Unit – I. An Introduction to Java

1. History of Java
2. Features of Java
3. Comparison of Java and C++
4. Java Tools And Editors(Appletviewer,Jar,Jdb)
5. Java Environment.

1. What is Java? A short history of Java.

Java is a general purpose object oriented programming language used for developing stand alone applications, distributed applications and web based applications.

Java language developed at SUN (Standard University Network) Microsystems in the year 1995 under the guidance of James Gosling and their team.

James Gosling and his team member given a project name as Green in 1990. The James Gosling team develops a new language called **Oak** but this name is already selected by some other company, there fore Oak is renamed as Java.

Initially Java was designed for the development of software for consumer electronic devices like TV, VCRs and such other electronic machines.

In 1993 the World Wide Web appeared on the internet transformed the text-based internet into a graphics based internet. The Green Project team came up with the idea of developing Web applets using the new language that could run on all types of computers connected to the internet.

In 1994 team develop a Web browser called **"Hot Java"** to locate and run applet programs on internet.

In 1995 Oak was renamed Java.

In 1996 SUN Microsystems released Java Development Kit 1.0 (JDK1.0).

2. Features of Java

Features of a language are nothing but the set of services or facilities provided by the language vendors to the programmers. Some important features of Java language are:

1. Simple, Small and Familiar
2. Object Oriented
3. Compiled and Interpreted
4. Platform independent
5. Architectural neutral
6. Portable
7. Multithreaded
8. Distributed
9. Networked
10. Robust
11. Dynamic
12. Secured
13. High Performance

1. **Simple, Small and Familiar**

Java is simple because of the following factors:

- Java doesn't contain pointer due to this execution time of application improved.
- Java have rich set of API(Application Programming Interface).
- Java has garbage collector which is always used to collect unused memory locations for improving performance of the program.
- Java is familiar language because it contains user friendly syntax for developing Java Application.
- Java uses syntax derived from C and C++. In fact Java is a simplified version of C++.

2. **Object Oriented**

- Java is a pure object oriented programming language. All programs code and data reside within object and classes.
- Java supports all the features of OOPs like inheritance, encapsulation, abstraction and polymorphism.

3. **Compiled and Interpreted**

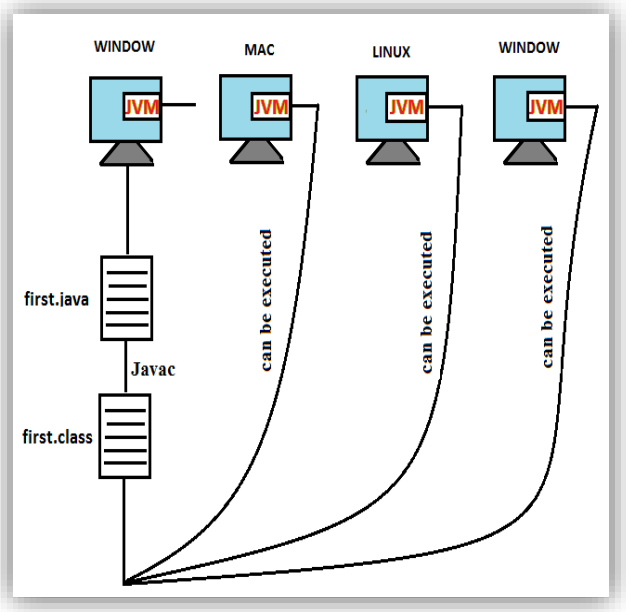
Normally a computer language is either compiled or interpreted. But java is a compiled as well as interpreted language.

First compiler translates source code into byte code instructions. Byte codes are not machine code or machine instructions and therefore in the second stage, Java interpreter translates these byte codes into machine code that can be directly executes by the machine that is running the Java program. Therefore we can say Java is a compiled and interpreted language.

4. **Platform independent**

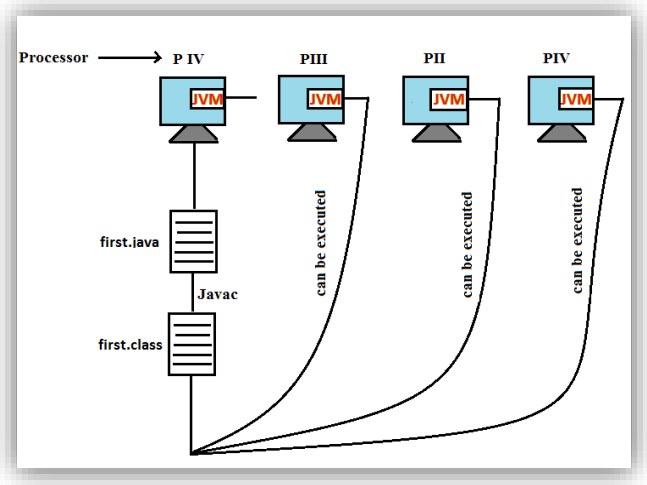
The most important feature of Java is it is a platform independent. A language is said to be platform independent if and only if which can run all available O.S with respect to its development and compilation.

Platform independent means compiled code of any programming language should run on any operating system, for example if we develop a java program on windows operating system then this java program should run on MAC, LINUX, UNIX etc.



5. Architectural neutral

Architecture represents processor. A programming language is said to be architectural neutral which can run on any available processors in the real world without considering their architecture.



6. Portable

The most significant contribution of Java over other languages is its portability. Java programs can be easily moved from one computer system to another computer system anytime and anywhere. Changes and upgrades in operating system, processors and system resources are not force any changes in Java programs. This is the reason why java has become a popular programming language for internet which interconnects different computer systems world wide.

