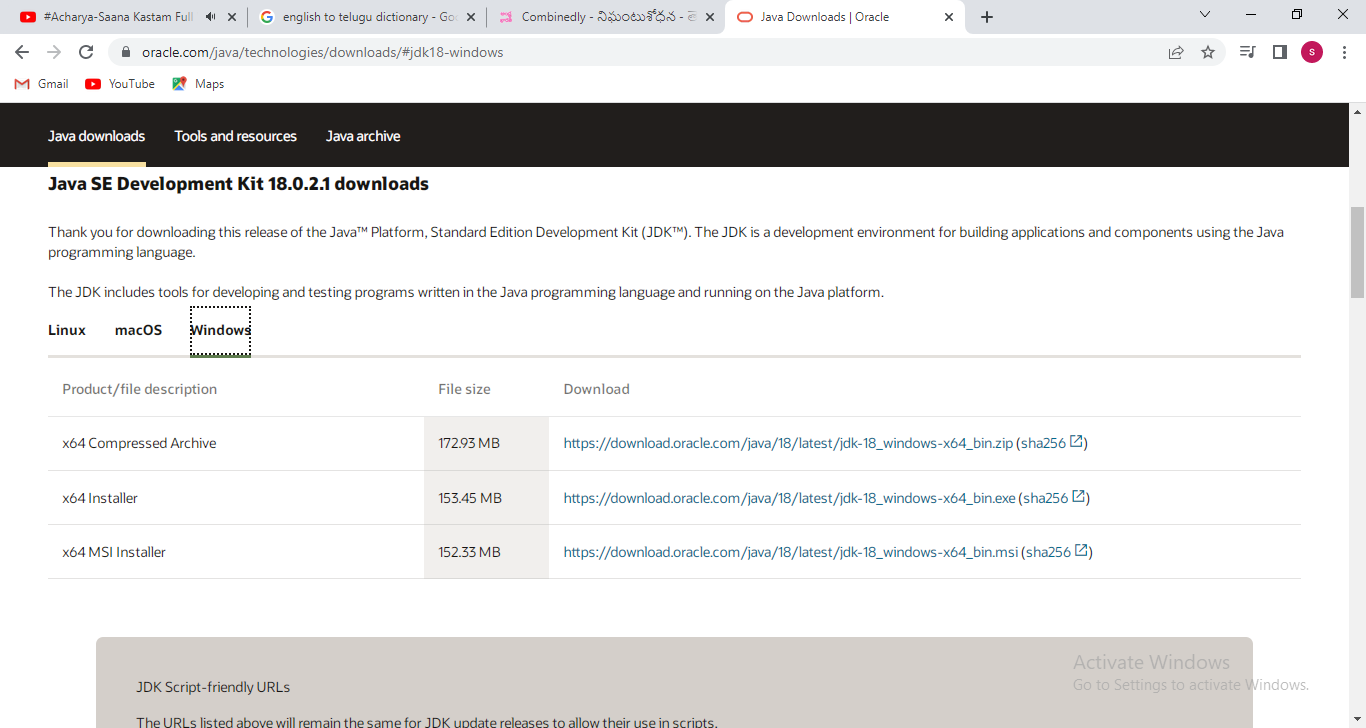
Step 3: click the java downloads-oracle.



Step 4: click the ‘java download’ tab.



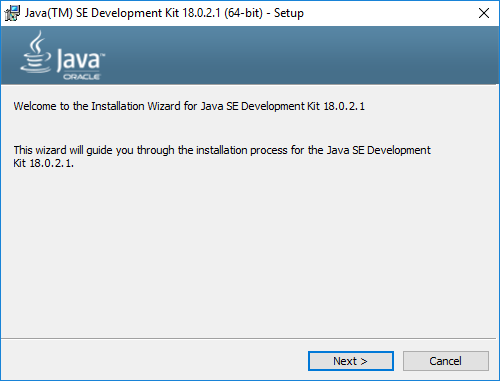
Step 5: click on ‘ windows’ tab.



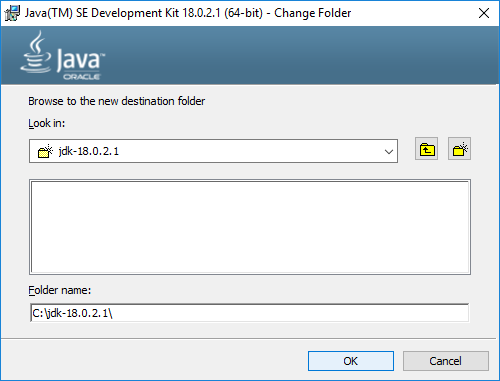
Step 6: click on ‘windows-x64\_bin.exe’ file. After clicking the that, JDK will be downloaded into local machine.

