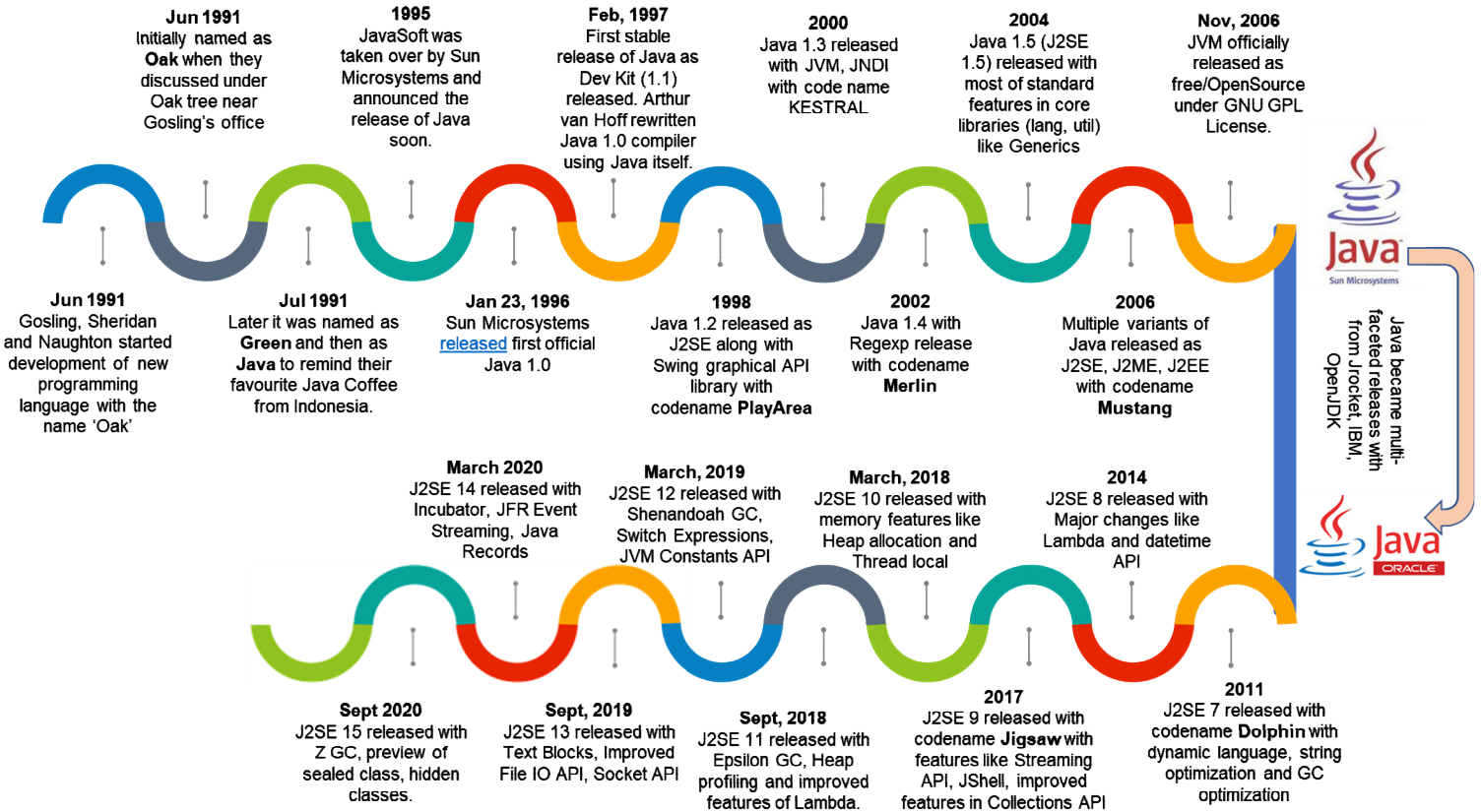
Department of
COMPUTER SCIENCE
AND ENGINEERING

PRASAD V. POTLURI SIDDHARTHA INSTITUTE OF TECHNOLOGY
VIJAYAWADA

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in
Professor, Department of CSE,
Prasad V Potluri Siddhartha Institute of Technology
Dr. P Sai Kiran

JAVA HISTORY



57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

DESIGN FEATURES OF JAVA

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

FEATURES OF JAVA

Simple

- Simple to learn and use.
- Java syntax is based on C++.
- Java has removed many complicated and rarely-used features, for example, explicit pointers, operator overloading, etc.
- There is no need to remove unreferenced objects because there is an Automatic Garbage Collection in Java.
- Another aspect of being simple is being small.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

FEATURES OF JAVA

Object-oriented

Java is an object-oriented programming language.

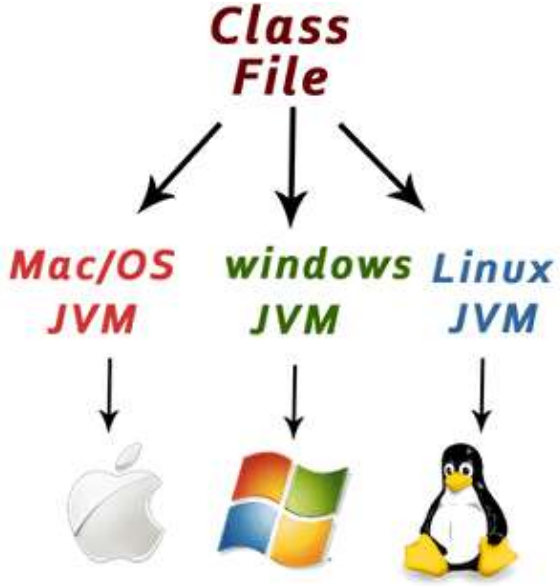
- Everything in Java is an object.
- No free functions.
- All code belong to some class.
- Classes are in turn arranged in a hierarchy or package structure.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

FEATURES OF JAVA

Platform Independent

- Languages like C, C++, etc. which are compiled into platform specific machines.
- Java is Write Once and Run Anywhere (WORA).
- Java code is compiled by the compiler and converted into bytecode.
- Java Virtual Machine executes the bytecode.

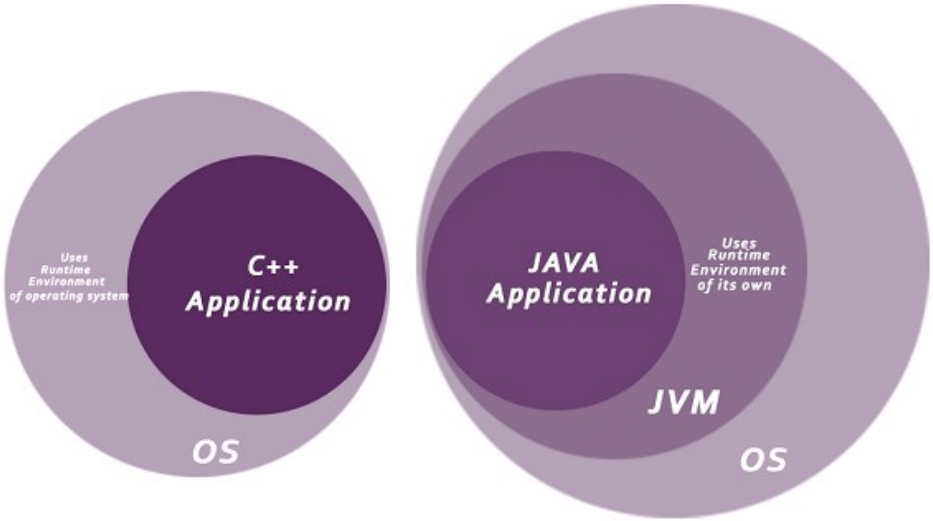


FEATURES OF JAVA

Secured

Java is best known for its security.

- No explicit pointer
- Java Programs run inside a virtual machine sandbox
- Classloader
- Bytecode Verifier
- Security Manager



57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

FEATURES OF JAVA

Robust

- It uses strong memory management.
- There is a lack of pointers that avoids security problems.
- Java provides **automatic garbage collection** which runs on the Java Virtual Machine to get rid of objects which are not being used by a Java application anymore.
- There are **exception handling** and the type checking mechanism in Java.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

FEATURES OF JAVA

Architecture-neutral

Java is architecture neutral because there are no implementation dependent features, for example, the size of primitive types is fixed.

Portable

Java is portable because it facilitates you to carry the Java bytecode to any platform. It doesn't require any implementation.

Multi-threaded

We can write Java programs that deal with many tasks at once by defining multiple threads.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

FEATURES OF JAVA

Dynamic

Java is a dynamic language. It supports the dynamic loading of classes. It means classes are loaded on demand.

