Chapter-1

Introduction:

Dulal Acharjee

JAVA is a general purpose programming language. This language can be used for normal programming, network-internet programming, Graphics, hardware and software interfacing development purposes. JAVA has also low level programming features which can be used for hardware programming. Sometimes it can be used for embedded programming, i.e. keeping the code in ROM(Read Only Memory) into electronics instruments and running the machine according the program residing within the ROM chip. JAVA is a fully object oriented language and every individual program is written as a Class. Later, within this book, Class and Object will be discussed in details.

History of JAVA:

In 1991, Sun Microsystem leaded by James Gosling of a team named as “Green Team” formed a committee to develop a more popular language which can give better support to internet programming. Previously its name was Oak, in 1995, it was renamed as JAVA.
Now a days JAVA is owned by ORACLE Inc.

In around 1991, Sun Microsystem developed the JAVA. Previously its name was Oak; in 1995, it was renamed as JAVA. In 2010, Sun Microsystem was acquired by ORACLE Corporation, so, now a days JAVA is the product of Oracle.

Within a programming language many software modules exist and in JAVA these are:

- **Compiler:** javac.exe resides within …/bin/ folder. This program compiles the java program and makes another compiled program name as ‘class’ file. Compiler checks and analyses the source program lexically whether the grammatical rules of that language (e.g, java) is maintained or not. If all syntaxes are written properly within the source program, then, a new file is generated which is called object code or machine code or byte code having larger size than the source program. Compiler can find exact errors of source code, type and nature of errors, line number and position of the error within the line and the probable corrections are suggested by compilers also.

- **Interpreter:** java.exe resides within …/bin/ folder. In java, it is the interpreter of compiled program or '.class' file. It is called as Java Virtual Machine (JVM) which can interpret all lines of the .call file and give output. For different machines like: Apple, IMB compatible, Mainframe, Linux/Unix based computers, different versions of JVM are available which version of JVM would be required to interpret the .class file to get output.
- **Editor:** not standard editors are available, but, jshell.exe is there within /bin folder which a line editor and small programs or one line statement can be written and executed.
- **Translator:** its role is to convert one form of code to another form. In JAVA, it may be thought that both javac.exe and java.exe converts code one form to another form. So, these two(compile and interpret) are the examples of translation.
- **Debugger:** jdb.exe is a simple command line debugger to check and inspect any errors if available within a class.
- **Libraries:** these are some classes, packages required to compile or translate the code. Generally, to understand, *'.dll, .h, .class* and packages' are examples of libraries.
- **Applications:** many in built application development tools are available within JAVA package. As example, making .exe project, version management, documentation preparation etc.

Features of JAVA:

Different features of JAVA can't be summarised in a short paragraph, but, some important features are mentioned below:

1. JAVA is a General Purpose Language and using this language, any type of programming applications can be developed.
2. Generally called machine independent language and the key point of machine independence is the concept of Java Virtual Machine(JVM). It is an interpreter of bytecode or machine code of the java program for a particular type of a machine. Java byte code, .class file, is a compiled code of the program which should be interpreted with the help of a interpreter and its name is 'java.exe'.
3. Portable: Source code is required to compile which generates machine code or Class file. This class file can be taken to any machine say, IBM, Apple, Linux OS for execution.
4. Java Virtual Machine or JVM: The machine code for the Java Virtual Machine is known as Java bytecode. Machine code can be taken to other computers like: Apple, Linux OS based machine, Mainframe computer etc. and can be interpreted there with the help of JVM of that version of JAVA for that computer. Or in other words, we can say, JAVA has developed different versions of interpreter for different machine, so why it has become possible to interpret *.class* file to be executed in other computer.
5. Internet Programming Language: when JAVA was designed and developed, at that decade internet started to explore. Internet is a network of connected computers and

which is the basis of communication engineering. If data is required to send from one computer to another, it transmits data in the form of packet. To transmit or receive data, some protocol or rules of formation of packet, header address, encoding rule, transmission and reception speed etc. are required to formulate. These rules are known as protocol. There are many communication protocols like TCP(Transmission Control Protocol), UDP(User Datagram Protocol), IP(Internet Protocol) etc. JAVA programming made it easier to write these network protocols and other communication programs. Within JAVA, 'servlet' package contains required Class, methods, constructors to develop communication program.

