Object Oriented Programming with Java
Practical-1
Understanding Java Environment and IDE

1. Write steps to install JAVA Development Kit (JDK).

Step 0: Un-Install Older Version(s) of JDK/JRE

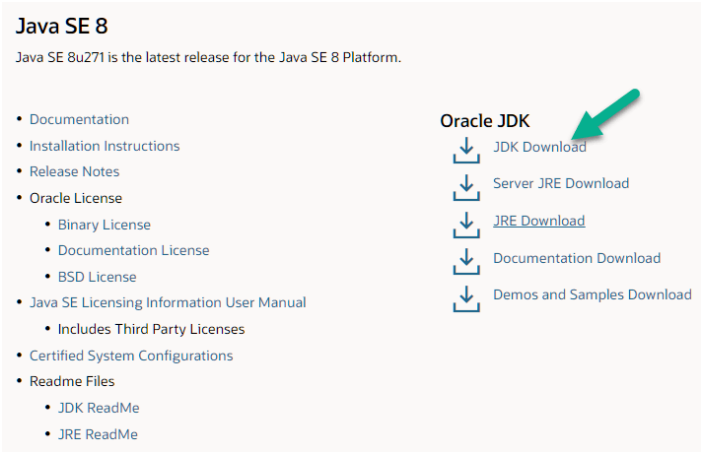
I recommend that you install only the *latest* JDK. Although you can install multiple versions of JDK/JRE concurrently, it is messy.

WE USE JDK 1.8 in LAB

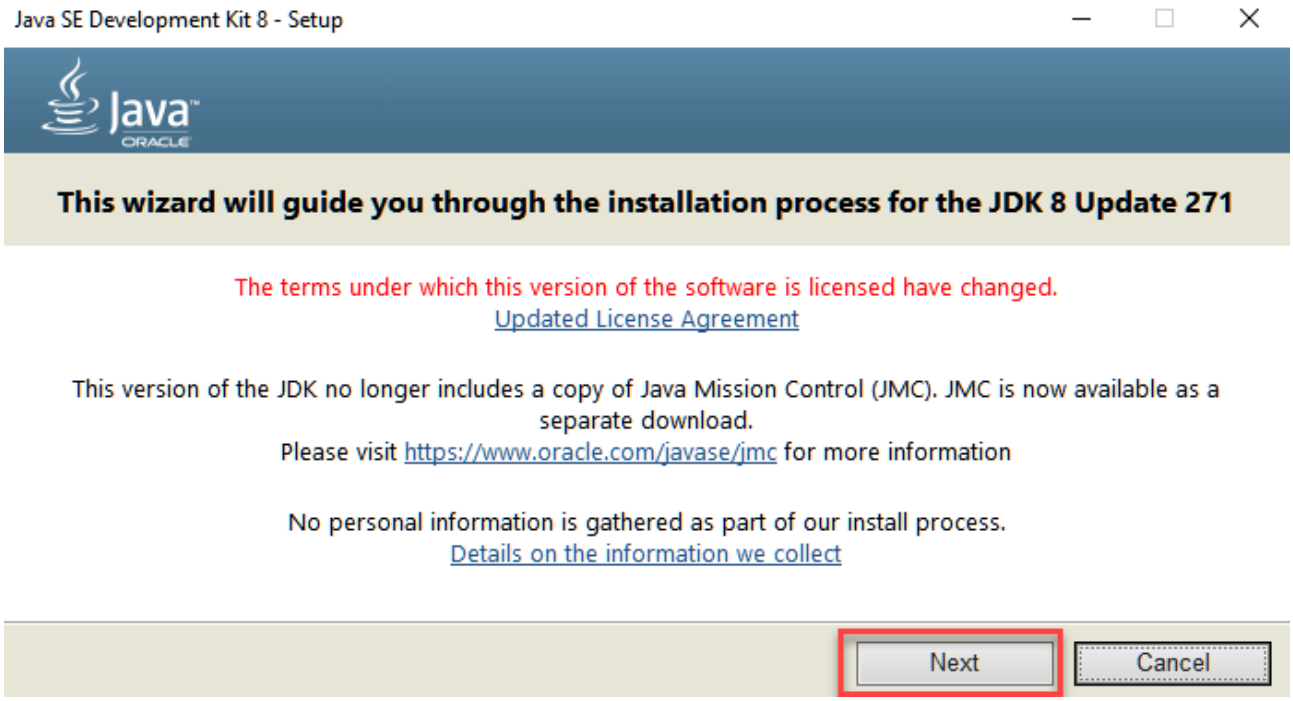
If you have previously installed older version(s) of JDK/JRE, un-install ALL of them. Goto "Control Panel" □ "Programs" □ "Programs and Features" □ Un-install ALL programs begin with "Java", such as "Java SE Development Kit ...", "Java SE Runtime ...", "Java X Update ...", and etc.