Interpreted

- The program are compiled into Java Virtual Machine (JVM) code called bytecode
- Each bytecode instruction is translated into machine code at the time of execution

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

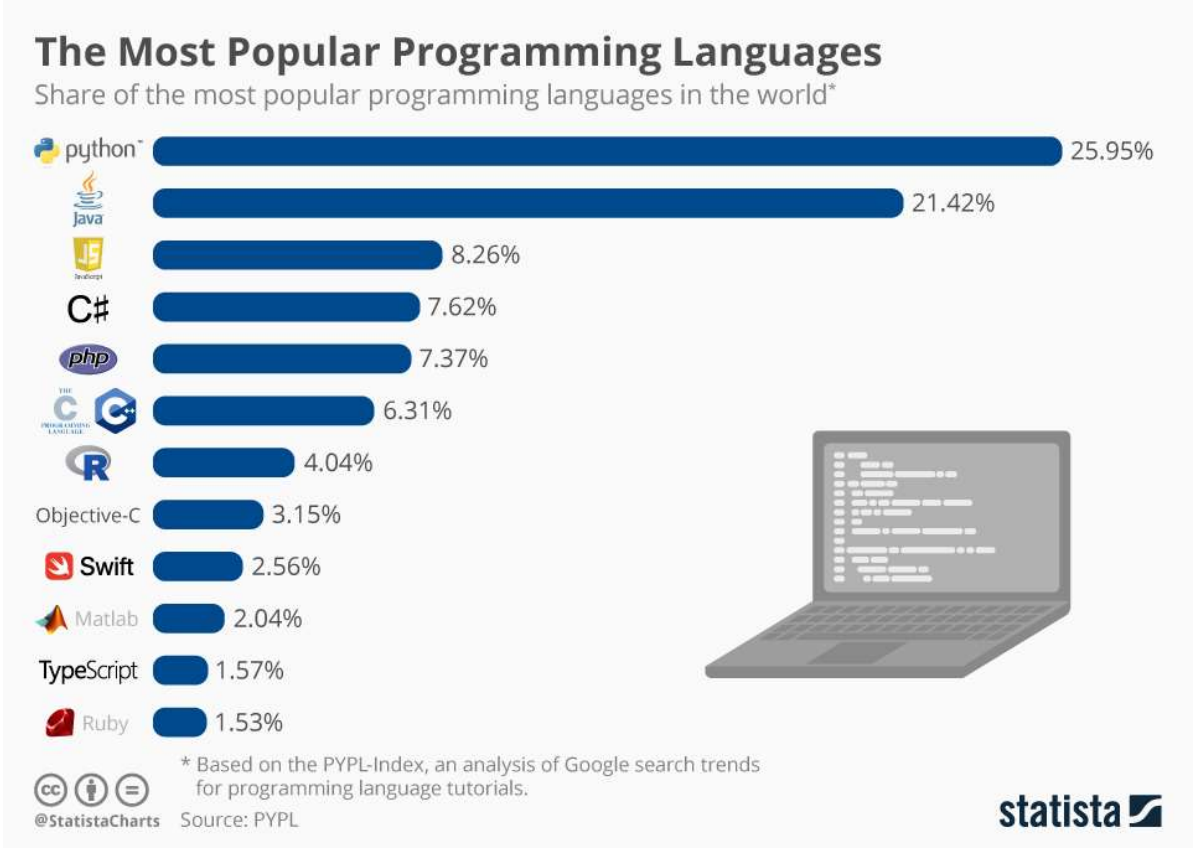
FEATURES OF JAVA

Distributed

- Java is distributed because it facilitates users to create distributed applications in Java.
- Fully supports IPv4, with structures to support IPv6
- Includes support for Applets: small programs embedded in HTML documents
- RMI and EJB are used for creating distributed applications.
- This feature of Java makes us able to access files by calling the methods from any machine on the internet.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

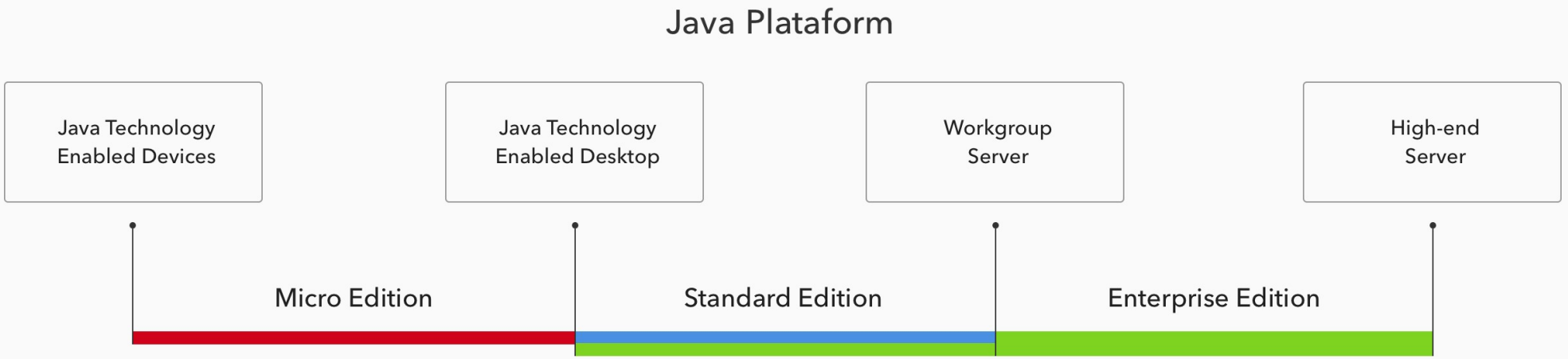
POPULARITY OF JAVA



Rank	Language	Type	Score
1	Python		100.0
2	Java		95.3
3	C		94.6
4	C++		87.0
5	JavaScript		79.5
6	R		78.6
7	Arduino		73.2
8	Go		73.1
9	Swift		70.5
10	Matlab		68.4

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in
Professor, Department of CSE,
Prasad V Potluri Siddhartha Institute of Technology
Dr. P Sai Kiran

JAVA PLATFORMS



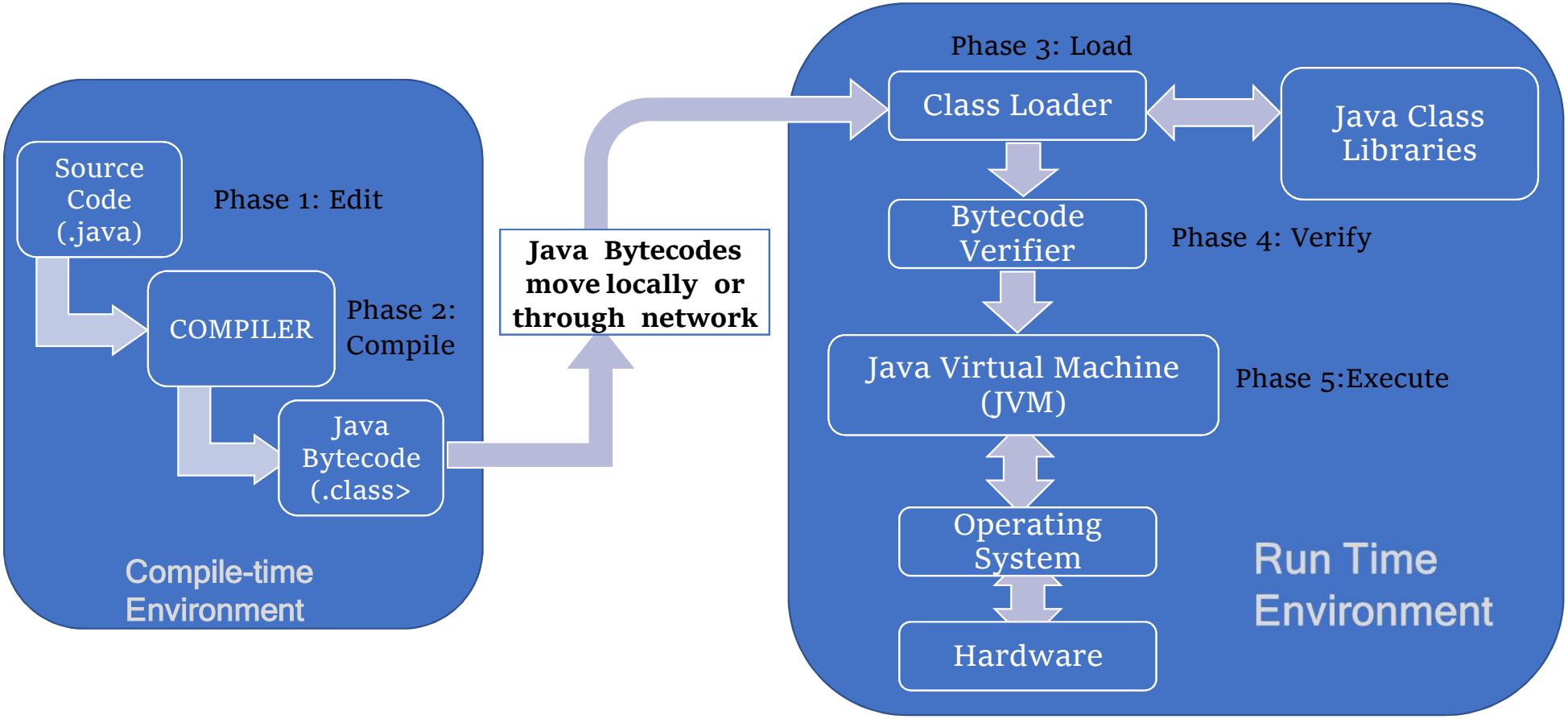
Used for developing applications for small memory-constrained devices, such as cell phones, pagers and PDAs

Used for developing Desktop based application and networking applications

Used for developing large-scale, distributed networking applications and Web-based applications

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

JAVA ARCHITECTURE



57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Quiz

1. Write the correct order of the Java program execution
   A. Class Loader
   B. Interpretation
   C. Compilation
   D. Byte Code Verification
   E. Java Source Code
   F. Execution

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Quiz

2. Which of the following is used to load a .class file?

   A. Class Loader

   B. Byte Code Verifier

   C. JIT Compiler

   D. Interpreter

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Quiz

3. When a java program is compiled, it creates a

   A. an obj file

   B. an exe file

   C. a .class file

   D. a .sh file

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

First Program


```
class FirstProgram
 {
    public static void main(String args[])
      {
        System.out.println("This is my First Java Program");
      }
 }
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Understand First Program

- A Java source file can contain multiple classes, but only one class can be a public class.
- The source file name must match the name of the public class defined in the file with the .java extension.
- A public class is accessible across packages.
- Body of every member function of a class (called method in Java) must be written when the method is declared.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Understand First Program

public static void main(String[] args)

- **main** is the starting point of every Java application
- **public** is used to make the method accessible by all
- **static** is used to make main a static method of class.
- Static methods can be called by JVM without using any object; just using the class name.
- **void** means main does not return anything
- **String args[ ]** represents an array of String objects that holds the command line arguments passed to the application.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Understand First Program

System.out.println()

- Used to print a line of text followed by a new line
- **System** is a class inside the Java API
- **out** is a public static member of class System
- **out** represents the standard output (similar to stdout or cout)
- **println** is a public method of the class of which out is an object

We can use the **plus operator (+)** to concatenate multiple String objects and create a new String object.

How to Execute a Java Program

1. Using Java Online Compiler
2. Using JDK and Notepad
3. Using Editors and JRE

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

PATH

- PATH is an *environmental variable* in DOS(Disk Operating System), Windows and other operating systems like Unix.
- PATH tells the operating system which directories(folders) to search for executable files, in response to commands issued by a user .
- It is a convenient way of executing files without bothering about providing the absolute path to the folder, where the file is located.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

CLASSPATH

- CLASSPATH is a parameter which tells the JVM or the Compiler, where to locate classes that are not part of Java Development ToolKit(JDK).
- CLASSPATH is set either on command-line or through environment variable.
- CLASSPATH set on command-line is temporary in nature, while the environment variable is permanent.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Naming Conventions

Class Names

- Class names should be **nouns**, in mixed case with the first letter of each internal word capitalized
- Class names should be simple and descriptive
- Eg: class **Student**, class **TestStudent**

Method Names

- Methods should be **verbs**, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized
- Eg: void run(), void getColor()

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me

Sample.java