7. Multithreaded

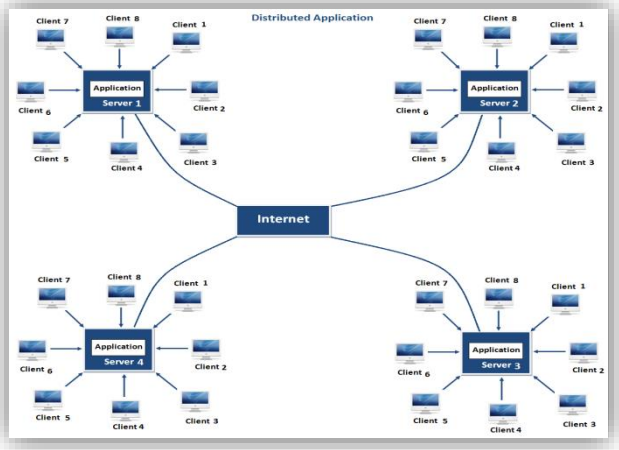
A flow of control is known as a thread. When any language executes multiple threads at a time that language is known as multithreaded language. This means that we need not wait for the applications to finish one task before beginning another.

For example, we can listen an audio clip while scrolling a page and at the same time we can download a file from internet. This feature greatly improves the interactive performance of graphical applications.

8. Distributed

Java is designed as a distributed language for operating applications on networks. We can create distributed applications in Java.

In distributed application multiple client systems are depends on multiple server systems. So that even problem occurred in one server will never be reflected on any client system. In this architecture same application is distributed in multiple server system.



9. Networked

Java is mainly design for web based applications, J2EE is used for developing network based applications.

10. Robust

Simply means of Robust is strong. Java is robust or strong Programming Language because of its capability to handle Run-time Error, automatic garbage collection, lack of pointer concept, Exception Handling. All these points make Java robust Language.

11. Dynamic

Java programming support Dynamic memory allocation due to this memory wastage is reduce and improve performance of application. The process of allocating the memory space to the input of the program at a run-time is known as dynamic memory allocation, In java programming to allocate memory space by dynamically we use an operator called 'new' operator is known as dynamic memory allocation operator.

12. Secure

Java is more secure language as compare to other language. Security becomes important issue for a language as well as applications that is used for the internet. Java system ensure that no viruses are communicated with java applications. The absence of pointer in Java ensure that program can not gain access to memory locations.

13. High Performance

The performance of Java language is high as compare to other object oriented programming language because it uses byte code which is faster than ordinary object code.

Garbage collector collect the unused memory space and improves the performance of the system.

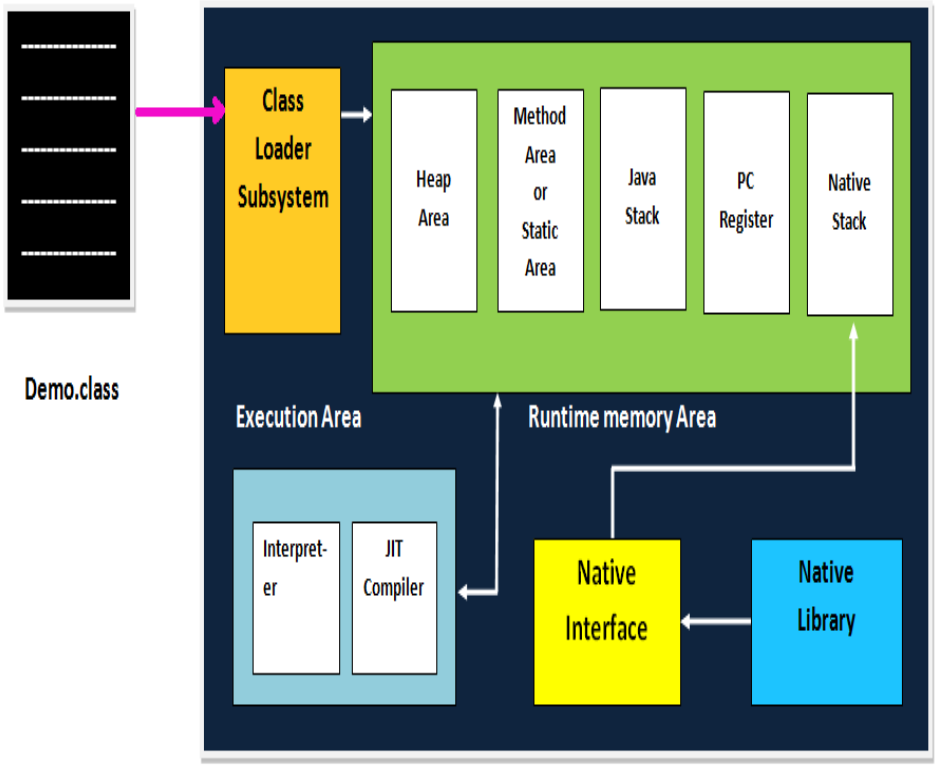
It supports multithreading because of this time consuming process can be reduced to execute the programs.

3. Comparison of Java and C++

- Java is true object oriented programming languages while C++ is basically C with object oriented features or extension.
- Java doesn't support operator overloading.
- Java doesn't support multiple inheritance using classes but this is accomplished by using new concept called "**interfaces**"
- Java doesn't support global variables. Every variable and method declared within a class and forms part of that class.
- Java doesn't use pointers.
- Java has replaced destructor function with finalize() method or function.
- There are no header files in Java instead it uses packages.

4. JVM Architecture

JVM (Java Virtual Machine) is a software. It is a specification that provides runtime environment in which java bytecode can be executed.



Operation of JVM

JVM mainly performs following operations.

1. Allocating sufficient memory space for the class properties.
2. Provides runtime environment in which java bytecode can be executed
3. Converting byte code instruction into machine level instruction.

JVM is separately available for every Operating System by installing java software so that JVM is platform dependent.

Note: Java is platform Independent but JVM is platform dependent because every Operating system have different different JVM which is install along with JDK Software.

Class loader subsystem:

Class loader subsystem will load the .class file into java stack and later sufficient memory will be allocated for all the properties of the java program into following five memory locations.