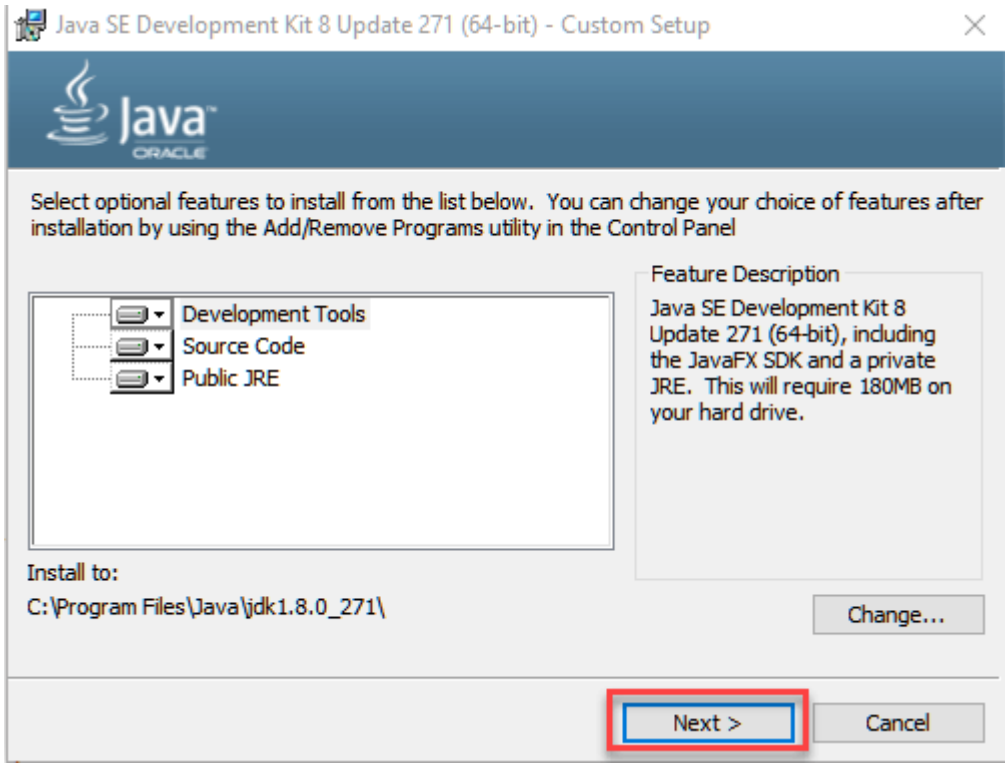
Step 1: Download JDK

1. Goto JDK (or Java SE) download site @ https://www.oracle.com/java/technologies/javase-downloads.html.
2. Under "Oracle JDK", click "JDK Download".
3. Download the "Windows x64 Installer" (e.g., "jdk-XX.0.{x}_windows-x64_bin.**exe**" - about 120-170MB)
4. Install the exe.



Solaris SPARC 64-bit	88.75 MB	jdk-8u271-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	134.42 MB	jdk-8u271-solaris-x64.tar.Z
Solaris x64	92.52 MB	jdk-8u271-solaris-x64.tar.gz
Windows x86	154.48 MB	jdk-8u271-windows-i586.exe
Windows x64	166.79 MB	jdk-8u271-windows-x64.exe


Java SE Development Kit 8 - Setup
Java
ORACLE
This wizard will guide you through the installation process for the JDK 8 Update 271
The terms under which this version of the software is licensed have changed.
Updated License Agreement
This version of the JDK no longer includes a copy of Java Mission Control (JMC). JMC is now available as a separate download.
Please visit https://www.oracle.com/javase/jmc for more information
No personal information is gathered as part of our install process.
Details on the information we collect
Next
Cancel


Java SE Development Kit 8 Update 271 (64-bit) - Custom Setup
Java
ORACLE
Select optional features to install from the list below. You can change your choice of features after installation by using the Add/Remove Programs utility in the Control Panel
Development Tools
Source Code
Public JRE
Feature Description
Java SE Development Kit 8 Update 271 (64-bit), including the JavaFX SDK and a private JRE. This will require 180MB on your hard drive.
Install to:
C:\Program Files\Java\jdk1.8.0_271\
Change...
Next >
Cancel


Java SE Development Kit 8 Update 271 (64-bit) - Complete
Java
ORACLE
Java SE Development Kit 8 Update 271 (64-bit) Successfully Installed
Click Next Steps to access tutorials, API documentation, developer guides, release notes and more to help you get started with the JDK.
Next Steps
Close

2. Write steps to set Java path in different ways on Windows OS.

The path is required to be set for using tools such as javac, java, etc.

If you are saving the Java source file inside the JDK/bin directory, the path is not required to be set because all the tools will be available in the current directory.

However, if you have your Java file outside the JDK/bin folder, it is necessary to set the path of JDK.

Include JDK's "bin" Directory in the PATH

Windows' Command Prompt (CMD) searches the current directory and the directories listed in the PATH *environment variable* for executable programs.

JDK's programs (such as Java compiler "javac.exe" and Java runtime "java.exe") reside in the *sub-directory* "bin" of the JDK installed directory. JDK's "bin" needs to be added into the PATH.

Prior to JDK 15, you need to *explicitly* add JDK's "bin" into the PATH.