```
class A {
    void m1() { }
    }
class B {
    void m2() { }
    }
class C {
    void m3() { }
   }
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class A {
    void m1() { }
    }
public class B {
    void m2() { }
    }
class C {
    void m3() { }
   }
```

What should be the Source File Name

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me

Sample.java

```
class Sample {
    public static void main() {
        System.out.println("Welcome");
    }
}
```

- Compilation Error
- Runtime Error
- The program compiles and executes successfully but prints nothing.
- It will print "Welcome"

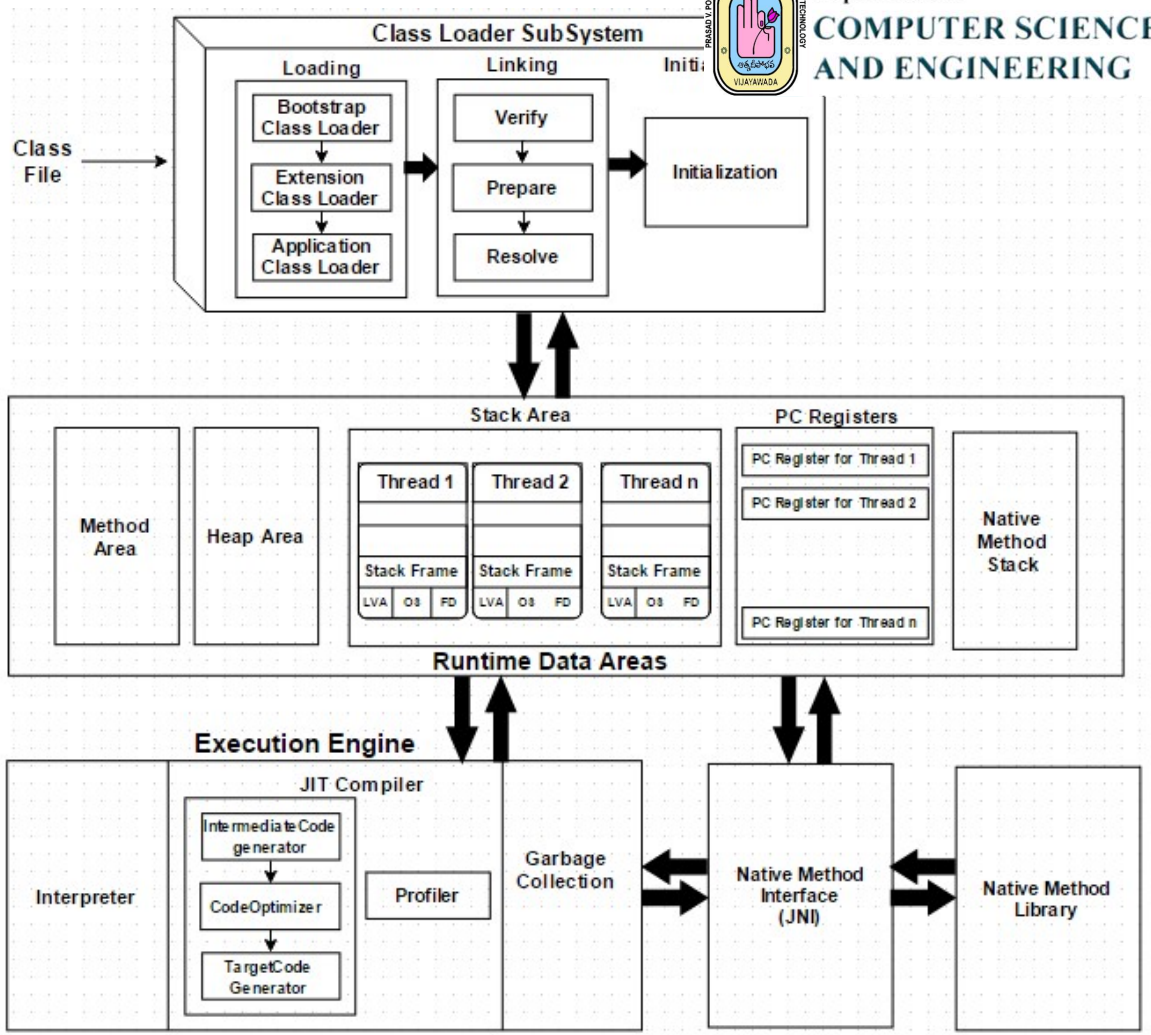
JVM Architecture

There are mainly three sub systems in the JVM as shown in the above diagram,

1.ClassLoader

2.Runtime Memory/Data Areas

3.Execution Engine



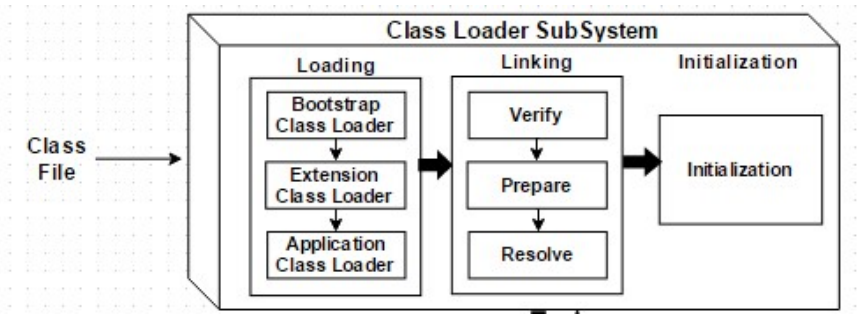
Department of COMPUTER SCIENCE AND ENGINEERING

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in

Professor, Department of CSE,
Prasad V Potluri Siddhartha Institute of Technology

Dr. P Sai Kiran

JVM Classloader



Bootstrap ClassLoader: It loads the rt.jar file which contains all class files of Java Standard Edition.

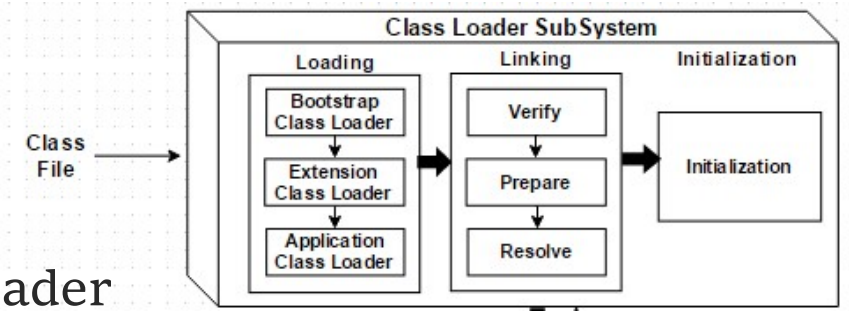
Extension ClassLoader: It loads the jar files located inside *$JAVA_HOME/jre/lib/ext* directory.

System/Application ClassLoader: It loads the classfiles from classpath.

You can change the classpath using "-cp" or "-classpath" switch.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

JVM Classloader

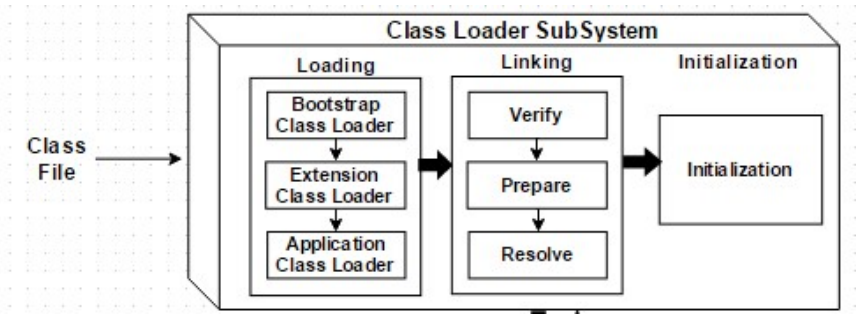


The four main principles in JVM, Class Loader

1. *Visibility Principle:* ClassLoader of a child can see the class loaded by Parent but not vice versa.
2. *Uniqueness Principle*: A class loaded by the parent ClassLoader shouldn't be loaded by the child again.
3. *Delegation Hierarchy Principle:* JVM follows a hierarchy of delegation to choose the class loader for each class loading request.
4. *No Unloading Principle*: class cannot be unloaded by the Classloader

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

JVM Classloader



Linking

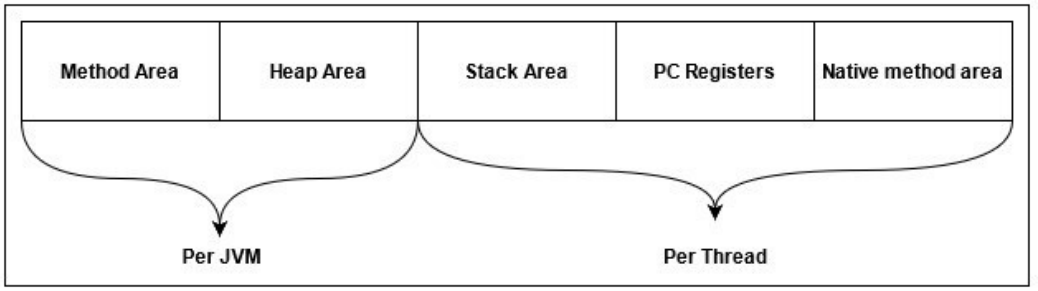
1. *Verification:* Whether it is coming from a valid compiler or not and code has a correct structure and format.

2. *Preparation:* Static variables memory will be allocated and assigned with default values based on the data types.

3. *Resolution:* JVM will assign memory location for those objects by replacing their symbolic links with direct links.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

JVM Memory Areas



Class(Method) Area

Class(Method) Area stores per-class structures such as the runtime constant pool, field and method data, the code for methods.

Heap

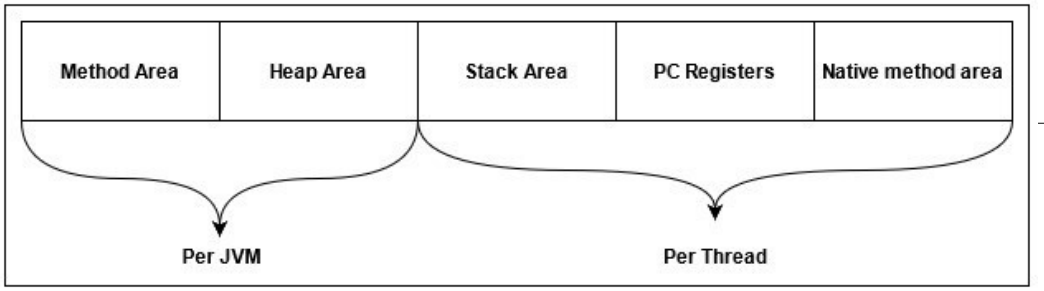
It is the runtime data area in which objects are allocated.

Stack

Java Stack stores frames. It holds local variables and partial results and plays a part in method invocation and return.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

JVM Memory Areas



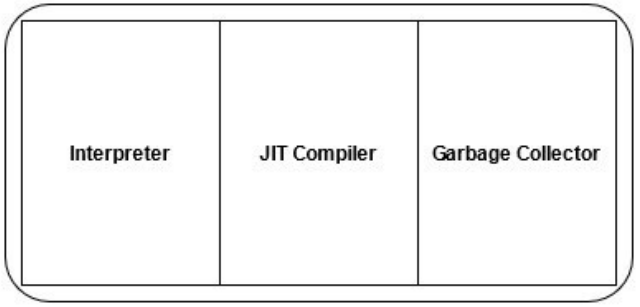
Program Counter Register

PC (program counter) register contains the address of the Java virtual machine instruction currently being executed.

Native Method Stack

It contains all the native methods used in the application.

JVM Execution



Execution Engine:

1.*Interpreter*: Read bytecode stream then execute the instructions.

2. *Just-In-Time(JIT) compiler*: It is used to improve the performance.

3.*Garbage Collector*: Mark and Sweep Phases.

Java Native Interface

Java Native Interface (JNI) is a framework which provides an interface to communicate with another application written in another language like C, C++, Assembly etc.

Command line arguments

While executing a java program, command line arguments can be passed by

java Simple <argument1> <argument2>....<argument-n>

args[0] args[1] args[2]

You can access these arguments in your program, using the String array that you have passed as an argument to the main method.

String[] args

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Command line arguments