1. Heap area
2. Method area
3. Java stack
4. PC register
5. Native stack

1. Heap area:

In which object references will be stored.

2. Method area

In which static variables non-static and static method will be stored.

3. Java Stack

In which all the non-static variable of class will be stored and whose address referred by object reference.

4. Pc Register

Which holds the address of next executable instruction that means that use the priority for the method in the execution process?

5. ***Native Stack***

Native stack holds the instruction of native code (other than java code) native stack depends on native library. Native interface will access interface between native stack and native library.

Execution Engine

Which contains Interpreter and JIT compiler whenever any java program is executing at the first time interpreter will comes into picture and it converts one by one byte code instruction into machine level instruction JIT compiler (just in time compiler) will comes into picture from the second time onward if the same java program is executing and it gives the machine level instruction to the process which are available in the buffer memory.

Note: the main aim of JIT compiler is to speed up the execution of java program.

---

5. Java Environment

Java environment includes a large number of development tools and thousands of classes and methods. Development tools are part of the system known as **Java Development Kit(JDK)** and the classes and methods are part of the **Java Standard Library(JSL)** also known as Application Programming Interface.

Java Development Kit

Java Development Kit comes with a collection of tools that are used for developing and running Java programs.

1. Appletviwer (To run java applets)
2. javac (Java Compiler)
3. java (Java Interpreter)
4. javadoc (For creating HTML documents)
5. jdb (Java Debugger)

1. **appletviewer**

   It is used to run Java applets(Without using Java compatible browser)

2. **javac**

   It is a Java compiler, which translates Java source code into bytecode files that the interpreter can understand.

3. **java**

   Java interpreter which runs applications by reading and interpreting bytecode files.

4. **javadoc**

   It is used to creates HTML-format documentation from java source code files.

5. **jdb**

It is a Java debugger, which helps us to find errors in java programs.

Application Programming Interface

The Java standard library(or API) includes hundreds of classes and methods grouped into several functional packages. Most commonly used packages are:

1. **Language Support Package**

   A collection of classes and methods required for basic features of Java.

2. **Utilities Package**

A collection of classes to provide utility functions such as date and time functions.

3. **I/O package**

A collection of classes required for I/O operations.

4. **Networking package**

A collection of classes required for communicating with other computers via internet.

5. **AWT package**

Abstract Window Toolkit package contains classes and methods used to create Graphical User Interface.

6. **Applet Package**

This package contains a set of classes that allows us to create Java applets.

6. A Java programs structure.

A Java program may contain many classes of which only one class defines a main() method. Classes contain data members and methods that operate on the data members of the class. Methods may contain data member declarations and executable statements. A Java program may contains one or more sections as follows.

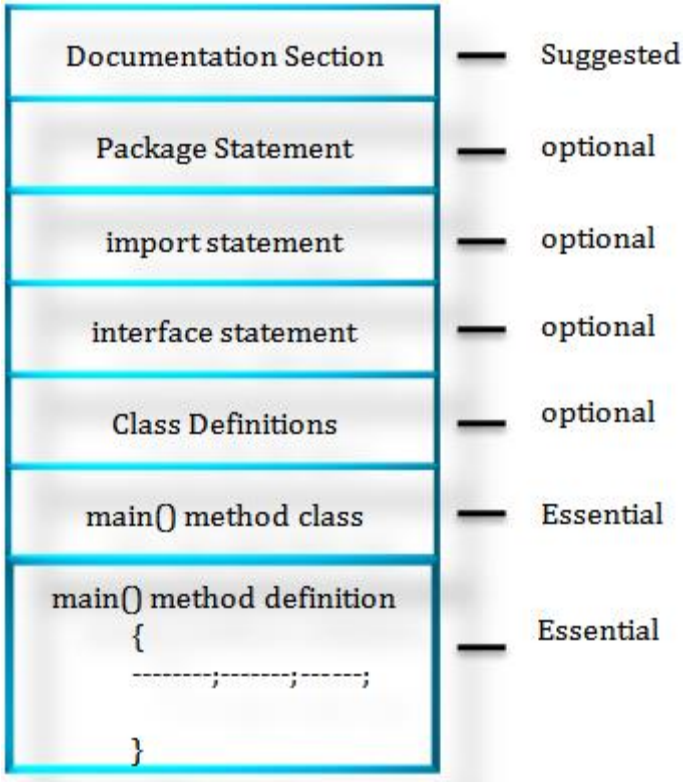


Fig. Java Program Structure

1. **Documentation Section**

The documentation section contains a set of comment lines giving the name of the program, author of the program and other details. Java also uses third style of comment called as documentation comment start with /** --- and ends with ----*/.

2. **Package Statement**

The first executable statement allowed in Java file is a package statement. This statement declares a package name and informs the compiler that the classes defined here belong to this package. Package statement is optional.

E.g. package com.student;

3. Import statement

The next statement after a package statement(but before any class definition) may be a number of import statement. This is similar to #include statement in C.

E.g. import student.test.*;

import java.lang.Math;

Using import statements, we can have access to classes that are part of other named packages.

4. Interface statement

An interface is like a class but includes a group of method declarations. This is also optional section and is used only when we wish to implement the multiple inheritance features in the program.

5. Class Definitions

A Java program may contain multiple class definitions. Classes are the primary and essential elements of a Java program. These classes are used to map the objects of real worlds problems.

6. Main() method class

Every standalone Java program requires a main() method as its starting point, this class is a essential part of the Java program.

7. Main() method definition

The main() method creates objects of various classes and establishes communication between them. On reaching the end of main() method, the program terminates and control passes back to the operating system.

7. Installing and Configuring Java

For executing every Java program we require following things.