Starting from JDK 15, the installation process adds the directory "C:\Program Files\Common Files\Oracle\Java\javapath" to the PATH. The "javapath" directory is a link to "javapath_target_xxxxxx", which contains a copy of the following JDK programs:

- java.exe: Java Runtime
- javac.exe: Java Compiler

To edit the PATH environment variable in Windows 10:

Two ways 1) Temporary

2) Permanent

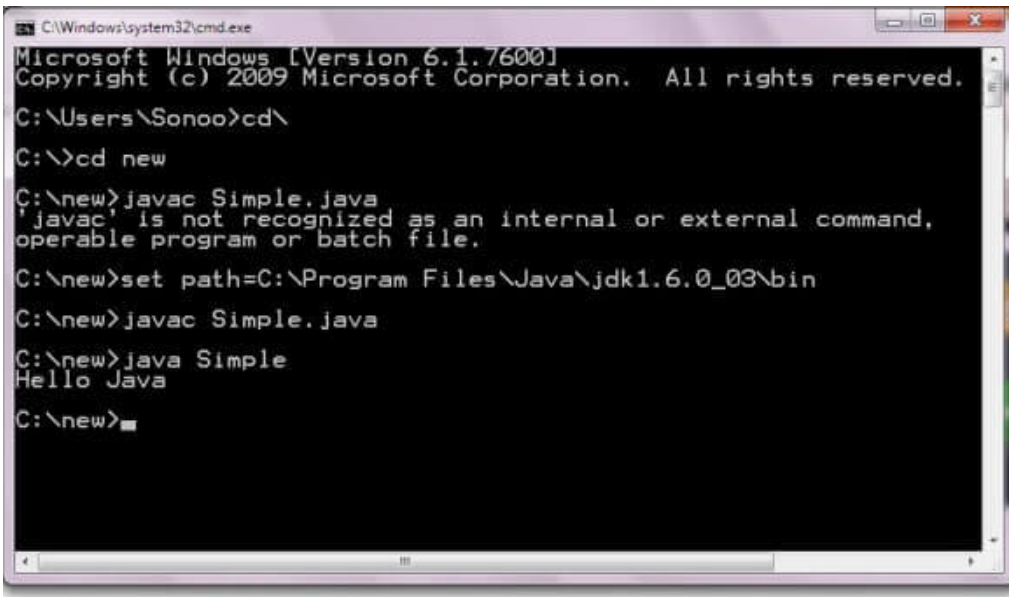
Temporary

To set the temporary path of JDK, you need to follow the following steps:

- Open the command prompt
- Copy the path of the JDK/bin directory
- Write in command prompt: set path=copied_path

For Example:

set path=C:\Program Files\Java\jdk1.8.0_19\bin

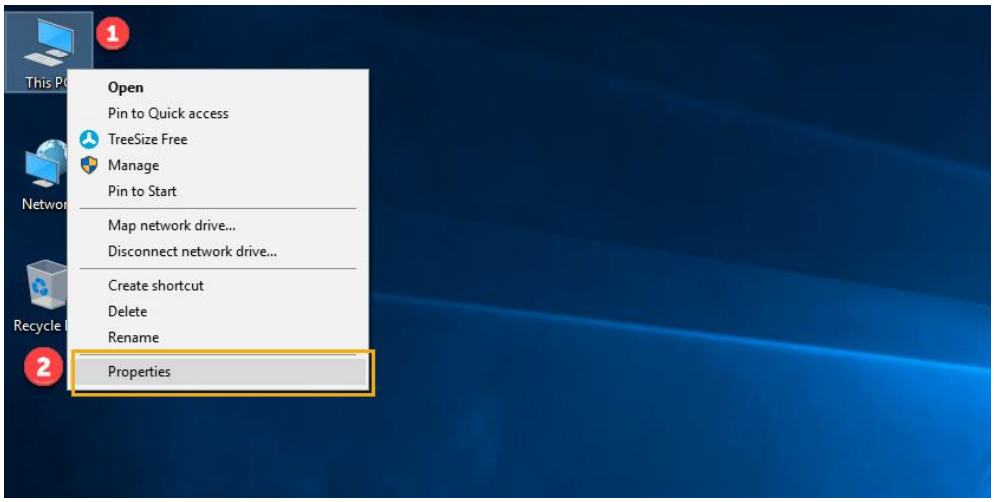


Permanent

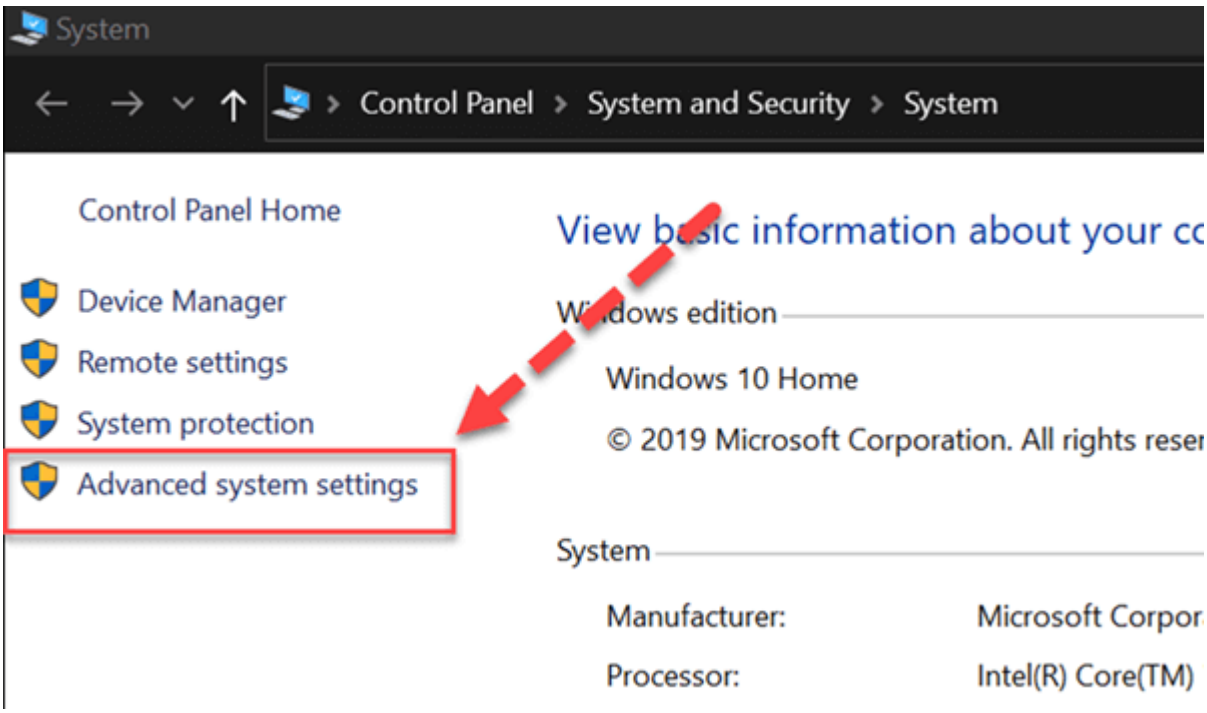
To edit the PATH environment variable in Windows 10:

1. Launch "Control Panel" □ "System and Security" □ "System" □ Click "Advanced system settings" on the left pane.

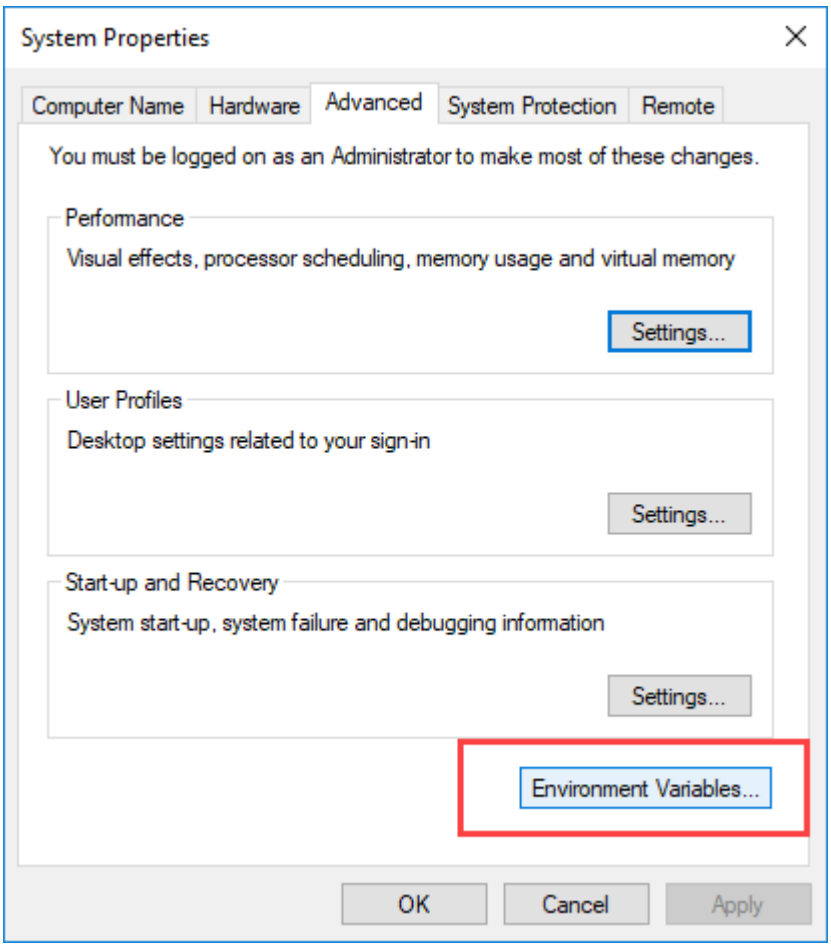
   or Right Click This PC and Select Properties their you can find the Advanced System Setting option