**2.2.Installing:**

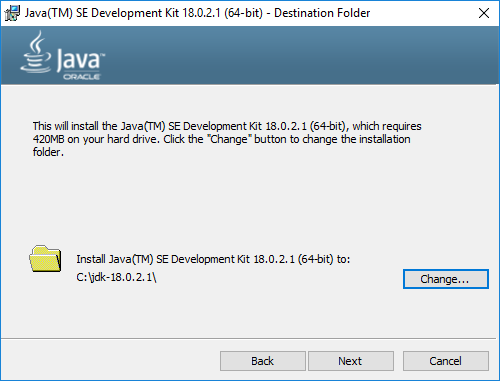
Step1. Click on .exe file.



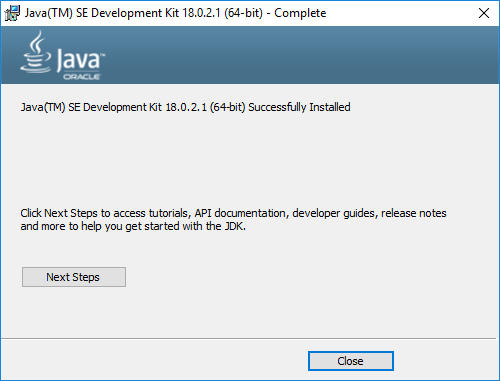
Step2. Click on ‘Next’ button. Change the installation location



Step3. Click ‘ok’ button.



Step4: click the ‘next’ button.



Step 5: click on ‘close’ button.

Now installation completed.

Note:- 1.Java14 on wards, we don’t need to configure(Ex: setting class path,…etc).

2.The directory in which JDK installed is called a JAVA\_HOME.

EX:C:\jdk-18.0.2.1 is called JAVA\_HOME.

**3.Environment Variables:**

1. Environment variables are not case-sensitive.
2. Values set to environment variable from command prompt are specific to that command prompt. i.e The values disappear once command prompt closed.
3. Values set to environment variable from Mycomputer will remain permanent for ever irrespective computer restars & command prompt opening and closing.
4. Environment variables are two types.
5. User variables: -These are specific to logged on users.
6. System Variables:- These are common for all windows users of computers.
7. Multiple values added to environment variables must be separated by “;”.
8. “.” 🡪 represents the current working directory.
9. “..” 🡪 represent the parent directory.

Note:- In java Environment, Two environment variables are used. They are

a.PATH.

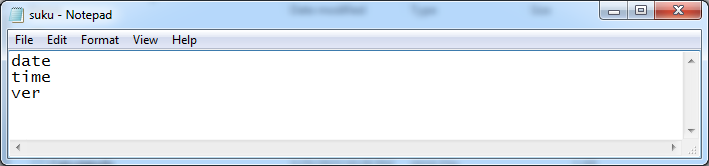
b.CLASSPATH

**a.PATH:-** If we want to make the execution of binary file(.exe file, .batch file, etc …) from any location of computer then add its original address of file to PATH file.