1. Install the JDK version.
2. Set path of the jdk/bin directory
3. Create the Java program
4. Compile and run Java Program

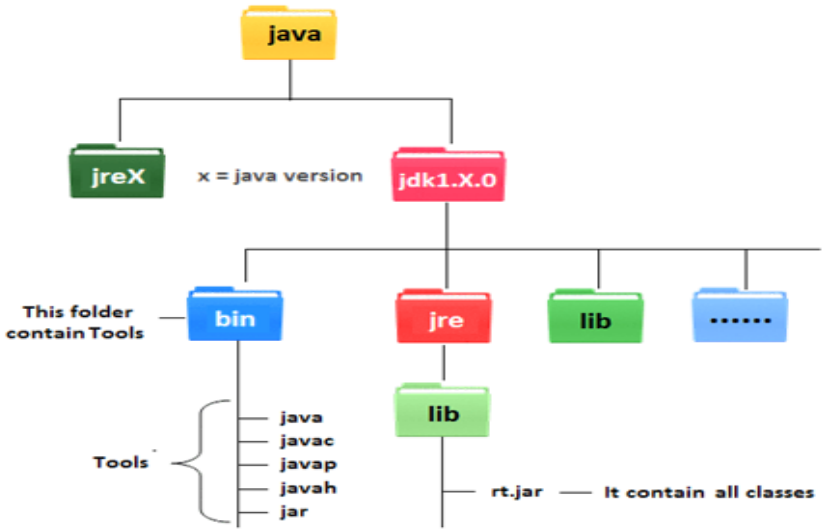
1. Install the JDK version.

Download and install the JDK in C drive.

2. Set path of the jdk/bin directory

Path

Path variable is set for providing path for all java tools like java, javac, javap, javah, jar, appletviewer which are use in java programming. These all tools are available in bin folders so we set path upto bin folders.



Why set path ?

The following programming error is general for all java programmers when they compile any java program.

'javac' is not recognized as an internal or external command, operable program or batch file.

When you get this type of error, then your operating system cannot find the java compiler**(javac)**. To solve this error you need to set the PATH variable.

Javac is a tool which is available in bin folder so you must set the PATH upto bin folder. In a **bin**folder all tools are available like **javap, javah, jar, javac, java, appletviewer** etc. These all tools are used for different-different purpose.

How to set path in Java

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, path is not required to be set because all the tools will be available in the current directory.

But If you are having your java file outside the jdk/bin folder, it is necessary to set path of JDK.

There are 2 ways to set java path:

1. temporary
2. permanent

1. How to set Temporary Path of JDK in Windows

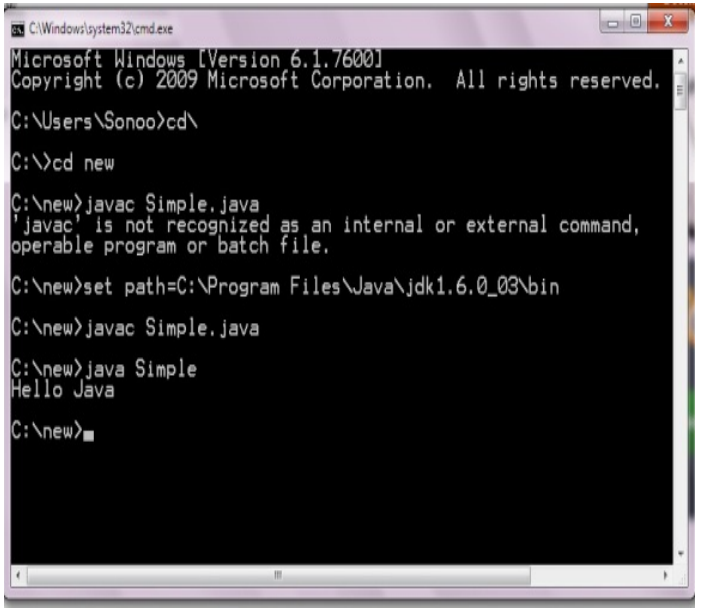
To set the temporary path of JDK, you need to follow following steps:

- ✓ Open command prompt
- ✓ copy the path of jdk/bin directory
- ✓ write in command prompt: set path=copied_path

For Example:

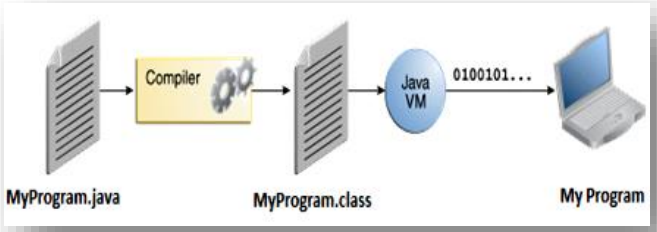
set path=C:\Program Files\Java\jdk1.6.0_23\bin

Let's see it in the figure given below:



3. Create the Java program

In the Java programming language, all source code is first written in plain text files and save with the .java extension. After compilation, .class files are generated by javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes (it is machine language of the Java Virtual Machine1 (JVM)).



The java launcher tool then runs your application with an instance of the Java Virtual Machine (JVM).

Unit-2. An Overview of Java Language

1. Introduction,
2. Types of Comment
3. Java Tokens –
   - Reserve Keywords
   - Identifiers
   - Literals
   - Operators
4. Variable
5. Final variable
6. Data Types
7. Array
8. Type Casting
9. Control Statement - Branching statement - Looping statement

1. Types of Comments.

Java provides three types of comments:

1. Single Line comments

The simplest comment in the Java is single line comment. It starts with two forward slashes and continues to the end of line.

E.g.

```
// This is the single line comment
// int x=10;
```

2. Multiple Line comments

Java also provides a comment that can span multiple lines. This type of comment starts with a forward slash followed by asterisk (/*) and end with an asterisk followed by forward slash.

E.g.

```
/*
This is a
Multiline comment
*/
```

This type of comment is also used for single line comment.