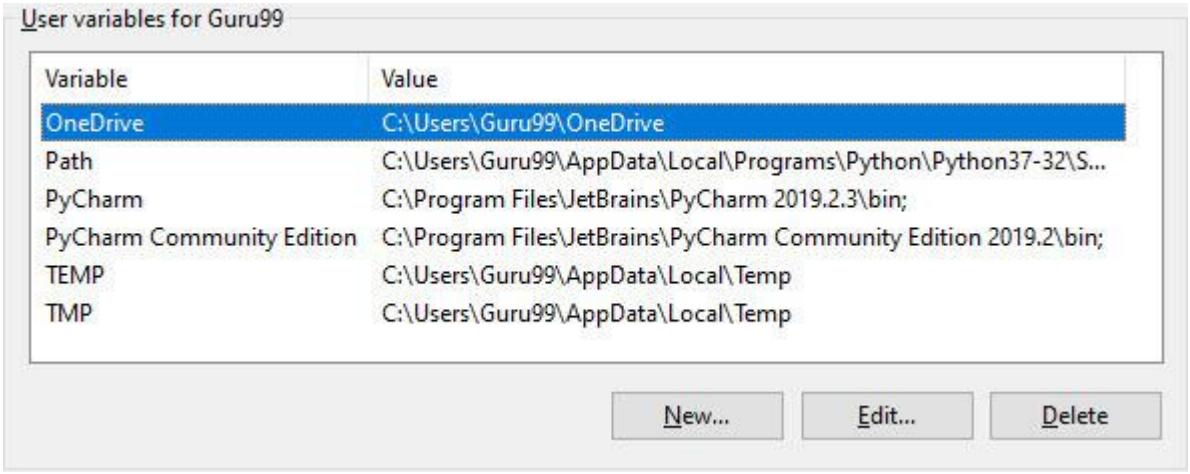


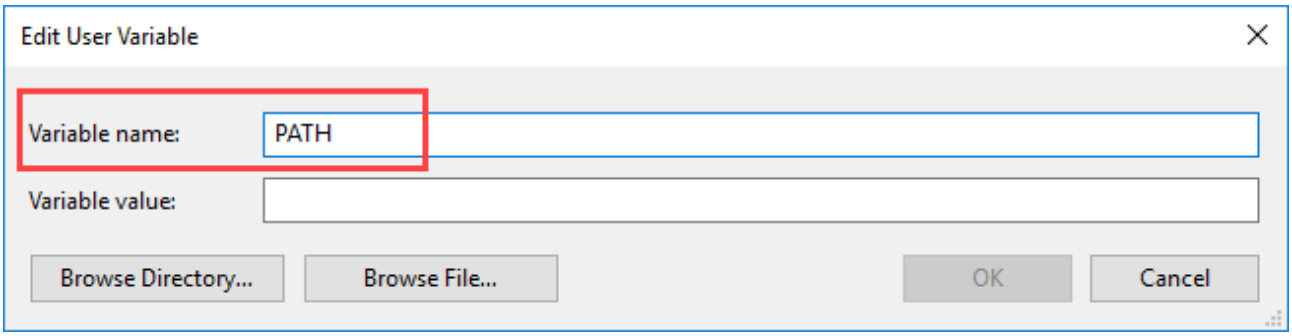
2. Switch to "Advanced" tab □ Click "Environment Variables" button.

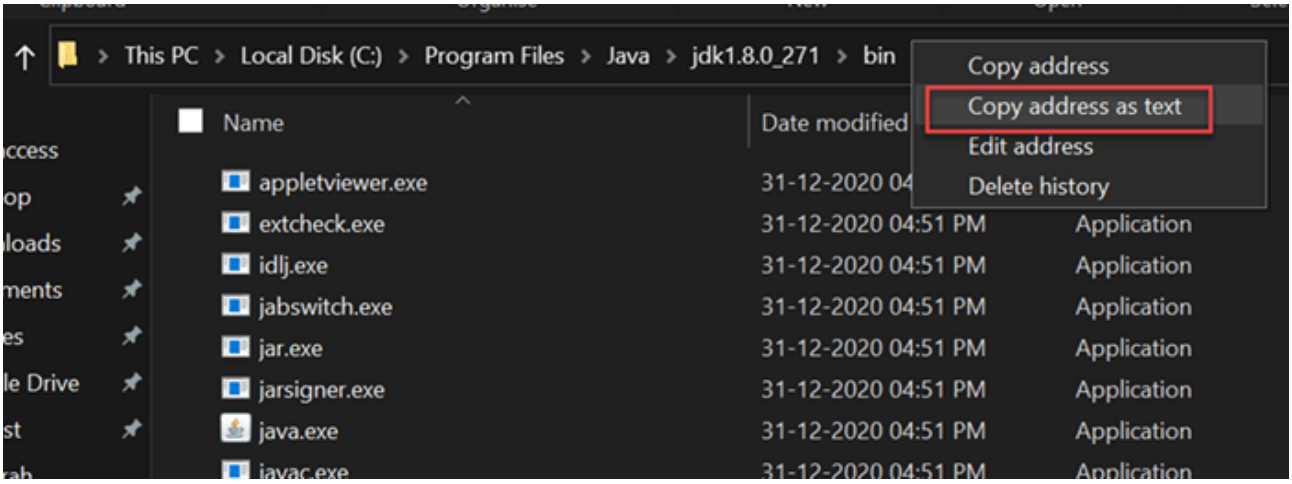


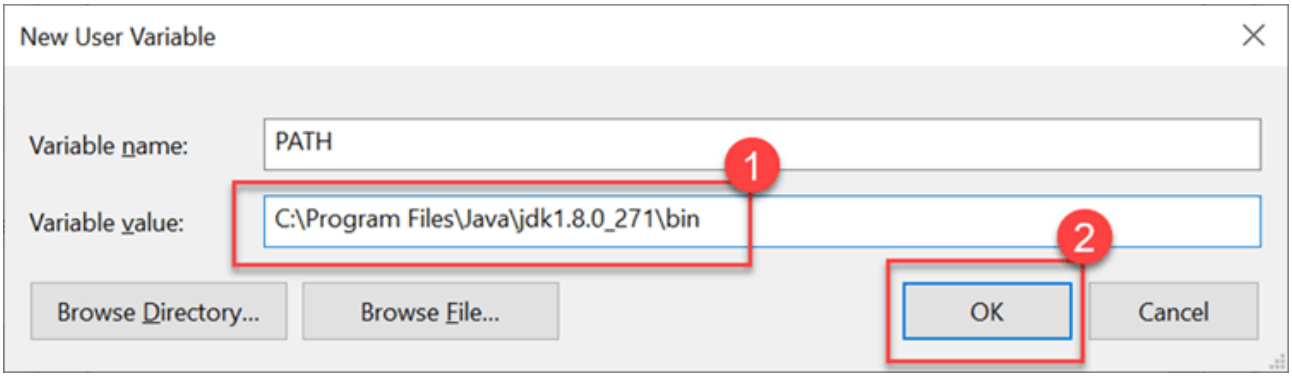
3. Under "System Variables" (the bottom pane), scroll down to select variable "Path" □ Click "Edit...".