6. **Good web page designer:** in a webpage there are many jobs integrated at a place like: designing frontend with the help of HTML, WML, XML scripting codes within JSP code, connection with the server, writing the server side program to receive the request from the client under internet connections. There are mainly five JAVA tools to make these jobs easier and these are: (i) JavaServer Pages (JSP), (ii) Servlet API(servlet, filter, filterchain, servletconfig etc.) (iii) Java Persistence API (JPA) (iv) Java Database Connectivity (JDBC) (v) JavaServer Faces (JSF). These tools provide all necessary components required to design and develop a large webpage. For a webpage development required jobs are: markup language which can make link with another server page from a webpage, read-write data from/to databases residing within the Server either of Database Server provider or webpage server provider, writing socket programming for client-server communication etc. Using these tools, all of these jobs essential for webpage development can be performed efficiently.

7. **JAVA has four editions:** these are known as Platforms of JAVA where all programming tools are available. Each of the editions contain Java Virtual Machine(JVM) and Application Interface (API). JVM is the translator of *.class* file under a particular hardware, but, after the version of 11, JVM alone can translate and execute any *.java* program. API's are other components of java language which are required to develop any application program. When application specific development is required, then, any of these editions can provide particular solutions:

i. **Client-Server Edition or Standard Edition (SE):** this edition of JAVA is known as general programming edition with provides all facilities of uses of Classes, GUIs, networking application developments, database connectivity, server connectivity, XML parsing etc.

ii. **Server side applications or Enterprise Edition (EE):** to develop multi-tier based network application EE is used. This edition is developed upon the SE of Java, so, there is a lot of components common between these two. Before, its name was J2EE but now it is known as Jakarta EE. It provides features of enterprise software development for distributed systems and web services. Large enterprise software are developed using EE of JAVA.

iii. **Mobile apps or Micro Edition(ME):** like other editions, ME also has separate API and JVM for developing applications for small electronic devices like mobile phone, microwoven, washing machine, airconditioner etc. Generally, small electronic devices run by embedded program loaded into a ROM chip. The microcontroller which run the smart electronic devices are controlled by program residing within memory of the device (RAM / ROM) and these types of programs can be written by ME of Java.

iv. **Java platform Effects(**JavaFX): it is a part of SE of Java and available as API within Standard Edition. It is a set of graphical and media packages to design, develop, test and deploy dynamic and beautiful graphical interfaces. For developing qualitative internet interactive GUI applications JavaFX is used. It is a open source toolkit to develop modern, rich and internet based dynamic graphical user interface type of client application. It runs of JDK(Java Development Kit) and starts from JDK version 11. For most of cases, on the desktop of client, different user interactive scene of pictures are developed using JavaFX.

8. **Object Oriented Language:** Java is a fully Object Oriented Programming Language. All contents of the Program are written within a class by starting a '{' and also ended by '}' and the name of the program must be the same as the name of the class. Though is known as 'Object Oriented Programming Language' but, it is really 'Class Oriented Programming Language'. An object is the instance of a class, i.e. at the time of coding; all works are done with object not from class. A class is a virtual thing but object is a physical model of a class residing within the memory of the computer. A class has many components like: variables of different data types, constructors with different arguments, methods with different arguments etc. These components can't be called or used directly but through object. A class can create another class; a class can create an object. When a class generates another class, the new born class gets all properties of the father class- it is known as inheritance. There are some rules of inheriting properties of super classes. When properties of grandfather and father classes are inherited within child class- it is known as multiple inheritances.

9. **Thread/Multithreaded:** thread is a piece of code within a process and it is written for giving protection or security to critical data section. Multithread is multiple thread units written within the same process or within different processes to give protections to critical section of the program that concurrent processes may not interrupt the critical section and may not cause erroneous output. A thread has five states as: **New, Runnable, Running, Waiting/Blocked, Dead/Terminated** and by these conditions of a thread intervention from outside is managed and controlled. Within Java, the 'Thread' class under '*.lang*' package is available to manage Multithread/Thread related jobs.

10. **Distributed system:** JAVA is a fully distributed language and most of the components of this language are distributed among different packages, classes and methods. Using many of its components JVM or compiler works. A distributed system uses many resources like: memory, disk space, read/write ports, target address, temporary buffers, file systems, server access, uses of databases under different systems etc. all of these resources may be required to process the central processing system. It uses different resources distributed at different locations of

either in same machine or in different machine, so why it is called a distributed system.

Some topics related to Java:

Atomic Operation: it is thread related an execution concept which says there are some critical operations under multithreaded environment, but, these types of operations will execute only once at the proper time as mentioned within the program. It has another meaning – execute the smallest unit of the program, break into smallest unit and then execute and check results. As example, there are some operations which need to be completed as unit, before completion of that unit if the thread halts, stops or do interchange data with other threads, errors may happen. As an example, x += 1; it has three steps as say: x=0; increase value of x by 1; take back the value to x as : x= x + 1 = 0 +1 = 1; now it is clear that before completion of these jobs, if any interrupt happens within the thread, result will be error and which is dangerous for valuable data. So, safety of code is essential. In Java, thread protection has been designed with safety.