2. Reserved Words:

In Java some identifiers are reserved to associate some functionality or meaning such type of reserved identifiers are called **Reserved Words.**

Keywords for data types:

byte	short	int	char
long	float	double	boolean

Keywords for flow controls:

if	else	Switch	Case
default	for	Do	while
break	continue		

Keywords for modifiers:

public	private	protected	static
final	abstract	synchronized	native
strictfp	transient	volatile	

Keywords for exception handling:

try	catch	finally	throw
throws	assert		

Keywords related with class:

class	static	Import	extends
implements	interface		

Keywords related with objects:

new	instanceof	super	this

Keywords for void return type:

void			

Unused Keywords:

goto	const		

Reserved Literals:

true	false	null	

enum- It is a keyword used to define group of named constant.

3. Identifier:

A name in Java program is called an identifier. It may be class name, method name, variable name or label name.

```
class Test          1
{                   2  3  4
public staic void main(String a[])
{                   5
int count=10;
}
}
```

Rules to define Java Identifiers

1. The only allowed characters in Java identifiers are:

```
a to z, A to Z,
0 to 9, _ , $
```

2. Identifiers are not allowed to starts with digits.

```
abc123// valid
abc_123$// valid
12abc// invalid
```

3. Java language is case sensitive i.e. java identifiers case sensitive
4. There are no length limit of Java identifiers but it is not recommended to use more than 15 length.
5. We can't use reserved words as an identifiers.

4. Variables

In java variable is a location in the memory of computer used to store the values.

Declaration of variables:

The general form of declaring variable is:

```
Syntax1:
Datatype variableName;
Syntax2:
Datatype variable1, variable2, …….. Variablen;
```

```
e.g.
int x;
int x,num,count;
```

Declaration of variables does three things:

1. It tells the compiler what the variable name is.
2. It specifies what type of data the variable will hold.
3. The place of declaration decides the scope of the variables.

Initialization of variables:

A variable must be given a value after it has been declared but before it is used. This can achieved in two ways:

1. By using an assignment statement
2. By using a read statement

1. By using an assignment statement

A simple method of giving a value to a variable is through the assignment statement.

```
Syntax1:
variable_name=value;
E.g.
count=10;
```

It is also possible to assign a value to a variable at the time of its declaration.

```
Syntax1:
datatype variable_name=value;
E.g.
int count=10;
```

Rules to declare variables:

1. Every variable should start with either alphabet or underscore or dollar sign($).
2. No space is allowed in variable declaration.
3. There is no length limit of variable length but it is not recommended to take more than 15 characters.

5. Final Variable in Java

The **final** is a keyword related with a variable, methods and classes.

Final keyword is used to make a variable as a constant. If we declare a variable as a final then we cant change the value of that variable throughout the program.

The following is the syntax of declaring final variable:

```
final datatype variableName=value;
```

E.g.

```
final int x=10;
final float PI=3.14;
```

- If we try to change the value of final variables then we will get compile time error.

  E.g.

  PI=20.5;

  CE: Can not assign a value to final variable PI.

- If the instance variable declared as a final then JVM want provide any default values compulsory we should perform initialization explicitly whether we are using or not otherwise we will get compile time error.

  E.g.

```
class Test
{
final int x;
}
```

  CE: variable I might not have been initialized.

- For the final instance variables compulsory we should perform initialization before constructor completion.

6. Built In Data Types.

Datatype is a special keyword used to allocate sufficient memory space for the data in the primary memory of computer.

In Java there are eight data types which are organized into four groups:

1. Integer Category Data types
2. Character Category Data Types
3. Float Category Data Types
4. Boolean Category Data Types

1. Integer Category Data Types

Integer category data types are used for storing integer data in the main memory (RAM) of computer by allocating sufficient amount of memory.

Integer category data types are divided into four types which are given below:

Data Type	Size	Range
byte	1	-128 to +127
short	2	-32768 to +32767
int	4	-2,147,483,648 to +2,147,483,647
long	8	-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807

2. Character Category Data Types

A character is an identifier which is enclosed within single quotes. In java to represent character we use a data type called "**char**".

Data Type	Size	Range
char	2	65536

This data type takes two bytes space in the memory.

Why Java take two bytes of memory to store character.

Java supports more than 18 international languages so Java takes two bytes for characters. For 18 international languages 1 byte of memory is not sufficient for storing all characters and symbols present in 18 languages. Java supports Unicode but C language supports ASCII code. In ASCII code only English language is present, so for storing all English characters and symbols 1 byte is sufficient.

Unicode character is one which contains all the characters which are available in 18 international languages and it contains **65536 characters.**

3. Float Category Data Type

Float category data types are used for representing floating point values. This category contains two data types:

Data Type	Size	Range
float	4	-2,147,483,648 to +2,147,483,647
Double	8	- 9.223*1018

4. Boolean Category Data type

Boolean category data type is used to representing or storing logical values i.e. true and false. To represent boolean values we use a data type called boolean.

Boolean data types takes zero(0) byte of memory space because boolean datatype of java implemented by SUN Microsystems with a concept of flip-flop. A flip-flop is a general purpose register which stores one bit of information.

Types of variables:

Based on the type of value represented by variable all variables are divided into two types:

1. **Primitive type variables**

It can be used to represent primitive values. E.g. int x=10;

2. **Reference type variables**

It can be used to represent objects.

Student s1=new Student();

Based on the purpose and the position of declaration all variables are divided into three types:

1. **Instance variables**
2. **Static variables**
3. **Local variables**

1. Instance variables

A variable that is declared inside the class but outside the methods and blocks is called instance variables. It is not declared as static.

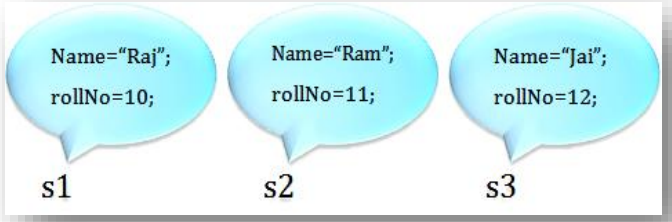
If the variable value is varied from object to object then we should go for instance variables.

For every object a separate value of instance variable is created.