4. **For Newer Windows 10**:
   You shall see a **TABLE** listing all the existing PATH entries (if not, goto next step).

Click "New" □ Click "Browse" and navigate to your JDK's "bin" directory, i.e., "c:\Program Files\Java\jdk-15.0.{x}**bin**", where {x} is your installation update number □ Select "Move Up" to move this entry all the way to the TOP.

if not present add to the new n\line and move up.

For Older Windows 10 (Time to change your computer!):
(CAUTION: Read this paragraph 3 times before doing this step! Don't push "Apply" or "OK" until you are 101% sure. There is no UNDO!!!)
(To be SAFE, copy the content of the "Variable value" to Notepad before changing it!!!)
In "Variable value" field, APPEND "c:\Program Files\Java\jdk-15.0.{x}\bin" (where {x} is your installation update number) IN FRONT of all the existing directories, followed by a semi-colon (;) to separate the JDK's bin directory from the rest of the existing directories. DO NOT DELETE any existing entries; otherwise, some existing applications may not run.
Variable name : **PATH**

Variable value : **c:\Program Files\Java\jdk-15.0.{x}\bin;[*do not delete exiting entries...*]**

Verify the JDK Installation

Launch a CMD via one of the following means:

1. Click "Search" button □ Type "cmd" □ Choose "Command Prompt", or
2. Right-click "Start" button □ run... □ enter "cmd", or
3. Click "Start" button □ Windows System □ Command Prompt

Issue the following commands to verify your JDK installation:

1. (Skip for JDK 15) Issue "path" command to list the contents of the PATH environment variable. Check to make sure that your JDK's "bin" is listed in the PATH.
   path

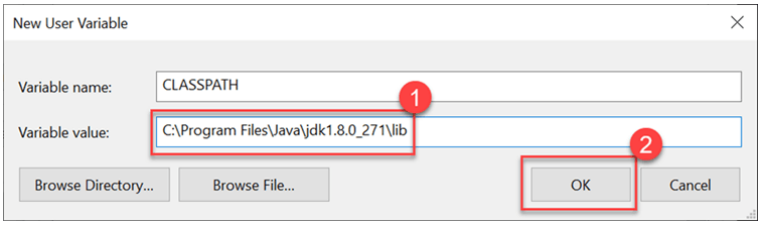
   PATH=c:\Program Files\Java\jdk-15.0.{x}\bin;other entries…

2. Issue the following commands to verify that JDK/JRE are properly installed and display their version:
   // Display the JDK version

   javac -version
   javac 15.0.1

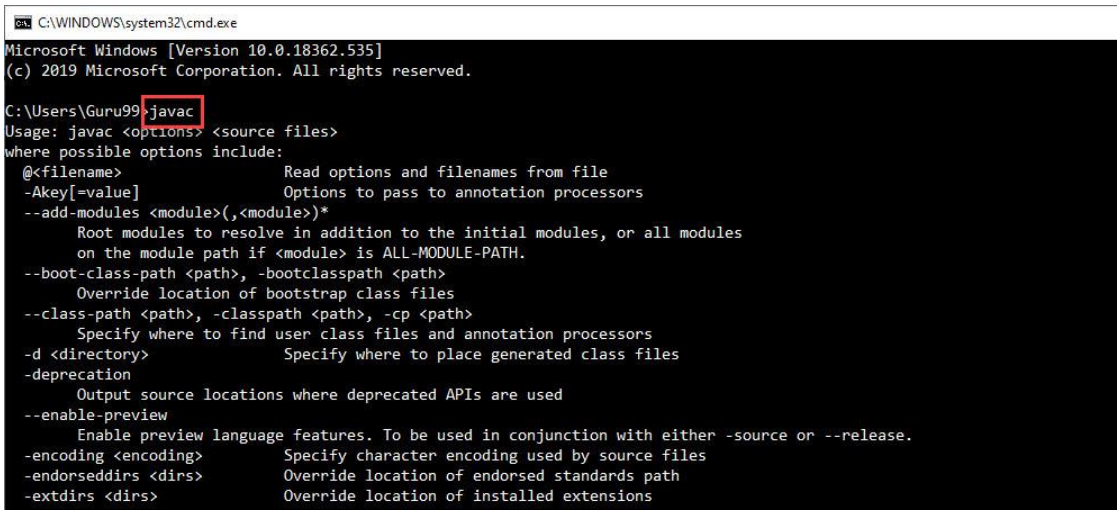
   // Display the JRE version
   java -version
   java version "15.0.1" 2020-10-20
   Java(TM) SE Runtime Environment (build 15.0.1+9-18)
   Java HotSpot(TM) 64-Bit Server VM (build 15.0.1+9, mixed mode, sharing)

Similar process can be carried out to set CLASSPATH if needed.(not needed now)



JAVA_HOME (if needed)

Check



Diff between path and classpath for info only

PATH − The path environment variable is used to specify the set of directories which contains executional programs.

When you try to execute a program from command line, the operating system searches for the specified program in the current directly, if available, executes it.