```
class Argument {
  public static void main(String[] args) {
    System.out.println(args[0]);
  }
}
```

*If we run **java Argument Welcome***
What will be the output

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a Program to accept a String as a Command line argument and the program should print a Welcome message.

Example :

C:\> java Message John

O/P Expected : Welcome John

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Command line arguments


```
class Arguments {
  public static void main(String[] args) {
    System.out.println(args[0]);
    System.out.println(args[1]);
  }
}
```

*If we run **java Arguments Welcome Sai***
What will be the output

Do It Yourself

Write a Program that accepts two Strings as command line arguments and generate the output in a specific way as given below.

Example:

If the two command line arguments are ***Wipro*** and ***Bangalore*** then the output generated should be ***Wipro Technologies, Bangalore, India.***

If the command line arguments are **ABC** and Mumbai then the output generated should be ***ABC Technologies, Mumbai, India***

[Note: It is mandatory to pass two arguments in command line]

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Finding length of an Array

To find the number of command line arguments that a user may pass while executing a java program.

args.length

where args is the String array that we pass to the main method and length is the property of the Array Object

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class FindLength{
    public static void main(String[ ] args) {
        int len = args.length;
        System.out.println(len);
    }
}
```

What will be the output of following Executions

java FindLength 1 2 3 4 5 6 7

java FindLength Tom John Lee

SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Primitive Data Types

Data Type	Size (in bits)	Minimum Range	Maximum Range	Default Value (for fields)
byte	8	-128	+127	0
short	16	-32768	+32767	0
int	32	-2147483648	+2147483647	0
long	64	-9223372036854775808	+9223372036854775807	0L
float	32	1.40E-45	3.402823466385288 60e+38	0.0f
double	64	4.94065645841246544e-324d	1.79769313486231570e+308d	0.0d
char	16		0 to 65,535	'\u0000'
boolean	1	NA	NA	false

SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me

What will be the result, if we try to compile and execute the following code?

```
class Test {
    public static void main(String args[]) {
        byte b=128;
        System.out.println(b);
    }
}
```

Try it and Tell me

What will be the result, if we try to compile and execute the following code?

```
class Test {
    public static void main(String ar[]) {
        double f=1.2;
        boolean b=1;
        System.out.println(f);
        System.out.println(b);
    }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class Test {
    public static void main(String ar[]) {
        float f=1;
        float b=1.2;
        float c=1.2323f;
        System.out.println(f);
        System.out.println(b);
        System.out.println(c);
    }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class FloatExample
{
  public static void main(String args[])
   {
     float d=987654321.1234567f;
     float c=6.123456789f;
     System.out.println(d);
     System.out.println(c);
   }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class Test {

    public static void main(String [ ] ar) {

        int a=10,b=017,c=0X3A;

         System.out.println(a+","+b+","+c);

    }

}
```

Basically if an assigned integer value begins with 0,it is considered to be an octal value.
If the assigned integer value begins with 0x it is taken as hexadecimal value.

SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Type Casting

Type casting is when you assign a value of one primitive data type to another type.

Widening Casting (automatically) - converting a smaller type to a larger type size

byte -> short -> char -> int -> long -> float -> double

Narrowing Casting (manually) - converting a larger type to a smaller size type

double -> float -> long -> int -> char -> short -> byte

SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class WideType {
  public static void main(String[] args) {
    int myInt = 9;
    double myDouble = myInt;
    System.out.println(myInt);
    System.out.println(myDouble);
  }
}
```

SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class NarrowCast{
  public static void main(String[] args) {
    double myDouble = 9.78d;
    int myInt = (int) myDouble;
    System.out.println(myDouble);
    System.out.println(myInt);
  }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

WRAPPER CLASSES

- For all the primitive data types available in Java, there is a corresponding Object representation available which is known as Wrapper Classes
- **Need for Wrapper Classes**
- All Collection classes in Java can store only Objects
- Primitive data types cannot be stored directly in these classes and hence the primitive values needs to be converted to objects
- We have to wrap the primitive data types in a corresponding object, and give them an object representation

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

WRAPPER CLASSES

- Definition: The process of converting the primitive data types into objects is called **wrapping**
- To declare an integer 'i' holding the value 10, you write

```
int i = 10;
```

- The object representation of integer 'i' holding the value 10 will be:

```
Integer iref = new Integer(i);
```

- Here, class Integer is the wrapper class wrapping a primitive data type i

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

WRAPPER CLASSES

- For all the primitive data types, there are corresponding wrapper classes.
- Representing an integer via a wrapper takes about 12-16 bytes, compared to 4 in an actual integer. Also, retrieving the value of an integer uses the method Integer.intValue().
- For example, you can take the integer input from the user in the form of a String, and convert it into integer type using the following statements:

```
String str = “100”;

int j = Integer.parseInt(str);
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

WRAPPER CLASSES

- Class **Integer** is a wrapper for values of type int
- The **constructors** for Integer are shown here:
  - Integer(int num)
  - Integer(String str) throws NumberFormatException
- Some **methods** of the Integer class:
  - static int parseInt(String str) throws NumberFormatException
  - int intValue( ) returns the value of the invoking object as a int value

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

WRAPPER CLASSES

- Class **Integer** is a wrapper for values of type int
- The **constructors** for Integer are shown here:
  - Integer(int num)
  - Integer(String str) throws NumberFormatException
- Some **methods** of the Integer class:
  - static int parseInt(String str) throws NumberFormatException
  - int intValue( ) returns the value of the invoking object as a int value