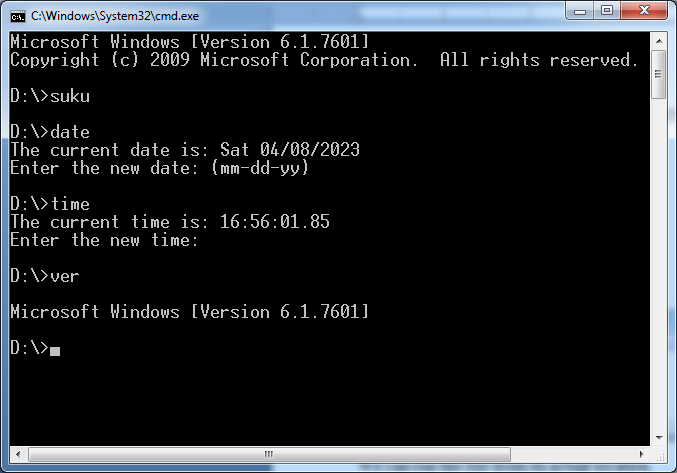
**Example:**

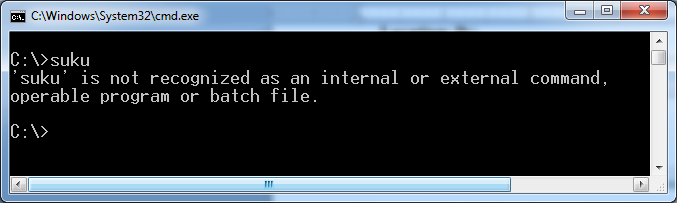
**FileName:Suku.bat**

**Location: D:\**

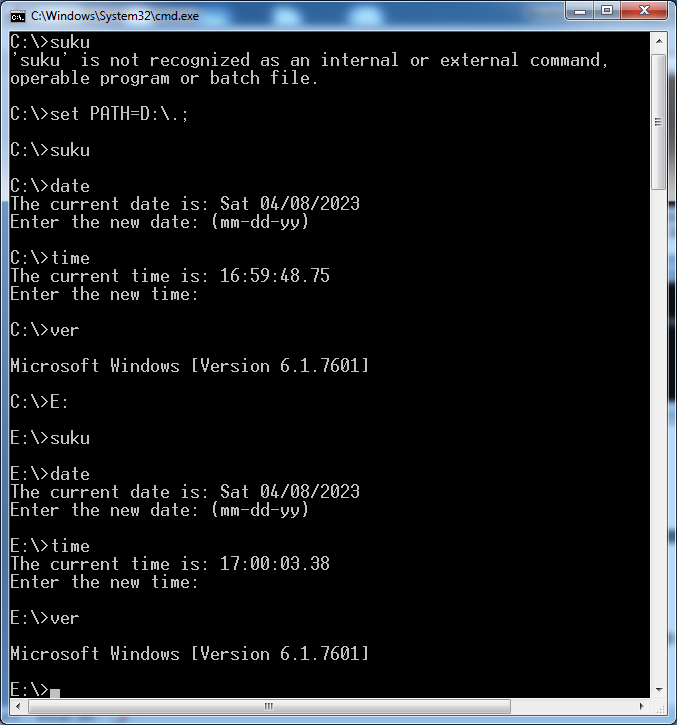
****

We can make .bat file execution from its actual location.

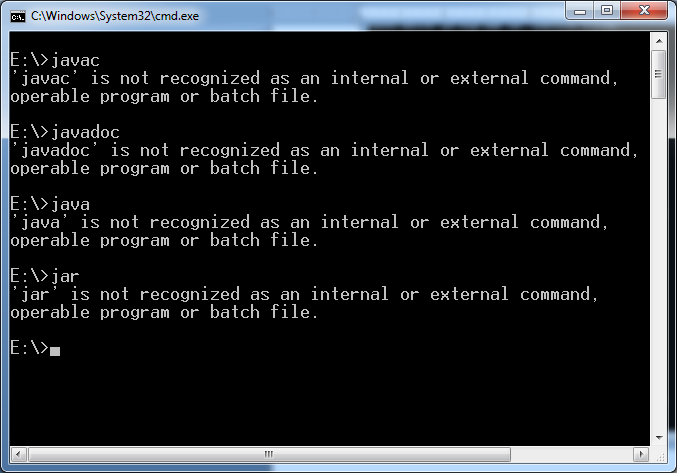
**Problem:-**when we make execution this file from another location, we get the following error.



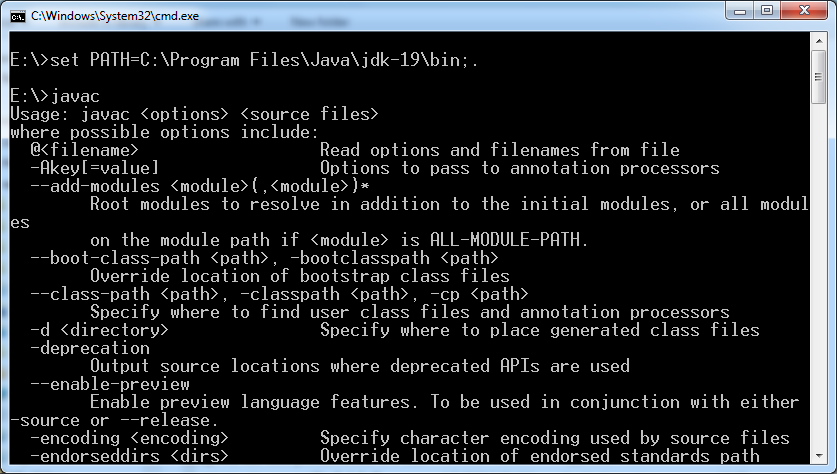
**Solution:- Add location of batch file to PATH variable.**