E.g.

```
class Student
{
        String name;
        int rollNo;
}
```

```
Student s1=new Student();
Student s2=new Student();
Student s3=new Student();
```



Instance variables will be created at the time of object creation and destroyed at the time of object destruction. Hence the scope of the instance variable is exactly same as scope of objects.

Instance variables can not be accessed directly from static area, we should use object reference but from instance area we can access instance variables directly.

For the instance variables it is not required to perform initialization explicitly JVM will always provides default values.

2. Static variables

- A variable that is declared as a static is called static variables. It can not be local.
- Static variable should declare at the class level by using static modifier.

- If the value of a variable is not varied from object to object such type of variables we should declare as a static.

  E.g.

```
class Test
{
  static collegeName="MGM";
}
```

- In the case of instance variables for every object a separate copy will be created but in the case of static variables at class level a single copy will be created and shared that copy by all objects of that class.
- We can access static variables directly from both the instance variables and static block or area.
- We can access static variables by using object reference or by using class name but usage of class name is recommended.
- We should declare static variables within the class directly but outside of any methods or blocks or constructors.

3. Local variables

- A variable that is declared inside the methods or blocks or constructor is called local variables.

```
class Student
{
  Student()
  {
   int count=10;
  }
```

```
void show()
{
  int x=5;
}
}
```

- Sometimes to meet temporary requirement of the programmer we should use local variables.
- Local variables are also known as temporary variables.
- Local variable will be created while executing the block in which we declare it. After completing the block execution automatically local variables will be destroyed. Hence the scope of the local variables is exactly same as scope of the block in which we declare it.
- For local variables JVM doesn't provide any default values compulsory we should perform initialization explicitly before using that variable.
- It is highly recommended to perform initialization for the local variables at the time of declaration at least with default values.
- The only applicable modifiers for the local variable is final, if we are using any other modifier we will get compile time error.

7. Memory allocation using new operator

It means creating a new object of a particular class by allocating required memory space to store all the instance variables of that class. Creating an object is also known as instantiating an object.

Objects in Java are created using the "**new**" operator. The new operator creates an object of a specified class and returns the reference to that object.

More technical definition of new operator is it is used to transfer the contents of .class file from secondary memory(hard disk) into primary memory(RAM).

The following is the syntax of creating object.

```
ClassName objectName;
objectName=new ClassName();
```

We can combine both above statement into a single statement.

```
ClassName objectName=new ClassName();
```

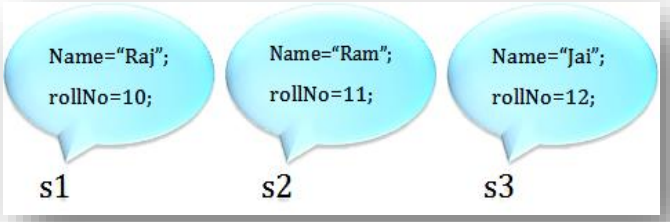
E.g.

```
1. Rectangle rect
rect=new Rectangle();
2. Student s1=new Student();
3  String s2=new String();
```

It is important to understand that each object has its own copy of the instance variables of its class. This means that any changes to the variables of one object have no effect on the variable of another object.

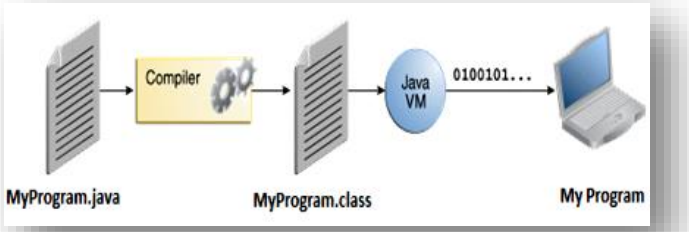
E.g.

```
Student s1=new Student();
Student s2=new Student();
Student s3=new Student();
```

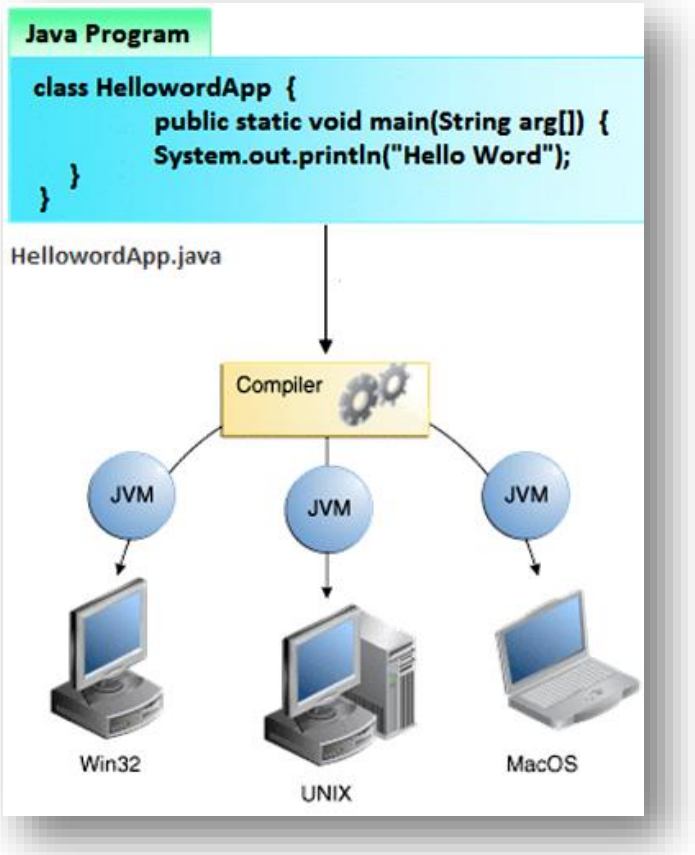


4. Create the Java program

In the Java programming language, all source code is first written in plain text files and save with the .java extension. After compilation, .class files are generated by javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes (it is machine language of the Java Virtual Machine1 (JVM)).