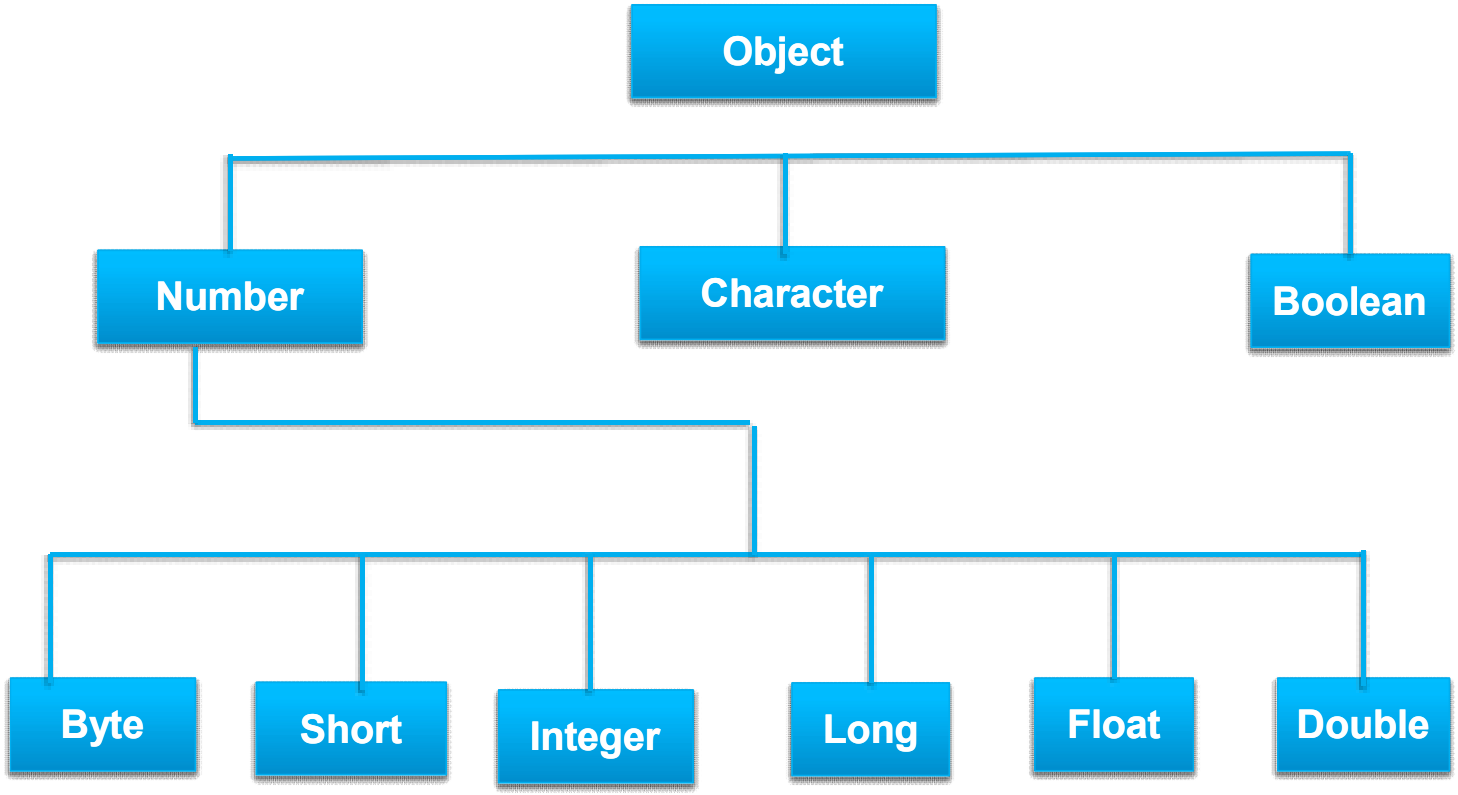
https://docs.oracle.com/javase/7/docs/api/java/lang/Integer.html

WRAPPER CLASSES

- Few more method of Integer class (These methods also available in Long, Short, Byte, Float, Double wrapper class)
- byteValue() Returns the value of the invoking object as a byte.
- doubleValue() Returns the value of the invoking object as a double.
- floatValue() Returns the value of the invoking object as a float.
- longValue() Returns the value of the invoking object as a long.
- shortValue() Returns the value of the invoking object as a short.
- Integer i1=new Integer(20);
- double d1=i1.doubleValue();

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

WRAPPER CLASSES



57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

WRAPPER CLASSES

Primitive Data Type	Wrapper Class
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean
char	Character

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

WRAPPER CLASSES


```
public class Main {
  public static void main(String[] args) {
    Integer myInt = 5;
    Double myDouble = 5.99;
    Character myChar = 'A';
    System.out.println(myInt);
    System.out.println(myDouble);
    System.out.println(myChar);
  }
}
```

```
public class Main {
  public static void main(String[] args) {
    Integer myInt = 5;
    Double myDouble = 5.99;
    Character myChar = 'A';
    System.out.println(myInt.intValue());
    System.out.println(myDouble.doubleValue());
    System.out.println(myChar.charValue());
  }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

AUTOBOXING & UNBOXING

- During assignment , the automatic transformation of the primitive type to corresponding wrapper type is known as autoboxing
- Primitive types → wrapper type **(autoboxing)**

  E. g. Integer i1=10;

- During assignment , the automatic transformation of wrapper type into their primitive equivalent is known as Unboxing
- wrapper type → primitive type **(unboxing)**

  int i=0;

  i=new Integer(10);

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

AUTOBOXING & UNBOXING

Boxing conversion converts values of primitive type to corresponding values of reference type. But the primitive types can not be widened/ Narrowed to the Wrapper classes and vice versa.

Wrong!!!

```
byte b  = 12;
Integer I1=b;
```

Wrong!!!

```
byte b  = 12;
Integer I1=(Integer)b;
```

Right!!!

```
byte b  = 12;
Integer I1=(int)b;
```

Try it and Tell me

What is the output of the following code?

```
class Test {
   public static void main(String ar[])
    {
       int x = 10;
       Integer y = new Integer(10);
       System.out.println(x == y);
}
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me

Which of the following is not a Wrapper Class?

Byte

Short

Integer

Long

String

Float

Double

Character

Boolean

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

READ INPUT IN JAVA

- In Java, apart from Command Line arguments, there are Three different ways for reading input from the user in the command line environment(console).
- **Using Buffered Reader Class**
- **Using Scanner Class**
- **Using Console Class**

Using Buffered Reader Class

- This is the Java classical method to take input, Introduced in JDK1.0.
- This method is used by wrapping the
  - **System.in** (standard input stream)
  - in an **InputStreamReader**
  - which is wrapped in a **BufferedReader,**
  - we can read input from the user in the command line.

•The input is buffered for efficient reading.

•The wrapping code is hard to remember.

Using Buffered Reader Class

```
// Enter data using BufferReader
    BufferedReader reader = new BufferedReader(
        new InputStreamReader(System.in));

    // Reading data using readLine
    String name = reader.readLine();
```

Using Buffered Reader Class

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class Test {
    public static void main(String[] args)
        throws IOException
```

Using Buffered Reader Class


```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class Test {
    public static void main(String[] args) throws IOException
    {
        BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
        String name = reader.readLine();
        System.out.println(name);
    }
}
```

Using Scanner Class

This is probably the most preferred method to take input. The main purpose of the Scanner class is to parse primitive types and strings using regular expressions.

```
import java.util.Scanner;

Scanner myObj = new Scanner(System.in);

 String userName = myObj.nextLine();
```

Using Scanner Class

Method	Description
nextBoolean()	Reads a boolean value from the user
nextByte()	Reads a byte value from the user
nextDouble()	Reads a double value from the user
nextFloat()	Reads a float value from the user
nextInt()int	Reads a int value from the user
nextLine()	Reads a String value from the user
nextLong()	Reads a long value from the user
nextShort()	Reads a short value from the user

Using Scanner Class

```
Scanner myObj = new Scanner(System.in);

    System.out.println("Enter name, age and salary:");

    // String input
    String name = myObj.nextLine();

    // Numerical input
    int age = myObj.nextInt();
    double salary = myObj.nextDouble();
```

Using Scanner Class

```
import java.util.Scanner;
class Main {
  public static void main(String[] args) {
    Scanner myObj = new Scanner(System.in);
    System.out.println("Enter name, age and salary:");
    String name = myObj.nextLine();
    int age = myObj.nextInt();
    double salary = myObj.nextDouble();
    System.out.println("Name: " + name);
    System.out.println("Age: " + age);
    System.out.println("Salary: " + salary);
  }
}
```

Using Console Class

It has been becoming a preferred way for reading user's input from the command line.

In addition, it can be used for reading password-like input without echoing the characters entered by the user

```
String name = System.console().readLine();

char[] name = System.console().readPassword()
```

Using Console Class

```
class Test
 {
    public static void main(String args[])
     {
         char[] name=System.console().readPassword();
         System.out.println(name);
     }
 }
```

Do It Yourself

Write a program to enter Name of the Student, Registration Number and marks of five subjects and calculate total, average and percentage.

Do It Yourself

Write a program to read and print elements of array

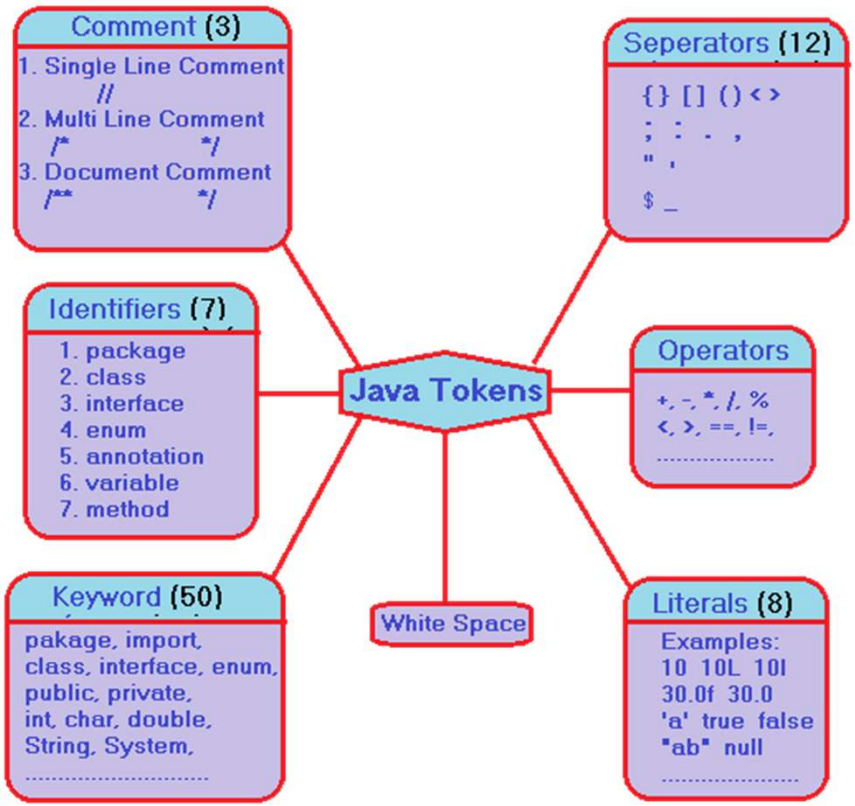
Do It Yourself

Write a program to delete all duplicate elements from an array.

Do It Yourself

Write a program to delete all duplicate elements from an array.

Tokens in JAVA




```
public class HelloWorld
{
    public static void main(String[] args)
    {
        System.out.println("Hello, World!");
    }
}
```

SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Keywords

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me

What will be the result, if we try to compile and execute the following code?