Garbage Collection: JAVA has automatic garbage collection and freeing memory mechanism of unused components which may reside within memory of the machine. When some variables of objects are created and assigned, these are linked with some memory space of the machine, but, when work with those components are over and program ends, some components also resides into the machine, these unused components residing within machine are known as garbage. Generally using the instructions like: free, kill, close, delink etc. created components are freed of delinked from the memory that those spaces may be used for other programming purposes.

Different Programming Languages:

High Level programming language: It has another name as 'general purpose programming language'. It is a programming language using which any type of programs can be written as example: financial, store management(inventory), Hotel/college/School management system, Graphics applications, network/web programming, Server programming, Driver programs of any hardware like Hard Disk, microphone(audio driver), Video driver, monitor/display driver, encryption/ description/ compression, file management, data management etc. The instruction sets of high level programming language are nearer to English language, so , it is easier to learn this types of programming languages. Examples are: BASIC, C, C++, JAVA, PASCAL, Fortran, Cobol, Python etc. In all of these computer languages, instructions are designed maintaining some grammatical rules which are known as syntax. All commands or instructions should be written as per syntax of the language.

Low Level or Machine language: these languages are used for hardware programming directly. These instruction sets are not like English languages but as different encoded forms. These instructions can address the ports, memory location, scanning of VDU(Visual Display Unit), interrupt control to microprocessor, read/write data from Modem(Modulation and Demodulation), math and logical operations within registers of microprocessor/microcontroller etc. Examples of low level programming languages are: Assembly language, ASM(assembler), TASM(turbo assembler), Machine Language etc. Assembly language based codes are required to convert to Machine code and then only the digital electronic circuits can execute those codes. There are some interfaces using which machine codes can be directly entered into the machine. Low level programming executes the machine very fast and size of the program is smaller.

Note: As this book is for the beginners of JAVA programming and also for learning practical programming with theoretical concepts, at first, it is asked to create a subdirectory by name 'example' where all working programs will be stored. It can be created in any drive like C:, D: or E: etc. Our folder name is E:\example> and in this Book we shall use this folder for storing all examples of programs. We shall create, edit and run any program from this folder. So, we shall consider this is our home folder.

How to write a simple program in JAVA?

JAVA program is a simple text program file and can be written in any text editor. But, care should be taken to give name of the file. As example, in notepad of Windows when any file is saved '.txt' extension is added automatically. But, if we write JAVA program, we have to give file extension as '.java'. To do that under editor, notepad, you should select:
File>Save As>Save As Type>All Files>give name as 'Hello.java' and then save.

```
// Hello.java: to display any message.
public class Hello
{
   public static void main(String[] args)
  {
     System.out.printf("My Name is ABC…");
  }
}
```

Run the Program:

Above program is saved as 'Hello.java'

Compile the program:

```
E:\example> javac Hello.java
```

This command '*javac*' compiles the program 'Hello.java' and generates a *class* file by name '*Hello.class*' as seen below. This file is the bytecode of the program and can be run by any interpreter of java program for this computer. Or in other words, to run this bytecode 'Hello.class' we need an interpreter for operating system Windows which is 'java.exe' available under /bin/ folder of java.

To run the program of *Hello.class* we have to give the command as:

```
E:\example>java Hello
```

We got the output "My name is ABC" as shown below.

N.B. at the time of installation of java, environmental path is set automatically by the installer. In some old versions of jdk(Java development kit) path is not set automatically, it is required to set manually by adding the path of present folder of 'bin' within the 'Environment variable' of windows.

Another simple example:

```
//HelloPrint.java: printing many lines,
// println()  prints at new lines from starting point.
class HelloPrint
{
        public static void main(String args[])
        {
                System.out.println("ABC");
                System.out.println("XYZ SDF");
                System.out.println("PQR MNB XOP");
        }
}
```