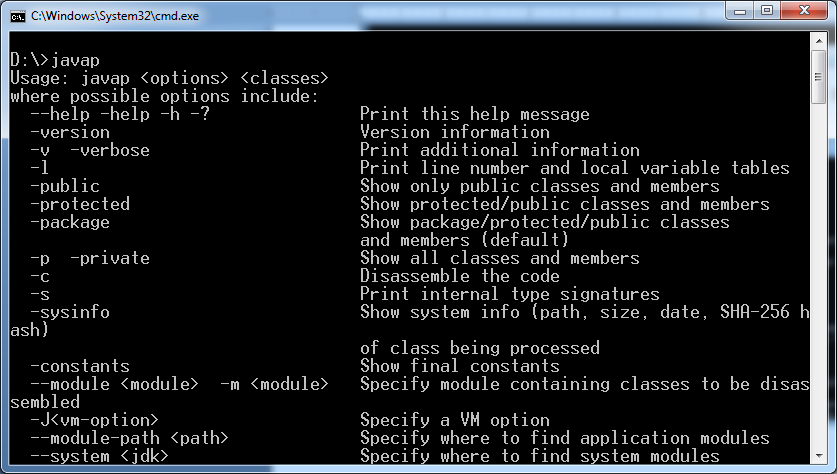
****

**All java commands like java,javac,javap,jar,javadoc ,etc … are .exe file. All present <java\_home>\bin directory. We use them from its original location.**

****

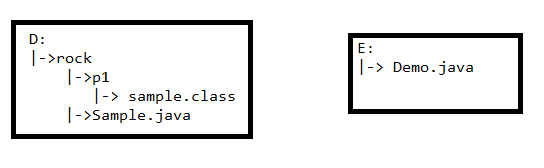
**To access them from any location of computer, add absolute path of bin directory to PATH variable.**

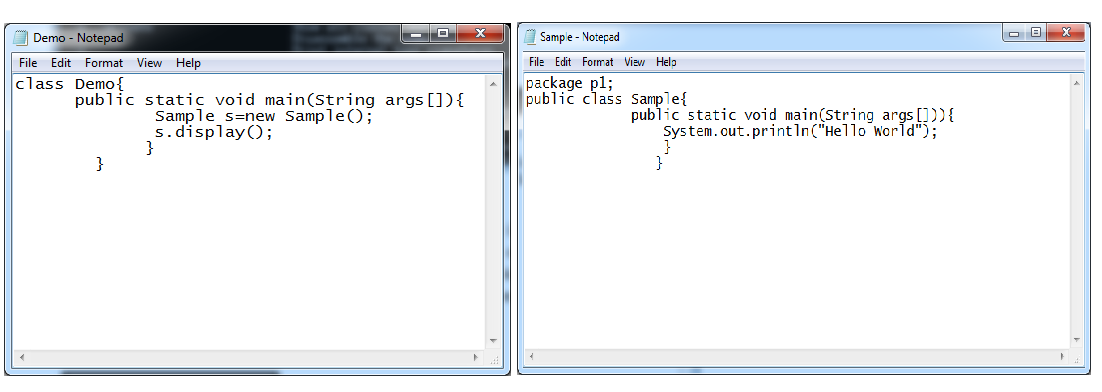
****

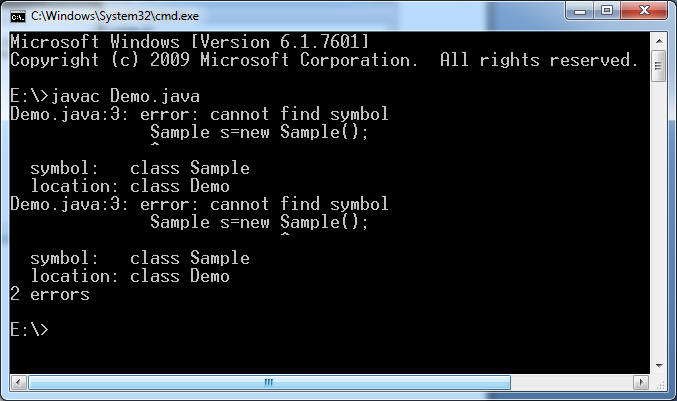
****

**b.CLASSPATH:-**  It is purely Java environment variable (i.e) It is only used in java environment . It is not used in non-java environment. If the our java app is using third party API’s or User defined APIS (other than JDK APIS) then we need to add those apis related jar files to CLASS PATH environment variable.