```
class Test {
    public static void main(String [ ] ar) {
        int for=2;
        System.out.println(for);
    }
}
```

Operators

- Java provides a set of operators to manipulate operations.
- Types of operators in java are,
  - Arithmetic Operators
  - Unary Operator
  - Relational Operators
  - Logical Operators
  - Simple Assignment Operator
  - Bitwise Operators

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Arithmetic Operators

Operator	Description	Example
+	Addition	A + B
-	Subtraction	A - B
*	Multiplication	A * B
/	Division	A/B
%	Modulus	A%B

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to accept two arguments of integer type and perform all the arithmetic operations and display the outputs.

Example: *java Calc 20 5*

Output:

The value of A is 20 and B is 5

The result of A + B is 25

The result of A - B is 15

*The result of A * B is 100*

The result of A / B is 4

The result of A%B is 0

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Unary Operators

Operator	Description	Example
+	Unary plus operator	+A
-	Unary minus operator	-A
++	Increment operator	++A or A++
--	Decrement operator	--A or A--

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class Sample
{
  public static void main(String args[])
   {
      int a = 10;
      System.out.println("a value is "+a);
      int b=++a;
      System.out.println("b value is "+b);
      int c=a++;
      System.out.println("c value is "+c);
      System.out.println("Final a value is "+a);
   }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Relational Operators

Operator	Description	Example
==	Two values are checked, and if equal, then the condition becomes true	(A == B)
!=	Two values are checked to determine whether they are equal or not, and if not equal, then the condition becomes true	(A != B)
>	Two values are checked and if the value on the left is greater than the value on the right, then the condition becomes true.	(A > B)
<	Two values are checked and if the value on the left is less than the value on the right, then the condition becomes true	(A < B)
>=	Two values are checked and if the value on the left is greater than equal to the value on the right, then the condition becomes true	(A >= B)
<=	Two values are checked and if the value on the left is less than equal to the value on the right, then the condition becomes true	(A <= B)

Try it and Tell me


```
class Sample{
  public static void main(String[] args){
        int a = 10;
        int b = 20;
        System.out.println("a == b = " + (a == b) );
        System.out.println("a != b = " + (a != b) );
        System.out.println("a > b = " + (a > b) );
        System.out.println("a < b = " + (a < b) );
        System.out.println("b >= a = " + (b >= a) );
        System.out.println("b <= a = " + (b <= a) );
   }
}
```

Logical Operators

Operator	Description	Example
&&	This is known as Logical AND & it combines two variables or expressions and if and only if both the operands are true, then it will return true	(A && B) is false
||	This is known as Logical OR & it combines two variables or expressions and if either one is true or both the operands are true, then it will return true	(A || B) is true
!	Called Logical NOT Operator. It reverses the value of a Boolean expression	!(A && B) is true

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
class Sample{
  public static void main(String[] args){
  boolean a = true;
  boolean b = false;
  System.out.println("a && b = " + (a&&b) );
  System.out.println("a || b = " + (a||b) );
  System.out.println("!(a && b) = " + !(a && b) );
  }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

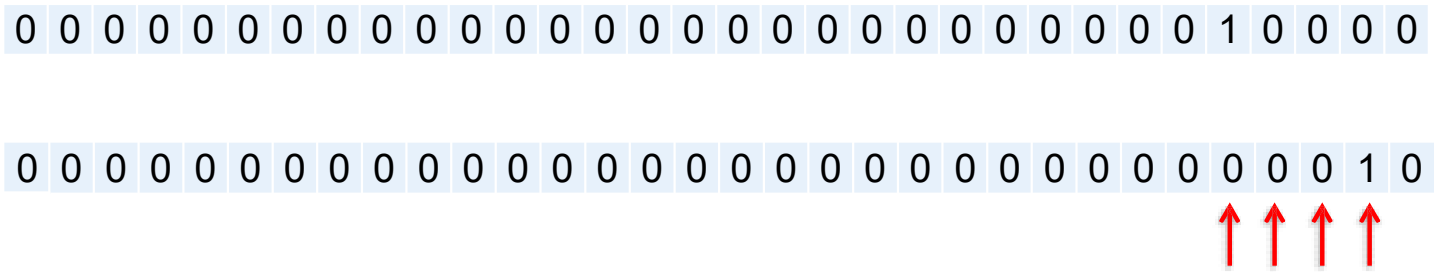
Shift Operators <<and >>

- The shift operators(<< and >>) shift the bits of a number to the left or right, resulting in a new number.
- They are used only on integral numbers(and not on floating point numbers, i.e. decimals).
- The right shift operator(>>) is used to divide a number in the multiples of 2.
- The left shift operator(<<) is used to multiply a number in the multiples of 2.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Right Shift Operator > >

- When we apply the right shift operator >>, the value gets divided by 2 to the power of number specified after the operator.
- int x = 16;
- x = x >> 3;
- 16 will be divided by the value 2 to the power of 3, which is 8.
- The result is 2.



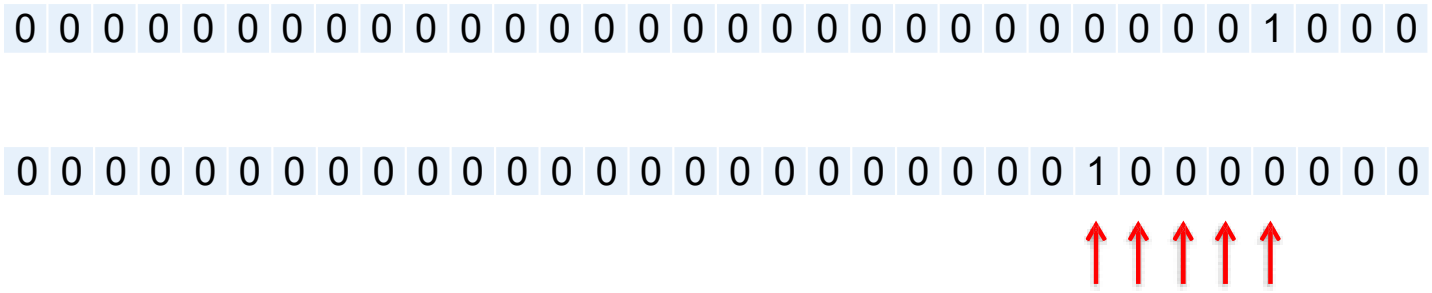
57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Left Shift Operator <<

- When we apply the left shift operator <<, the value gets multiplied by 2 to the power of number specified after the operator.

```
int x = 8;
x = x << 4;
```

- 8 will be multiplied by the value 2 to the power of 4, which is 16.
- The result is 128.



57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to accept two numbers.

Perform the following operations

1. *Left shift the First number by second number of times*
2. *Right Shift the first number by second number of times*

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Bitwise Operators

The bitwise operators take two bit numbers, use OR/AND to determine the result on a bit by bit basis.

The 3 bitwise operators are :

- & (which is the bitwise AND)
- | (which is the bitwise inclusive OR)
- ^ (which is the bitwise exclusive OR)

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Bitwise & Operators


```
class BitwiseExample1 {
  public static void main(String[] args) {
    int x = 7;
    int y = 9;
    int z = x & y;
    System.out.println("z = "+z);
  }
}
```

```
7 - >  0 1 1 1
9 - >  1 0 0 1
-------------
       0 0 0 1
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Bitwise | Operators


```
class BitwiseExample2 {
  public static void main(String[] args) {
    int x = 5;
    int y = 9;
    int z = x | y;
    System.out.println("z = "+z);
  }
}
```

```
5 -> 0 1 0 1
9 -> 1 0 0 1
-------------
     1 1 0 1
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Bitwise ^ Operators


```
class BitwiseExample3 {
  public static void main(String[] args) {
    int x = 5;
    int y = 9;
    int z = x ^ y;
    System.out.println("z = "+z);
  }
}
```

5 -> 0 1 0 1

9 -> 1 0 0 1

---

1 1 0 0

SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to accept two numbers.

Perform the following operations

1. *Bitwise AND operation on a and b*
2. *Bitwise OR operation on a and b*
3. *Bitwise EXOR operation on a and b*

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to print all the arguments passed to a ***FindArguments.java*** *program.*

Input: *Number of arguments can be 0 to 12*

Output Format:

The Number of Arguments are :

The Following are the Arguments:

Example: *java FindArguments Hi Sai Kiran*

Output:

The Number of Arguments are : 3

The Following are the Arguments:

1 Hi

2 Sai

3 Kiran

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Department of
COMPUTER SCIENCE
AND ENGINEERING

THANK YOU

PRASAD V. POTLURI SIDDHARTHA INSTITUTE OF TECHNOLOGY
VIJAYAWADA

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | **Dr. P Sai Kiran**

Department of
COMPUTER SCIENCE
AND ENGINEERING

OOP Through Java

PRASAD V. POTLURI SIDDHARTHA INSTITUTE OF TECHNOLOGY
VIJAYAWADA

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in
Professor, Department of CSE,
Prasad V Potluri Siddhartha Institute of Technology
Dr. P Sai Kiran

CONTROL STATEMENTS

- Control statements are statements which alter the normal execution flow of a program
- There are three types of Control Statements in java

Selection statement	Iteration Statement	Jumping Statement
if	while	break
if – else	for	continue
switch	do – while	return

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

SELECTION STATEMENT

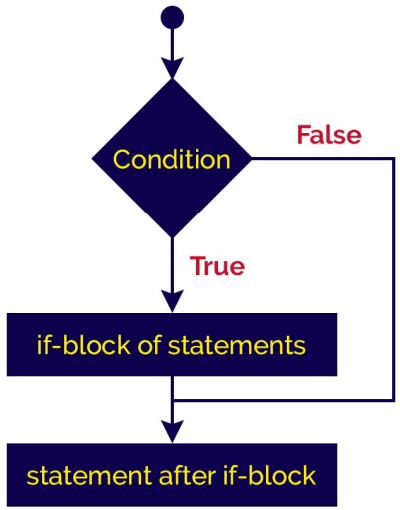
- In java, the selection statements are also known as **decision making statements** or branching statements or conditional control statements.