OUTPUT:

```
ABC
XYZ SDF
PQR MNB XOP
```

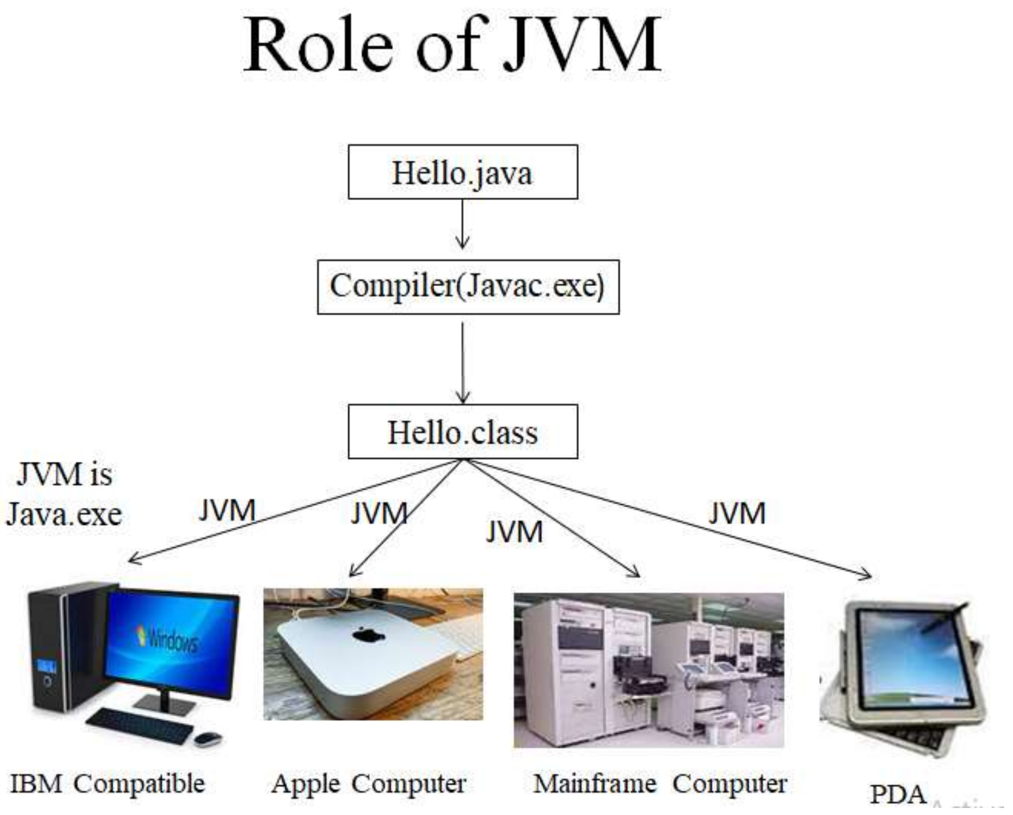

Fig.1.1 describes Role of JVM, Java Virtual Machine, under different hardware of computers.

Another way of running the java program:

From the version of SE 11, it is not required to compile the program; it is made simpler, only the interpreter, *java.exe*, can execute any *.java* program without compilation as shown below:

```
E:\example>java Hello.java
My Name is ABC
```

Observe that to run the *.java* file, when we give the command:

```
java Hello.java
```

Or

```
Java E:\example\Hello.java
```

Same output is displayed in both the cases. When we give command of running a program, the working default directory is used and *.java* file will be searched from current directory. As an example, we have given command from E:\example> directory and this is the working directory, so when we want to compile a program as **E:\example> javac abc.java**, then compiler will search the 'abc.java' within the working directory, if the file is not found, then, it will give error message. But, as path has been set to environmental PATH with the JAVA subdirectories where 'javac.exe, java.exe' executable files are residing, we can give these commands as C:\> java or D:\>javac etc. from any directory or subdirectory, but, you have to mention the path of your java program file. The best way is to give *java* or *javac* commands from the directory where all examples are residing as:

```
E:\example> javac abc.java
```

Suggestion: you should create a subdirectory by name 'example' in D:, E: drive, but, if these drives are not available in your computer, you may create within C: drive also.

Installation of JAVA

In 2010, JAVA is purchased by ORACLE; till version **jdk 8.0** it was under Sun Microsystem, but, now a days all components of JAVA are the property of ORACLE Corporation.
For installation you should read guideline and license statements available in Oracle website.
If you want to install **jdk 16.0**(Java Development Kits), then, **jdk-16_windows-x64_bin.exe** should be downloaded. It is of windows 64 bits latest version; it is free version for non-commercial uses. By double click and following next instructions step by steps you can install the JDK(Java Development Kits). You should not download JRE which is the runtime executable environments and does not contain 'javac.exe' file for compilation.
It will be installed within **C:\program files\java\** folder by default or can be installed in user defined directory also. You should check this folder whether Java is installed and can be run from any drive or not by giving the instructions:

```
C:\> javac
```

C:\>java

In Java version 16.0, path is set automatically. This version is for advanced users. Java 8.0 has 'appletviewer.exe' to develop graphical applications. For beginners of JAVA programming, version 8.0 is good for them but setting environment path may disappoint you. It is too old version, instead, we suggest to download JDK 16.