**Example:**

****

****

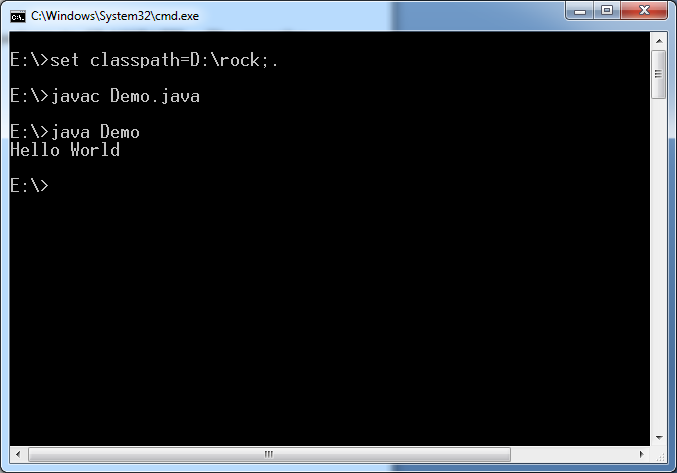
****

**Q) Why did we get this error?**

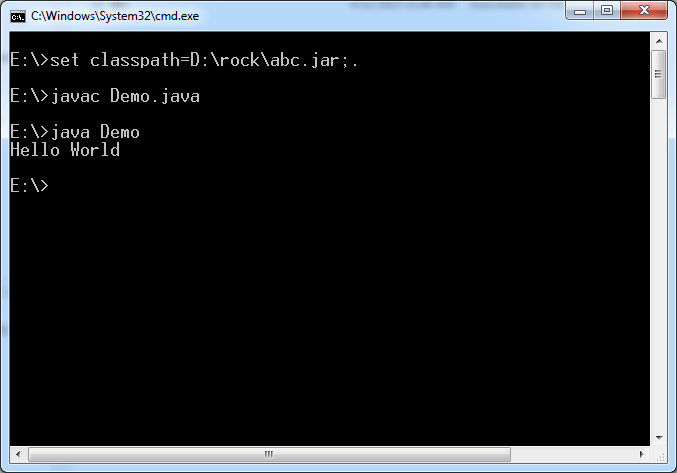
**The Sample class is user defined API. We did not specify compiler where it is available so Compiler could not recognize Sample class.**

**Solution: 1.By adding location of package to CLASSPATH , The compiler aware location of Sample class.**

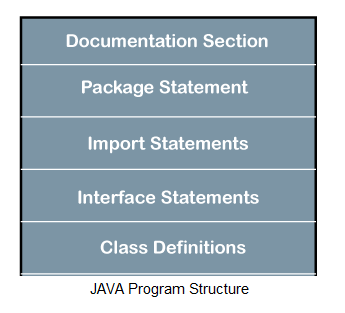
**Solution: 2. By adding location of jar file which has package to CLASSPATH, the compiler aware location of Sample class.**

****

**Solution: 2. By adding location of jar file which has package to CLASSPATH, the compiler aware location of Sample class.**

****

**4. Java Program Structure:**

****

**4.1.Documentation Section**: Documentation Section:- This section contains set of comment lines giving the name of program, name of the author and other details.

**4.1.1)comment :-** The comment is description of program elements (module, package, interface, class, variable, blocks, constructor, method, inner class..etc).

Java supports 3 types of comments.

1. Single line comment(Inline comments)
2. Multiline comment(Block comments)
3. Document comment.

**4.1.1.1) Single line comment:** // symbols are used to create single line comment. The description is only in single line.

Syntax: // statement

Example: // Salary variable stores the employ salary

Float salary;

**4.1.1.2) Multiline comment :** /\* \*/ symbols are used to create multiline comment.

Syntax: /\* statements \*/

Example:

/\* Author-Name: Sukumar

Date:04-05-2020

Program-Name: Addition.

Company-name: Raos’Degree college.