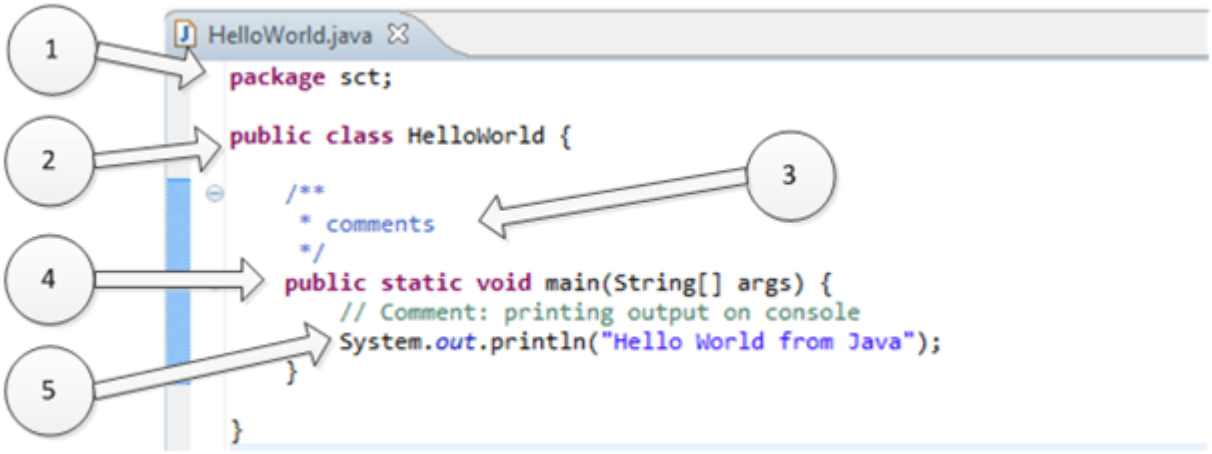
In case the programs are not available in the current directory, operating system verifies in the set of directories specified in the **'PATH '** environment variable.

CLASSPATH − The class path environment variable is used to specify the location of the classes and packages.

When we try to import classes and packages other that those that are available with Java Standard Library.

JVM verifies the current directly for them, if not available it verifies the set of directories specified in the **'CLASSPATH'** environment variable.

3. Write a simple Java program to print *"HELLO DDU"* on console. Explain the structure of a simple Java program.



1."package sct":

It is package declaration statement. The package statement defines a namespace in which classes are stored. The package is used to organize the classes based on functionality. If you omit the package statement, the class names are put into the default package, which has no name. Package statement cannot appear anywhere in the program. It must be the first line of your program or you can omit it.

2."public class HelloWorld":

This line has various aspects of java programming.

a. public: This is access modifier keyword which tells compiler access to class. Various values of access modifiers can be public, protected,private or default (no value).

b. class: This keyword used to declare a class. Name of class (HelloWorld) followed by this keyword.

3. Comments section:

We can write comments in java in two ways.

a. Line comments: It starts with two forward slashes (//) and continues to the end of the current line. Line comments do not require an ending symbol.

b. Block comments start with a forward slash and an asterisk (/*) and end with an asterisk and a forward slash (*/).Block comments can also extend across as many lines as needed.

4. "public static void main (String [ ] args)":

Its method (Function) named main with string array as an argument.

a. public: Access Modifier

b. static: static is a reserved keyword which means that a method is accessible and usable even though no objects of the class exist.

c. void: This keyword declares nothing would be returned from the method.

d. String[] args: It stores Java *command line arguments* and is an array of type *java.lang.String* class. Here, the name of the String array is *args* but it is not fixed and user can use any name in place of it.

e. Method content inside curly braces. { }

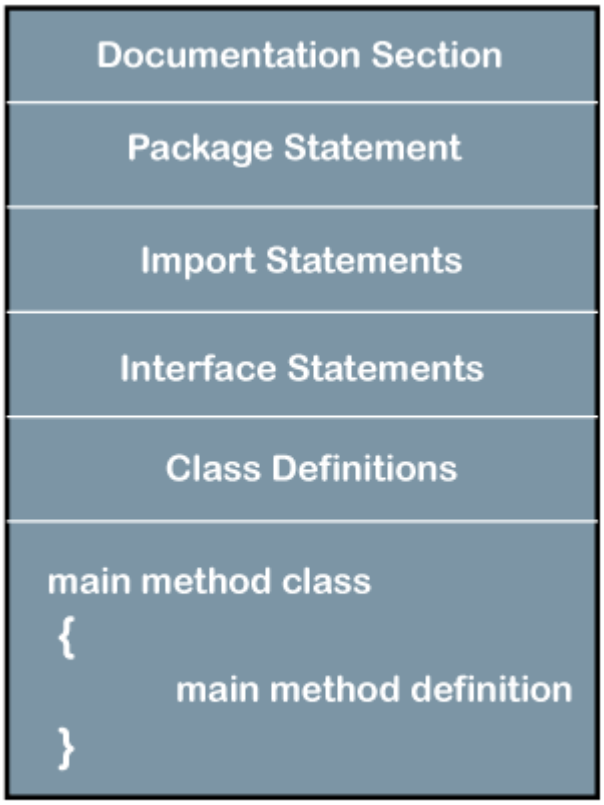
5. System.out.println("Hello World from Java") :

a. System: It is the name of Java utility class.

b. out:It is an object which belongs to System class.

c. println: It is utility method name which is used to send any String to the console.

d. "Hello World from Java": It is String literal set as argument to println method.