If version 16.0 is installed, environment path of Windows is set automatically, you don't need to set path manually. From any drive C:, D:, E: , if you give the commands mentioned above, it will show help instructions of these two commands, that means, system could identify these two executable programs and the path setup is done automatically by the JAVA installer.

Setting Path in Windows for JDK 8:

You can set path manually as:

- Editing *windows environment variable* and adding the path of 'c:\program files\java\jdk-16\bin' as an additional path. If you install other versions of JDK, then, instead of jdk-16 other texts will be displayed and that should be written in that place.
- Reboot the computer.

When java is installed path is automatically set for java commands, as an example, path of the machine where program was tested is shown below:

```
PATH=C:\Program Files\Common
Files\Oracle\Java\javapath;C:\WINDOWS\system32;
C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPo
werShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;C:\Program Files\Java\jdk-
16\bin;C:\Program Files\MySQL\MySQL Shell 8.0\bin\;
C:\Users\DULAL\AppData\Local\Microsoft\WindowsApps;
```

Here, for understanding path setup, it is underlined.

N.B. if path of **../bin** of java is not set at the time of installation automatically, then, it should be set through environmental parameter setup of windows. When we compile or run the program, required libraries should be accessed from the working path. For that path may be required to update.

Setting path manually for Windows 10:

1. Search the word 'env'
2. Click on 'Edit the System Environment variable' (n.b. this will set path for all users of your computer. If multiple user accounts are there in your computer, then, Edit Environment variable for your account to be selected).
3. Advanced>Environment Variable
4. From the window of system variable>path>edit>new> type these words "C:\Program Files\Java\jdk-16\bin" (n.b. for your computer path may be different, but, this is the way to edit path.)
5. Then select OK, OK (your wok is done).

Path Setting for other versions of Windows 7/Windows 8:

1. Control Panel> User Accounts and Family Safety>User Accounts>
2. Change my environment variable> System variable > Path>edit

(n.b. for different versions of windows, these names of options may vary.)

After installation, to check, the basic structure of java software is as mentioned below:

```
Directory of C:\Program Files\Java\jdk-16
10-04-2021  17:00    <DIR>          .
10-04-2021  17:00    <DIR>          ..
10-04-2021  17:00    <DIR>          bin
10-04-2021  17:00    <DIR>          conf
08-02-2021  12:02             3,244 COPYRIGHT
10-04-2021  17:00    <DIR>          include
10-04-2021  17:00    <DIR>          jmods
10-04-2021  17:00    <DIR>          legal
10-04-2021  17:00    <DIR>          lib
10-04-2021  17:00             1,187 release
```

To learn structure of Java language, we suggest opening each subdirectory and checking contents under it. Gradually, we shall explain some components later on.

Problem: write a program to display some message using print(), printf() and println() methods. Discuss different versions of print methods.

```
//Example: Hello2.java: this program shows different print methods.
import java.lang.*;
public class Hello2
{
        public static void main(String args[])
        {
                System.out.printf("My Name is ABC XYZ PQR");
                System.out.print("\nMy College name is: AAAAAAA PPP");
                System.out.println("\nAddress of my College:  XXXX YYYY ZZZ");
        }
}
```

Discussions: a method can be used in different ways as mentioned within the rules set of java language. There are six different ways print() can be used and these are: print(char s), print(char[] s), print(boolean x), print(double d), print(long l), print(float f), print(int x), print(Object ob).
As argument of print(…) method different variables can be used as mentioned here.
A general format of a java program is mentioned below:

package-declaration

```
optional-import
public class program-name
{
        #variable declarations and subroutines declarations;
        public static void main(String[] args)
        {
            statements ;
        }
#Variable declarations, subroutines declaration, statements etc;
}
```

Discussion: like other programming language, JAVA has standard template to write any program. Standard template is like:

1. Package declaration; it is optional, if you want to keep this program under a package, then this declaration is required.
2. Importing of existing packages of JAVA language.
3. Starting of main class
4. Starting of main() method
5. Close of main() method
6. Close of class module

These are the general rules to be followed, but, there are some exceptions which will be learnt gradually.