\*/

**4.1.1.3) Document Comment:-** The java provides document comment symbols which is /\*\* and \*/ . The document comment starts with /\*\* and ends with \*/. The documentation comment may precede the class header, interface header, method header, field declaration and other component. To make source code Documentation without distributing the source code , we need some tool. That tool is javadoc. The tool does opposite of what compiler does. It collects, formats and organize the java documents in socurce code.

**4.2. Package Section/Package Statement:**-

Syntax:

Package packagename;

This is an optional statement. This section contains only one package statement. If u put more than one package statement in this section, compiler displays error message.

**4.3.Import Section/Import Statement:** This section is optional section. It contains one or more than one import statements. if we want to use a class[es] of another package, then you can do this by importing it directly into our program.

Syntax:1

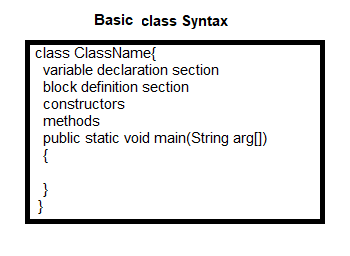
Import packagename.classname;

Syntax:2

Import packagename.\*;

**4.4.Interface Section/interface Statements:**  This section is optional section. It contains one or more than one interface definitions.

**4.5 Class Section**:This section is also optional section. It can also contain more than one class definitions where as class names must be unique.



**4.6 Main() method:** The main() is the starting point for JVM to start execution of a Java program.

**public:** It is an access specifier. We should use a public keyword before the main() method so that JVM can identify the execution point of the program. If we use private, protected, and default before the main() method, it will not be visible to JVM.

**static:** You can make a method as static by using the keyword static.JVM should call the main() method without creating an object. Static methods are the method which invokes without creating the objects, so JVM do not need any object to call the main() method.

**void:** In Java, every method has the return type. Void keyword acknowledges the compiler that main() method does not return any value.

**Main**: It takes only one argument. That argument data type is string array.

**5. Steps to Develop, compile & Execute Java Program:**

1. Open note pad[press start-> type notepad-> press enter]
2. Type the Java program.
3. Save the program with .java extension in your Desired Directory.
4. Open cmd prompt [press start->type cmd ->press enter]
5. Change and prompt path to directory which contain java program.
6. Compile java program with javac tool.

>javac filename.java

7. Execute java program with java tool as below.

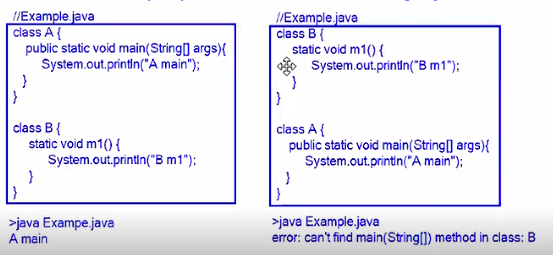
> java .class file name.

Note:- Java 11 onwards, we don’t need to compile .java file separetly. After saving the .java file, we can directly run the .java file using “java” tool with out compilation.

>java filename.java

There is small condition. If .java file have more than one class then the first class should has main method else we will get CE: can’t find the main method in first class name.

Example:



**6. Java Compiler Activities:**

1. The compiler searches given java file in PWD(present working directory).
2. If .java is not found,

It throws error: File Not found: filename.java

1. If .java file is found then It starts compiling the source code.
2. It read all source code from beginning to end and It verifies syntaxes.
3. If there are any syntax mistakes then compiler throughs errors and terminates compilation & does not generate .class files.

6. If there are no syntax mistakes, compiler creates .class file separately for every class with that class name.

7.Compiler converts class source code into bytecode(also called Virtual Machine Code) and store bytecode in appropriate .class file.

**7.Sample program:**

Filename: sample.java

Program:

class Sample{

public static void main (String args[]){

System.out.println(“Hello World “);

}

}

Compilation:

>javac sample.java

|-> sample.class

Execution:

>java sample

Output:

Hello World

**Note:**

**1. If java program has n no.of classes, then java compiler generates n no.of .class files**.

Does java compiler create .class files for following programs?

1.class A{} A)1

2.class A{} class B{} A)2

3.class A{} class B{} class C{} A)3

**2. If java file has public class, then java file name should be same as Public class name. Otherwise It leads to compile error.**

**If java files has only non-public classes, then java file name can be user defined.**

**Single java file can has only one public class.**

1. class A

{

}

Filename: A.java

1. Valid filename.
2. Class A{}

Class B{}

Class C{}

Filename: B.java

1. Valid filename.
2. public class A{}

Filename:xyz.java

Filename:Invalid.