Structure of Java Program

4. Explain different Java commands used to run a simple Java program. Explain how to compile and run Java program using command prompt (DOS)?

The **javac command** is used to compile Java programs, it takes .java file as input and produces bytecode. Following is the syntax of this command.

javac HelloWorld.java

Compile three source files at once, type:

javac Program1.java Program2.java Program3.java

Compile all source files whose filenames start with Swing

javac Swing.java*

Compile all source files:

*javac *.java*

The **java command** is used to execute the bytecode of java. It takes byte code as input and runs it and produces the output. Following is the syntax of this command.

5. How to add Java commands in the TEXTPAD editor? Also write shortcut keys to run and compile a Java program.

0. Install TextPad. **Be sure to do this AFTER Java is installed.**

1. Start TextPad.

2. Under 'Configure' at the top menu, select 'Preferences'.

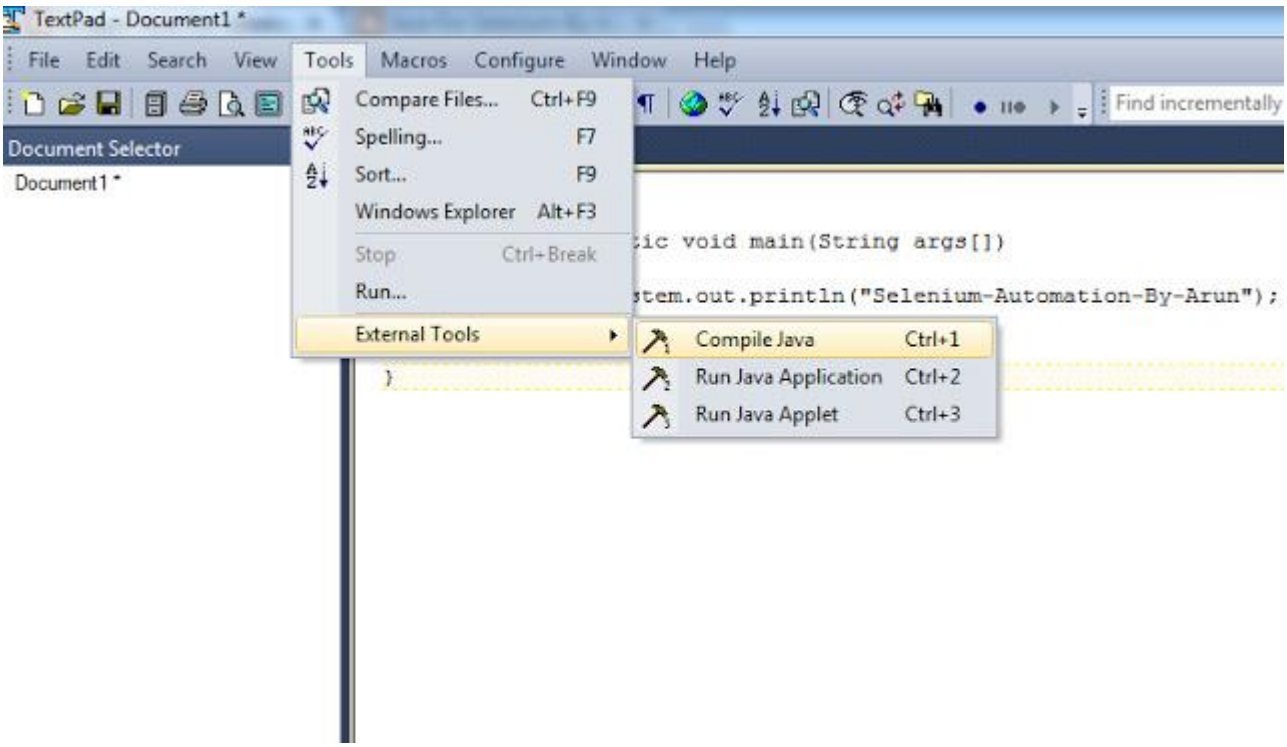
3. In the pop-up menu, expand 'Tools'.

4. In the sub-menu, select 'Compile Java'.

5. In 'Parameters' on the right pane, add in front of $File " **-classpath . $File**". Be sure to put a space before and after .

6. Click OK.

7. Do the same for 'Run Java Application' under the 'Tools' menu, that is, 'Configure' -> 'Tools' -> "Run Java Application', and in the right pane, make the 'Parameters' box look like "**-classpath . $BaseName**".


TextPad - Document1 *
File Edit Search View Tools Macros Configure Window Help
Find incrementally
Document Selector
Document1 *
Compare Files... Ctrl+F9
Spelling... F7
Sort... F9
Windows Explorer Alt+F3
Stop Ctrl+Break
Run...
External Tools
Compile Java Ctrl+1
Run Java Application Ctrl+2
Run Java Applet Ctrl+3
tic void main(String args[])
stem.out.println("Selenium-Automation-By-Arun");
}

6. Write a simple program to collect command line arguments from user and print on console using Dos prompt and Textpad.

```
class A{
        public static void main(String args[]){
                for(int i=0;i<args.length;i++)
                        System.out.println(args[i]);
        }
}
```

7. Write a Java program to print the multiplication table of the given number from command line argument.

```
public class Multiplication_Table
{
        public static void main(String[] args)
        {
                int n=Integer.parseInt(args[0]);
                for(int i=1; i <= 10; i++)
                {
                        System.out.println(n+" * "+i+" = "+n*i);
                }
        }
}
```