The java launcher tool then runs your application with an instance of the Java Virtual Machine (JVM).



Steps For compile Java Program

First Save Java program with same as class name with .java extension.

Example: Sum.java

Compile: javac Filename.java

Example, javac Sum.java

Note: Here javac is tools or application programs or exe files which is used for Compile the Java program.

Steps For Run Java Program

For run java program use java tool.

Run by: java Filename

Example: java sum

Note: Here java is tools or application programs or exe files which is used for run the Java program.

During the program execution internally following steps will be occurs.

1. Class loader subsystem loads or transfer the specified class into main memory(RAM) from secondary memory(hard disk).
2. JVM takes the loaded class.
3. JVM looks for main method because each and every java program start executing from main() method.
4. Since main() method of java is static in nature, JVM call the main() method with respect to loaded class (Example: First as First.main(--))

Note: A java program can contain any number of main method but JVM start execution from that main() method which is taking array of object of String class.

8. Control Flow Statements

The **control flow statements** are used to control the flow of execution of the program. In java programming language there are three types of control flow statement available.

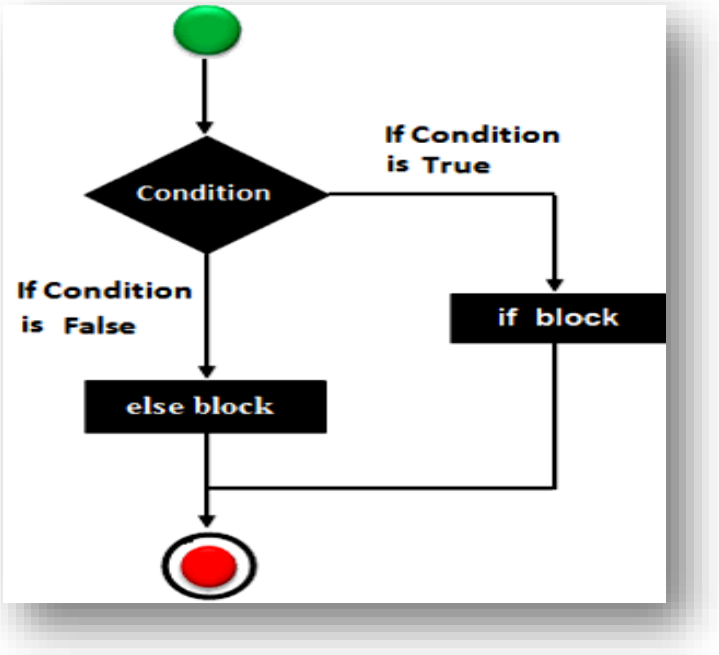
1. Decision making or selection statement (if, if-else, switch)
2. Looping or Iteration statement (while, for, do-while)
3. Breaking or Jumping statement (break, continue)

1. Decision Making Statements

Decision making statement statements is also called selection statement. That is depending on the condition block need to be executed or not while is decided by condition. If the condition is "true" statement block will be executed, if condition is "false" then statement block will not be executed.

In java there are three types of decision making statement.

1. if
2. if-else
3. switch



1. if-then Statement

if-then is most basic statement of Decision making statement. It tells to program to execute a certain part of code only if particular condition or test is true.

Syntax:

```
if(condition)
{
 Statement(s)
}
```

- Constructing the body if always optional, that is recommended to create the body when we are having multiple statements.
- For a single statement, it is not required to specify the body.
- If the body is not specified, then automatically condition part will be terminated with next semicolon (;).

2. IF-Else

- It is a keyword, by using this keyword we can create a alternative block for "if" part.
- Using else is always optional i.e, it is recommended to use when we are having alternate block of condition.
- When we are working with if else among those two block at given any point of time only one block will be executed.
- When if condition is false then else part will be executed, if part is executed then automatically else part will be ignored.

if-else statement

In general it can be used to execute one block of statement among two blocks, in java language if and else are the keyword in java.

```
if(condition)
 {
  Statement(s)
 }
 else
 {
  Statement(s)
 }
Statement(s)
```

In the above syntax whenever condition is true all the if block statement are executed remaining statement of the program by neglecting else block statement If the condition is false else block statement remaining statement of the program are executed by neglecting if block statements.

A java program to find the addition of two number if first number Is greater than second number Otherwise find the subtraction of two numbers.

Example

```
import java.util.*;
class num
```

```
{
 public static void main(String args[])
 {
  int  c;
  System.out.println("Enter any two num");
  Scanner sn=new Scanner(System.in);
  int a=sn.next();
  int b=sn.next();
  if(a>b)
  {
   c=a+b;
  }
  else
  {
   c=a-b;
  }
  System.out.println("Result="+c);
 }
}
```

3. Switch Statement

A switch statement work with byte, short, char and int primitive data type, it also works with enumerated types and string.

```
switch(expression/variable)
{
 case  value:
 //statements
 // any number of case statements
 break;  //optional
 default: //optional
 //statements
}
```

Rules for apply switch statement

With switch statement use only byte, short, int, char data type. You can use any number of case statements within a switch. Value for a case must be same as the variable in switch .

Limitations of switch statement

Logical operators cannot be used with switch statement. For instance

k>=0 is not allowed

Switch case variables can have only int and char data type. So float data type is not allowed.

2. Looping statement

These are the statements execute one or more statement repeatedly several number of times. In java programming language there are three types of loops are available, that is, while, for and do-while.

Advantage with looping statement

- Length on the developer is reducing.
- Burden on the developer is reducing.
- Burden of memory space is reduced time consuming process to execute the program is reduced.

Difference between conditional and looping statement

Conditional statement executes only once in the program where as looping statements executes repeatedly several number of time.

1. While loop

When we are working with while loop always pre-checking process will be occurred.

Pre-checking process means before evolution of statement block condition part will be executed.