Problem: write a program to assign two numbers within two integer variables, sum two numbers and store value to another integer variable, then, print/display on the monitor the output of the program.

```
//Number.java: to process some numerical operations
class Number
{
   public static void main(String args[])
  {
        int x = 5;
        int y = 12;
        int z = x+y;
        System.out.println(" Result = " + z);
  }
}
```

Symbolic constants:

Variable names and values can be reassigned, then, new set of variable is created with new value, but if we need to create a variable which will remain unchanged with its value, then, we should declare it as ‘final’ identifier. When ‘final’ identifier is given to any variable, then it can’t be recreated, remain fixed throughout the program.

```
 final int AGE=23;
 final float LENGTH=12.50;
```

It should be declared within the body of the class. It should remain unaltered throughout the program.

Different operators:

Like school algebra, operator symbols, + - * / , perform the same role in programming. Symbols look like an alphabet but internally it does a particular operation inside the machine or in generalized language we can say each of the symbols of operators perform a particular function.
Operators are symbols which mean or do some operation on one or more variables/parameters. There are different types of operators as:
Arithmetic Operators: These are the mathematical operators works with one or two variables. As examples: a + b ;   a – b ; a * b ; a / b ; a % b ; etc.

Operator	Function	Read as
+	Adding left and right variables/data	Plus sign
-	Minus right data from left one	Minus sign
*	Multiple of two vars/data	Multiplication Sign
/	Divide left data by right data	Division Sign
%	Modulus after division; returns remainder after dividing left by right	Modulo sign

Problem: write a program(wap) to declare any variable as *double* , make square root and hold result in another variable, then print the result.

```
//Example: SquareRoot.java:  using methods available under Math class.
import java.lang.Math;
class SquareRoot
{
   public static void main(String args[])
  {
          double x = 5;
            double y;
            y = Math.sqrt(x);
            System.out.println("y = " + y);
  }
}
```

Run the program:

```
E:\example>java SquareRoot.java
y = 2.23606797749979
```

Discussions: this program uses the Math class available under inbuilt '*lang*' or language package. Within the main() method, the computing method of sqareroot is 'sqrt()'. After decimal point there are 14 digits. (Any floating point number has four components as: a sign, a mantissa, a radix, and an exponent. The value of the float or double number is computed as:

sign * mantissa * radix $^{\text{exponent}}$
For datatypes of float and double, this mantissa and radix rules are different:
For Float : sign bit=1, exponent=8 bits and mantissa=23 bits.
For Double: sign bit=1, exponent=11 and mantissa= 52 bits

Different other methods available under Math class are:
Math.abs(x): returns the positive value of the number 'x';
Math.sin(x), Math.cos(x), and Math.tan(x): all output values of trigonometric functions are computed in radians not in degrees.
Math.asin(x), Math.acos(x), and Math.atan(x): inverse trigonometric functions like: arcsin, arccos, and arctan can be computed using these methods. Some other Math class method are:
Math.pow(x,n): returns (x to the power of n)
Math.exp(x): returns e^x where e is the base of the natural logarithms known as Euler number = 2.718281828459045 and x is of double data type.
Math.random(): returns pseudorandom numbers as > 0 and <1;
Math.floor(x): returns nearest number. E.g. Math.floor(995.21) = 995.0
Math.floor(-699.98) = -700.0 ; Math.floor(0) = 0.0; it follows the rule is that if the decimal part is >0.5 then one is added mantissa and if <0.5 then zero is added with. But in both cases returns full number eliminating decimal part.

Math.ceil(x): returns the integer part of the number eliminating the decimal part if any. E.g. Math.ceil(205. 456), the output is 205.

Math.PI: gives constant value of PI (pie, π) declared within java as: static double PI.
Value of PI= 3.141592653589 (approx.).
Math.log(x) : returns natural value of log(x) having base e.
Math.round(x): returns the nearest integer number, eg. Math.round(3573.845) = 3574; but, rounding any number after decimal position is done by the statement: String.format("%3.2g%n", 532.912300)

Program with Trigonometric formulae:

Mainly, Sin( ), Cos( ), Tan( ), Cosec( ), Sec ( ), Cot( ) are used in trigonometric computing. These are some ratios and have no unit values. As example,

$$\text{Sin(x)} = \frac{\text{Length of perpendicular}}{\text{Length of Hypotenuse}} = \frac{P}{H}$$

Sin, cos, tan etc. values are a ratio of two sides of a right angled triangle; it has no unit.