```
Class Vote
{
public static void main(String args[])
 {
   int age=Integer.parseInt(args[0]);
    if(age<18)
     {
        System.out.println("You are Not Eligible to Vote");
     }
 }
}
```

Syntax

```
if(condition){
        if-block of statements;
        ...
}
statement after if-block;
...
```

Flow of execution



SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

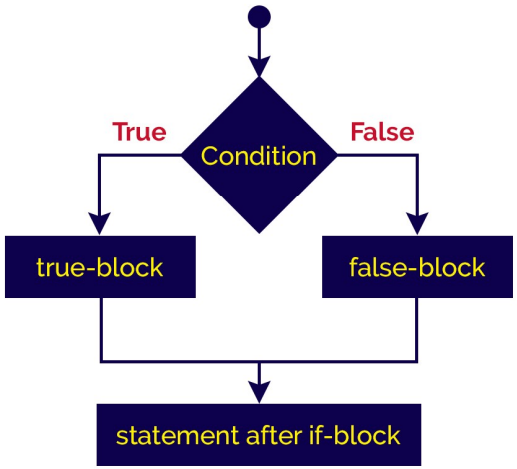
SELECTION STATEMENT


```
Class Vote
{
public static void main(String args[])
  {
    int age=Integer.parseInt(args[0]);
     if(age>=18)
      {
        System.out.println("You are Welcome to Vote");
      }
    else
      {
         System.out.println("You are Not Eligible to Vote");
      }
  }
}
```

Syntax

```
if(condition){
        true-block of statements;
        ...
}
else{
        false-block of statements;
        ...
}
statement after if-block;
...
```

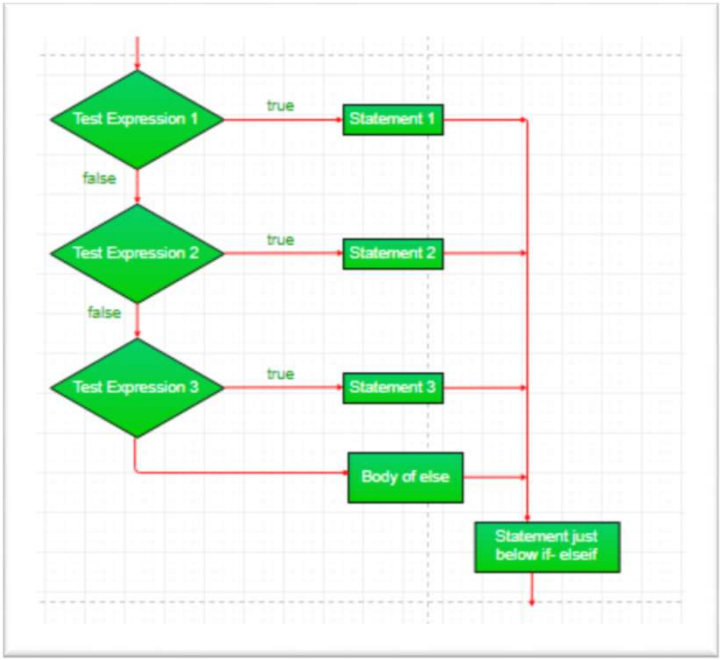
Flow of execution



57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

SELECTION STATEMENT

```
/*  program to print seasons for a month input using if &else if  */
class ElseIfDemo {
   public static void main(String[] args) {
      int month = Integer.parseInt(args[0]);
      if(month == 12 || month == 1 || month == 2)
        System.out.println("Winter");
      else if(month == 3 || month == 4 || month == 5)
        System.out.println("Spring");
      else if(month == 6 || month == 7 || month == 8)
        System.out.println("Summer");
      else if(month == 9 || month == 10 || month == 11)
        System.out.println("Autumn");
      else
       System.out.println("invalid month");
    }
}
```



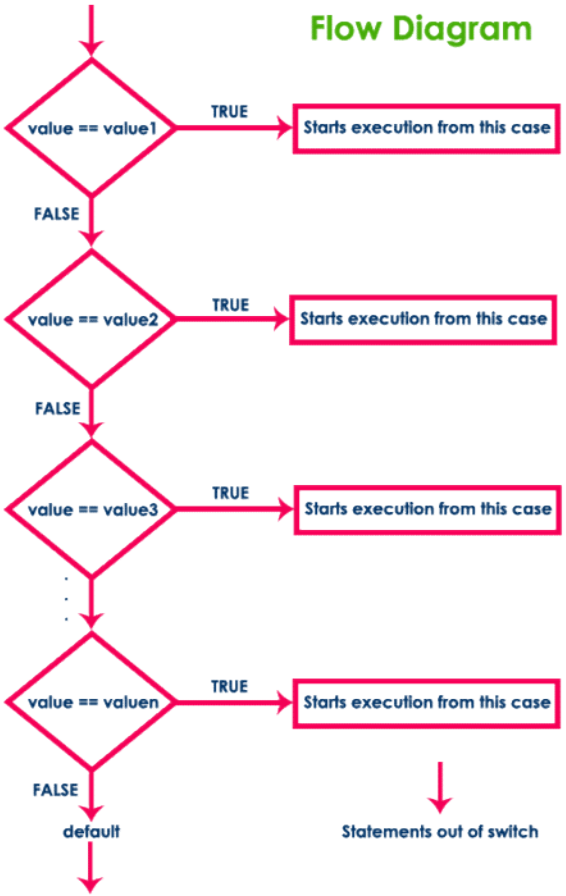
57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

SELECTION STATEMENT

Syntax

```
switch ( expression or value )
{
        case  value1:  set of statements;
                       ....
        case  value2:  set of statements;
                       ....
        case  value3:  set of statements;
                       ....
        case  value4:  set of statements;
                       ....
        case  value5:  set of statements;
                       ....
        .
        .
        default: set of statements;
}
```

Flow Diagram



SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in
Professor, Department of CSE,
Prasad V Potluri Siddhartha Institute of Technology
Dr. P Sai Kiran

SELECTION STATEMENT


```
class SwitchDemo {
    public static void main(String[] args) {
        int weekday = Integer.parseInt(args[0]);
        switch(weekday) {
        case 1: System.out.println("Sunday");
                break;
        case 2: System.out.println("Monday ");
                break;
        case 3: System.out.println("Tuesday");
                break;
        case 4: System.out.println("Wednesday");
                break;
        case 5: System.out.println("Thursday");
                break;
        case 6: System.out.println("Friday");
                break;
        case 7: System.out.println("Saturday");
                break;
        default: System.out.println("Invalid day");
```

Do It Yourself

Write a program to check if a given number is odd or even.

Input: As Argument

Example

Java EvenOdd 24

The Given Number 24 is Even

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to check if a given number is Positive, Negative, or Zero.

Input: As Argument

Example

Java PNZ 24

The Given Number 24 is Positive

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to accept gender ("Male" or "Female"), Amount (1 to 10,00,000) and age (1-120) from command line arguments and print the percentage of interest based on the given conditions.

Interest = 8.2%

Gender => Female Age =>1 to 58

Interest = 7.6%

Gender => Female Age =>59 -120

Interest = 9.2%

Gender => Male Age =>1-60

Interest = 8.3%

Gender => Male Age =>61-120

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Initialize two-character variables in a program and display the characters in alphabetical order.

Eg 1)char c1='e',c2='s'

O/P: e,s

1)char c1='a',c2='s'

O/P: a,s

1)char c1='B',c2='a'

O/P: a,B

1)char c1='b',c2='A'

O/P: A,b

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Intialize a character variable in a program and if the value is alphabet then print "Alphabet" if it's a number then print "Digit" and for other characters print "Special Character"

Char ch='@'

Output: Special Character

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to convert from upper case to lower case and vice versa of an alphabet and print the old character and new character as shown in example

(Ex: a->A, M->m).

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to print the color name, based on color code.

If color code in not valid then print "Invalid Code".

R->Red, B->Blue, G->Green, O->Orange, Y->Yellow, W->White.

The Color code is not Case Sensitive.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to print month in words, based on input month in numbers

Example 1:

C:\>java Sample 12

O/P Expected : December

Example 2:

C:\>java Sample

O/P Expected : Please enter the month in numbers

Example 3:

C:\>java Sample 15

O/P Expected : Invalid month

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

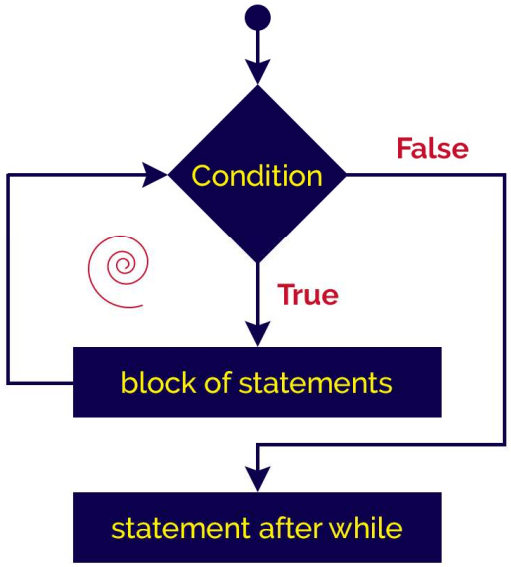
ITERATIVE STATEMENT


```
/*  This is an example for a while loop  */
class Sample{
public static void main(String[] args)
 {
   int i = 0;
   while (i < 5)
    {
      System.out.println("i: "+i);
      i = i + 1;
    }
 }
}
```

Syntax