While loop will be repeats in clock wise direction.

```
while(condition)
{
 Statement(s)
 Increment / decrements (++ or --);
}
```

Example

```
class  whileDemo
{
 public static void main(String args[])
 {
 int i=0;
 while(i<10)
 {
  System.out.println(+i);
  i++;
 }
}
}
```

Output

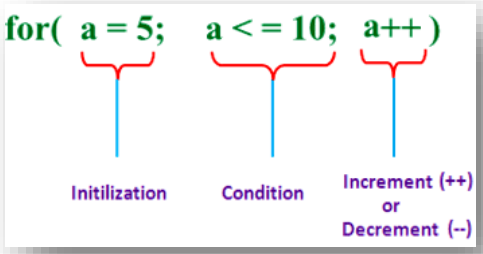
1
2
3
4
5
6
7
8
9
10

2. for loop

for loop is a statement which allows code to be repeatedly executed. For loop contains 3 parts.

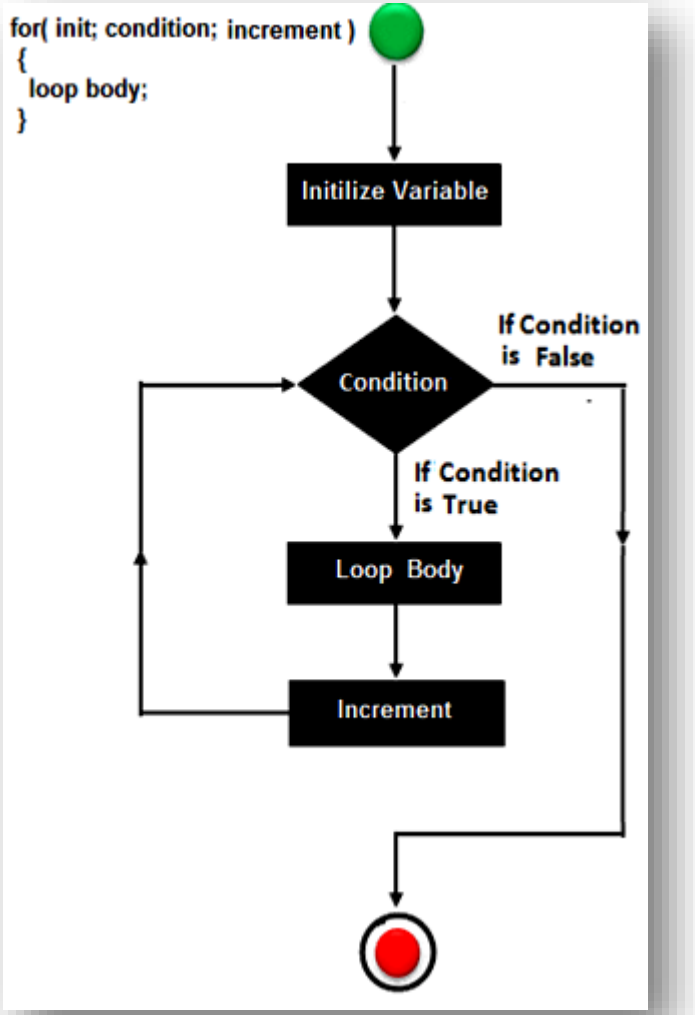
- Initialization
- Condition
- Increment or Decrements

```
for ( initialization;  condition; increment )
{
 statement(s);
}
```



- **Initialization:** step is execute first and this is execute only once when we are entering into the loop first time. This step is allow to declare and initialize any loop control variables.
- **Condition:** is next step after initialization step, if it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and flow of control goes outside the for loop.
- **Increment or Decrements:** After completion of Initialization and Condition steps loop body code is executed and then Increment or Decrements steps is execute. This statement allows to update any loop control variables.

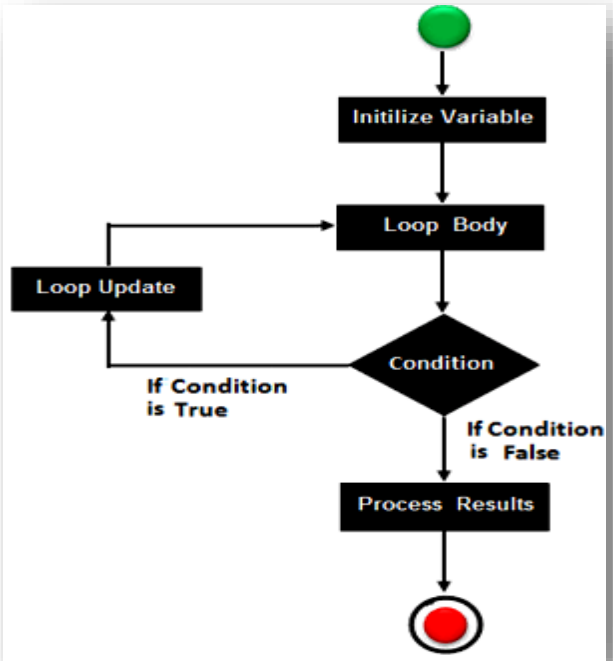
Flow Diagram



Control flow of for loop

```
        1       2       4
for ( a = 1;    a < = 5;    a ++ )
{
  System.out.println("Hello World")    3
}
```

- ✓ First initialize the variable
- ✓ In second step check condition
- ✓ In third step control goes inside loop body and execute.
- ✓ At last increase the value of variable
- ✓ Same process is repeat until condition not false.



3. **do-while**
   - ➢ when we need to repeat the statement block at least 1 then go for do-while.
   - ➢ In do-while loop post-checking process will be occur, that is after execution of the statement block condition part will be executed.

```
do
{
Statement(s)
increment/decrement (++ or --)
}while();
```

Example

```
class  dowhileDemo
{
 public static void main(String args[])
 {
 int i=0;
do
 {
System.out.println(+i);
 i++;
}
while(i<10);
}
}
```

Output

1
2
3
4
5
6
7
8
9
10