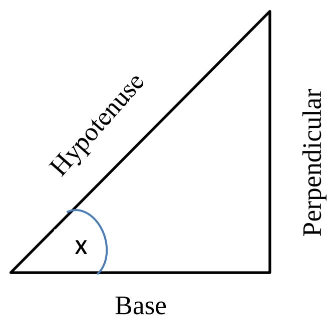

Fig.1.2 diagram of a simple perpendicular triangle showing Hypotenuse, Perpendicular and Base

Problem: wap to convert different degrees like: 30,45, 60 and 90 to radians and then diplay values of sin(30), sin(45), sin(60) and sin(90) using methods of **Math** class of JAVA.

//Trigono1.java: different values sin(30), sin(45), sin(60) and sin(90)
//will be computed.
// Theory: 180^0 = π radian =3.14 radian. We need to convert degree to radian value.

```
import java.lang.*;
class Trigono1
{
        public static void main(String agrs[])
        {
                double r, r1, r2, r3;
                // first convert degree in radians
                r=Math.sin(30*(3.14/180));
                r1=Math.sin(45*(3.14/180));
                r2=Math.sin(60*(3.14/180));
                r3=Math.sin(90*(3.14/180));
                //display output
                System.out.println("Sin(30)=" + r);
                System.out.println("Sin(45)=" + r1);
                System.out.println("Sin(60)=" + r2);
                System.out.println("Sin(90)=" + Math.ceil(r3));
        }
}
```

```
E:\example>java Trigono1
Sin(30)=0.49977010264310245  ;// equal to 0.5
Sin(45)=0.7068251811053661  ;// equal to 0.70; or 1 / square root(2);
Sin(60)=0.8657598394923445 ;// 0.86 ; or square root(3) / 2
Sin(90)=1.0// nearest upper value
```

Discussion: it is a simple program to show the techniques of printing values of different angles of sin, cos, tan etc.; in output it is shown that values are a large decimal number which we need to make rounded for two digits.

Table-1: Trigonometric values for different angles

	0^0	30^0	45^0	60^0	90^0
Sin θ	0	$\frac{1}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{\sqrt{3}}{2}$	1
Cos θ	1	$\frac{\sqrt{3}}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{1}{2}$	0
Tan θ	0	$\frac{1}{\sqrt{3}}$	1	$\sqrt{3}$	∞
Cosec θ	∞	2	$\sqrt{2}$	$\frac{2}{\sqrt{3}}$	1
Sec θ	1	$\frac{2}{\sqrt{3}}$	$\sqrt{2}$	2	∞
Cot θ	∞	$\sqrt{3}$	1	$\frac{1}{\sqrt{3}}$	0

∞ = infinity or not defined.

Problem: wap to prove that $Sin^2x + Cos^2x = 1$, take any degree as 30, 45 etc. as example.

```
//Example-: Trigono2.java: Compute value of Sin(30), Con(30)
//Then do square of both values, do sum
//show or prove that Sin square 30 + Cos square 30 = 1;

import java.lang.*;
class Trigono2
{
        public static void main(String agrs[])
        {
          double r, r1, r2, r3;
          // first convert degree in radians
          //PI=3.14 approximately
          r=Math.sin(30*(3.14/180));
          r1=Math.cos(30*(3.14/180));
          //compute Sin square theta + cos square theta
         // Here it is doing square of previous two values.
          r2=Math.pow(r,2)+ Math.pow(r1,2);
          System.out.println("Sin square 30 + Cos square 30 = " + Math.ceil(r2));
        }
}
```

```
E:\example>java Trigono2
Sin square 30 + Cos square 30 = 1.0
```

Discussion: we know that $Sin^2(x) + Cos^2(x) = 1$, this theory can be proofed by using any degree of x value. First of all, degree should be converted to radians in this example. JAVA uses radian values with sin, cos, tan etc.

JAVA comments:

Two ways comments can be written within the program.

i. // first convert degree in radians
ii. /* first convert degree in radians to use in sin(30) */

These comments can be put as new lines within the program or at end of any statement , after the semicolon .

```
//Example: Math2.java demonstrate log functions.
import java.lang.Math;
class Math2
{
	public static void main(String args[])
	{
		double x=5;
		double y, p, q; /* multiple declaration */
		//calculate values
		y = Math.sqrt(x);
		p=Math.log(100); // log base e value will be output.
		//display values
		System.out.println("Sq root of 5 is = " + y );
		System.out.println("\n Log base e of 100 is  :  " +p);
		System.out.println("\nValue of natural log(100) base 10 is :" +
								Math.log10(100));
	}
}
```

```
E:\example>java Math2
Sq root of 5 is = 2.23606797749979
 Log base e of 100 is:  4.605170185988092
Value of natural log(100) base 10 is :2.0
```

Discussion: function log(x) returns the natural logarithm of base e of a number.
Value of e=2.718 (approximately).
Log10(x): computes value of x when log of base is 10.