```
while(boolean-expression){
        block of statements;
        ...
}
statement after while;
...
```

Flow of execution



SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

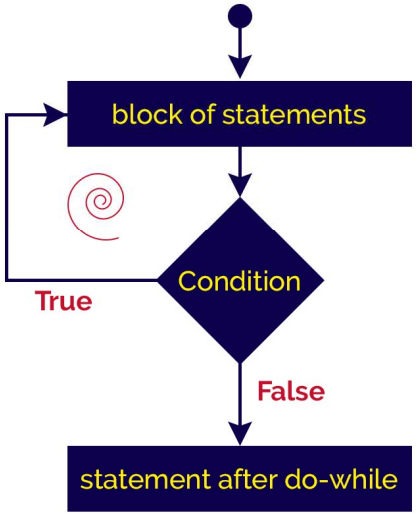
ITERATIVE STATEMENT


```
/*  This is an example of a do-while loop */
class Sample
 {
  public static void main(String[] args)
    {
     int i =5;
    do {
         System.out.println("i: "+i);
         i = i + 1;
        } while (i < 5);
    }
 }
```

Syntax

```
do{
        block of statements;
        ...
}while(boolean-expression);
statement after do-while;
...
```

Flow of execution



SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

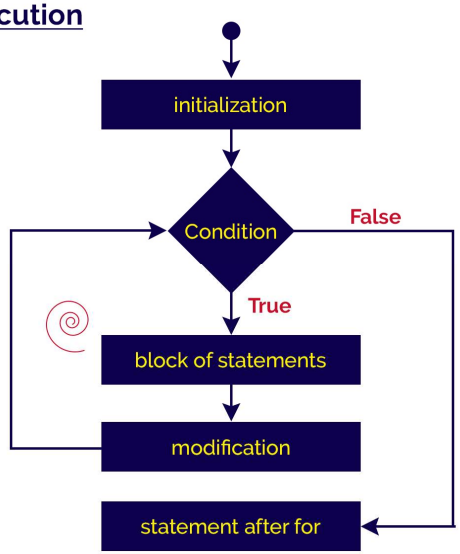
ITERATIVE STATEMENT


```
class Sample
 {
  public static void main(String[] args)
   {
     for (int i=1; i<=5; i++ )
      {
        System.out.println("i: "+i);
     }
   }
}
```

Syntax

```
for(initialization; boolean-expression; modofication){
        block of statements;
        ...
}
statement after for;
...
```

Flow of execution



SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

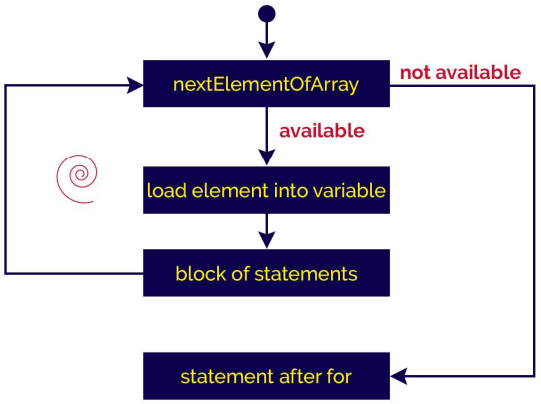
ITERATIVE STATEMENT


```
class Sample {
   public static void main(String[] args)
    {
      int [] numbers = {10, 20, 30, 40, 50};
      for(int i : numbers )
       {
         System.out.println( "i: "+i );
       }
    }
}
```

Syntax

```
for( dataType  variableName : Array ){
      block of statements;
      ...
}
statement after for;
...
```

Flow of execution



SC 57209254974 | P U-8822-2018 | VIDWAN 279509 | in drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to check if the program has received command line arguments or not. If the program has not received the values, then print "No Values", else print all the values in a single line separated by ,(comma).

Eg 1) java Example

O/P: No values

Eg 2) java Example Mumbai Bangalore

O/P: Mumbai, Bangalore

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to print even numbers between two number, where two numbers are given as command line arguments.

Each number should be printed in a separate row.

Example: java Even 10 14

10

12

14

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a Java program to find if the given number is prime or not.

Example1: C:\>java Sample

O/P Expected : Please enter an integer number

Example2: C:\>java Sample 1

O/P Expected : 1 is neither prime nor composite

Example3: C:\>java Sample 0

O/P Expected : 0 is neither prime nor composite

Example4: C:\>java Sample 10

O/P Expected : 10 is not a prime number

Example5: C:\>java Sample 7

O/P Expected : 7 is a prime number

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to print prime numbers between 1 and 99.

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to add all the values in a given number and print.

Ex: 1234->10

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a program to reverse a given number and print

Eg 1)

I/P: 1234

O/P:4321

Eg 2)

I/P:1004

O/P:4001

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Do It Yourself

Write a Java program to find if the given number is palindrome or not

Example1:

C:\>java Sample 110011

O/P Expected : 110011 is a palindrome

Example2:

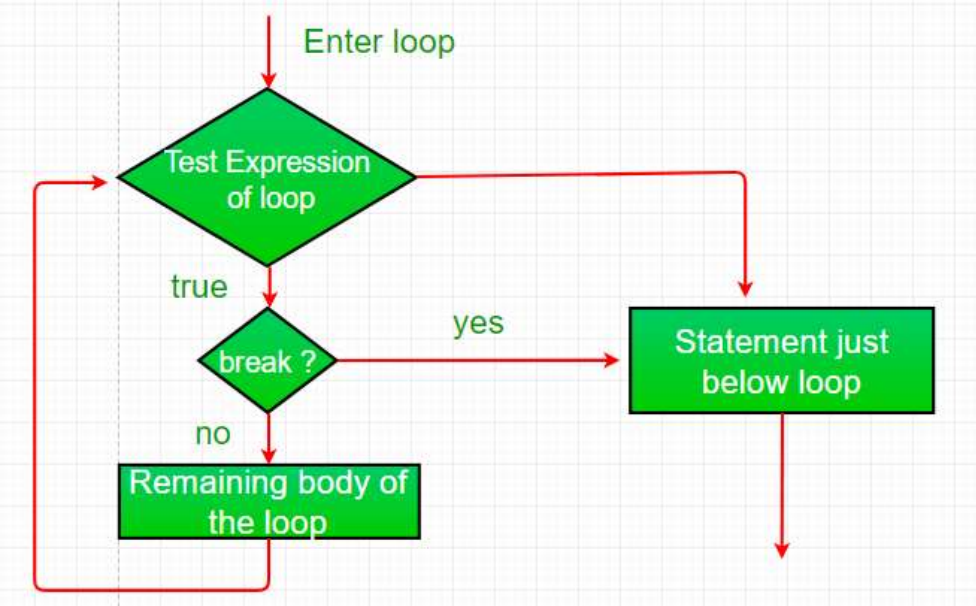
C:\>java Sample 1234

O/P Expected : 1234 is not a palindrome

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

JUMPING STATEMENT

- While the execution of program, the break statement will terminate the iteration or switch case block
- When a **break statement** is encountered in a loop, the loop is exited, and the program continues with the statements immediately following the loop
- When the loops are nested, the break will only terminate the corresponding loop body



57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in
Professor, Department of CSE,
Prasad V Potluri Siddhartha Institute of Technology
Dr. P Sai Kiran

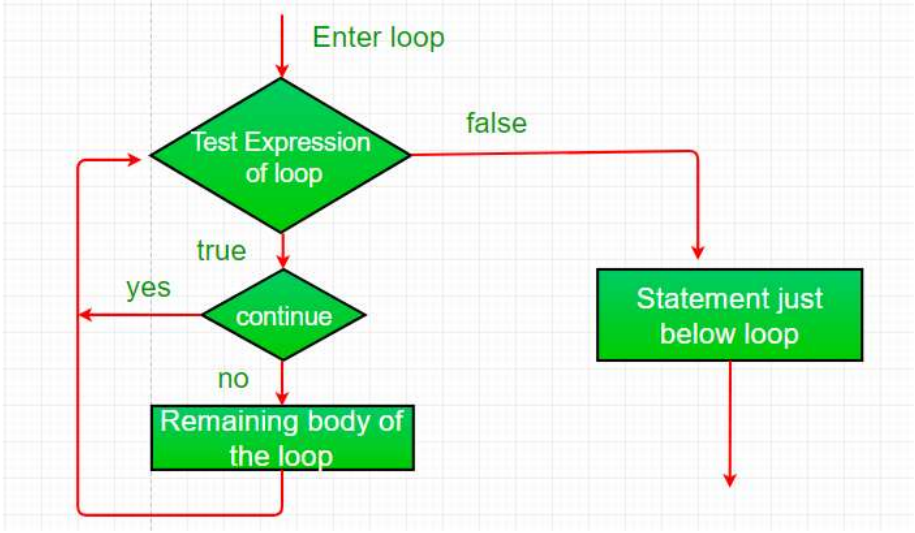
Try it and Tell me


```
public class Sample{
  public static void main(String[] args)
  {
    for (int    i=1; i<=5; i++ )
    {
        if(i==2)
          {
            break;
          }
        System.out.println("i: "+i);
    }
  }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

JUMPING STATEMENT

- The continue statement skips the current iteration of a loop
- In while and do loops, continue causes the control to go directly to the test-condition and then continue the iteration process
- In case of for loop, the increment section of the loop is executed before the test-condition is evaluated



57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Try it and Tell me


```
public class Sample {
  public static void main(String[] args) {
    int [] numbers = {1, 2, 3, 4, 5};
    for(int i : numbers ) {
      if( i == 3 ) {
            continue;
        }
        System.out.println( "i: "+i );
    }
  }
}
```

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | Dr. P Sai Kiran

Department of
COMPUTER SCIENCE
AND ENGINEERING

THANK YOU

PRASAD V. POTLURI SIDDHARTHA INSTITUTE OF TECHNOLOGY
VIJAYAWADA

57209254974 | U-8822-2018 | VIDWAN 279509 | drpsaikiran | 9490856188 | psaikiran@pvpsiddhartha.ac.in | Professor, Department of CSE, Prasad V Potluri Siddhartha Institute of Technology | **Dr. P Sai Kiran**