1. public class A{}

public class B{}

class C{}

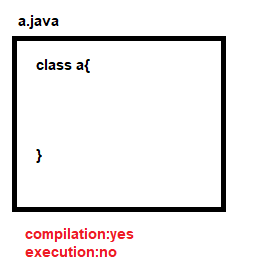
1. It leads to CE.
2. public class A{}

class B{}

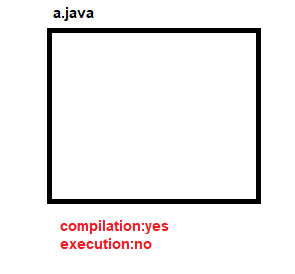
class c{}

filename: A.java

**3. If class does not have main method, then it can be compiled but can’t be executed.**

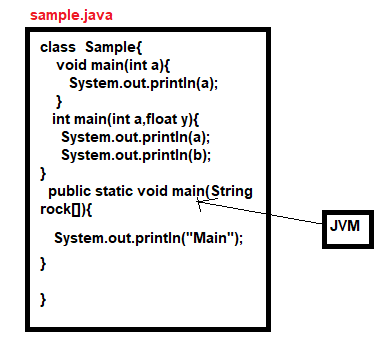
****

**4. The empty java file can be compiled . After compilation, .class file will not have main() method so it can not be executed.**

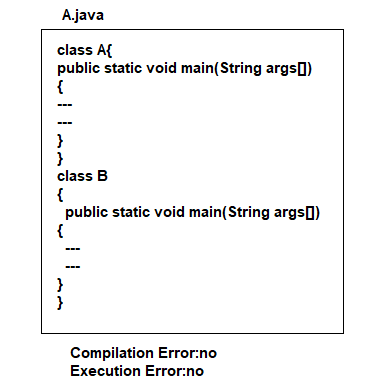
****

1. **even class has more than one main() method, At the time of execution , JVM invokes main which has following prototype.**

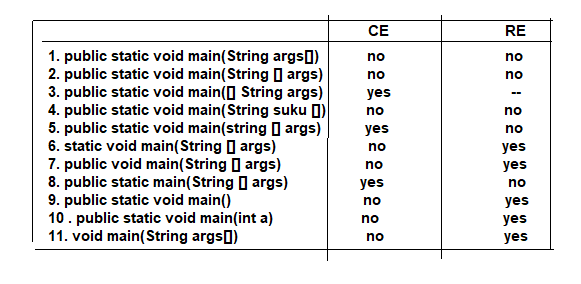
**Public static void main(String par\_name[])**

****

1. **Even one or more classes have main(), we don’t get CE and RE.**

****

**8.**

****

**Naming Conventions in Java**

The naming convention is not rule. It is just agreement/guidelines among java developers for easy readability of code.

**1.Class:-**

a. The className should be common noun.

b. whe you create class name with several words. You should capitalize each word.the uppercase letter acts as a separator.

Example:

Employee,TwoStairBuilding,Customer,ElectricCooker.

**2. Interface:-**

a. The Interface should be Adjective.

b. When you create Interface name with several words, You should capitalize each word.the uppercase letter acts as separator.

Example:- Runnable, Printable,Accessible …etc.

**3.Method:-**

a. The Method name should be verb.

b. If method name contains multiple words, the first word is in small letter then from second word onwards each new word starts with capital letter.

c. If method name contains single word, that word is small letter.

Example: print , display, readLine

**4. Variable:-**

a. If variable name contains multiple words, the first word is in small letter then from second word onwards each new word starts with capital letter.

b. If variable name contains single word, that word is small letter.

c. Avoid using one-character variables such as x,y,z ..etc.

Example:

empName,StudentRollNum,CarNo

**5. Package:-**

a. name of package in java is written in lowercase.

b. If name contains multiple words, It should be separated by(.)

c. The package name should have at least 3 words.

Example:-

Java.util

Java.io …etc

**6. Constants:-**

a. It should be in uppercase letters.

b. If name contains multiple words, It shouldbe separated by underscore( \_ ).

Example:

RED,YELLOW, MIN\_AGE,MAX\_AGE.

**Camel case:-** If name is combined with multiple words, except firstword, remaining all words starts with uppercase letter. This is camelcase.

Java follows camel-case syntax for naming classs, Interface,method and variable.