Log10(10)=1
Log10(10^^{2}) = 2 , it is same as log10(100).
Math.E=2.718 (approximately)
Math.PI= 3.14159

Some concepts of log mathematics:

$$\log_b a = \frac{\log_d a}{\log_d b}$$

From this formula, we can write:

$\text{Log}_{10}(1000) = \dfrac{\log_{16}(1000)}{\log_{16}(10)}$

Common logarithm is written as $\log(x) = \log_{10}(x)$
Natural logarithm is written as $\ln(x) = \log_e(x)$
Conversion rule:
$\text{Log}_b(x^k) = k * \log_b(x)$

```
// RoomArea.java: two classes-one is for declaration of parameter values and the
//other is calling those values and computes output.
// This program computes area of a square

import java.lang.*;
class Room
{
        float length;
        float breadth;
        void getdata(float a, float b)
        {
                length = a;
                breadth = b;
        }
} //end of class

class RoomArea
{
        public static void main(String agrs[])
        {
                float area;
                Room room1 = new Room();
                room1.getdata(12,10);
                area = room1.length * room1.breadth;
                System.out.println("Area=" + area);
        }
}
```

Run the Program:

```
E:\example>javac RoomArea.java
```

By this command RoomArea.java, a class file, RoomArea.class, will be generated. To run the class file, type the command,

```
E:\example>java RoomArea
```

Note that 'RoomArea' is sufficient to mention, it will take RoomArea.class file to interpret to output as:

```
Area=120.0
```

Some compiler accepts also to type RoomArea.class and will give output . But, in our compiler it is giving error message like:

```
E:\example>java RoomArea.class
Error: Could not find or load main class RoomArea.class
Caused by: java.lang.ClassNotFoundException: RoomArea.class
Correct way to command is:
E:\example>java RoomArea
Area=120.0
```

Translating Formulae in Java:

$(a + b)^2$: Math.pow((a+b), 2); first, a and b are added then power method works.

$\frac{a+b}{2}$: (a + b) /2 ; we can't write as : a + b /2 , then it means a plus half of b.

$\frac{a*b*c}{3}$: a * b * c / 3 ; here parentheses like : (a * b * c) / 3 is not essential, but if we give also result will be same; as per rule of precedence of acting operators- both * and / operators have same precedence and in that case it works from left to right side.

$\sqrt{(a^n + b^n)}$: Math.sqrt((Math.pow(a , n) + Math.pow(b , n)) ; first, a to the power n and b to the power n will be computed, then, two values will be added, last of all outer method Math.sqrt() will be computed.

Summary of the chapter:

After study of this chapter, knowledge of introduction to JAVA programming, some terms related to operating system like: Compiler, Interpreter, Editor, Debugger etc. will be developed. Then some concept will be gained about JAVA platform as-bytecode, JVM, concept of machine independent execution of JAVA program etc. In starting of this section, some features of JAVA are discussed as-multithread based language, different tools of JAVA like- JSP, JPA, Servlet, JDBC etc. are discussed. Finally some programs are written with its execution techniques.

Questions:

1.1.What is a general purpose programming language?
1.2.What is Java Virtual Machine?
1.3.What is Machine Code or Byte Code?
1.4.Mention some method names of Math class.
1.5.How Sin(45) will be converted to radians?

1.6.What are these, JDK, JRE and JVM?
1.7.How comments are put within java program? Is it executed or not?
1.8.What is the role of ‘import’ instruction?

Technical Questions:
The main method is written as **:** public static void main(String args[])
2.1.Why is ‘public’ defined here?
2.2. What is the meaning of ‘static’?
2.3. Why is it declared as ‘void’? can we declare as int or float? Explain
2.4.Why ‘{ }’ braces are required with main()?
2.5.Within main() if we don’t put any arguments- will it work?

Brain storming questions:
3.1.Differences between: i++ and ++i
3.2.Translate this $\sqrt{(a^3+b^3)}$ into formula of java.
3.3.In this “Room room1 = new Room();” what is ‘new’?
3.4. Write relation equation between degree and radian.
3.5.Differentiate between ‘Compiler and Debugger’.
3.6.Is command line argument required, explain your arguments.

Link of PPT of this book and all source codes:

```
https://drive.google.com/drive/folders/1bMxMCaqPe0W35COAdn_-
BrnV_uzQ-tNO?usp=sharing
```

(Note: to get access, copy and paste this link to your browser)