VIVEKANAND EDUCATION SOCIETY’S

College of Arts, Science and Commerce

# Core Java

Prerequisite

To learn Java, you must have the basic knowledge of C/C++ programming language.

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

# VIVEKANAND EDUCATION SOCIETY’S

V.E.S.
Since 1962

## College of Arts, Science and Commerce

- Textbook(s):
  - 1) Herbert Schildt, Java The Complete Reference, Ninth Edition, McGraw-Hill Education, 2014

- Additional Reference(s):

- 1) E. Balagurusamy, Programming with Java, Tata McGraw-Hill Education India, 2014
- 2) Programming in JAVA, 2nd Ed, Sachin Malhotra & Saurabh Choudhary, Oxford Press 3) The Java Tutorials: http://docs.oracle.com/javase/tutorial/

# Introduction

- Java is a **programming language** and a **platform**.
- Java is a high level, robust, object-oriented and secure programming language.
- It was developed by ‘James Gosling’ at Sun Microsystems.
- First version of Java was released in 1995.
- Before Java, its name was *Oak*
- Java provides the functionality of ‘write once, run anywhere’ (WORA).

**Platform**: Any hardware or software environment in which a program runs, is known as a platform. Since Java has a runtime environment (JRE) and API, it is called a platform.

# Application

- There are many devices where Java is currently used. Some of them are as follows:

1. Desktop Applications such as acrobat reader, media player, antivirus, etc. 2. Web Applications such as irctc.co.in, javatpoint.com, etc. 3. Enterprise Applications such as banking applications.

4. Mobile

5. Embedded System

6. Smart Card
7. Robotics
8. Games, etc.

# Functionalities of Java

- Java provides a huge library.
- Auto memory cleanup process i.e. automatic garbage collection.
- Platform independent means it is portable on every operating system.

- Code reusability which allows us to use the previous methods in next class.

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY'S**

**College of Arts, Science and Commerce**

# Types of Java Applications 1)

## Standalone Application

- AWT and Swing are used in Java for creating standalone applications.

### 2) Web Application

- . Currently, Servlet, JSP, Struts, Spring, Hibernate, JSF, etc. technologies are used for creating web applications in Java.

### 3) Enterprise Application

- In Java, EJB is used for creating enterprise applications.

### 4) Mobile Application

- An application which is created for mobile devices is called a mobile application. Currently, Android and Java ME are used for creating mobile applications.

# Java Platforms / Editions

1. Java SE (Java Standard Edition)
   - This is the core Java programming platform. It contains all of the libraries and APIs that any Java programmer should learn (java.lang, java.io, java.math, java.net, java.util, etc...)
2. Java EE (Java Enterprise Edition)
   - It is an enterprise platform which is mainly used to develop web and enterprise applications. It is built on the top of the Java SE platform.
   - It includes topics like Servlet, JSP, Web Services, EJB, JPA, etc.
3. Java ME (Java Micro Edition)
   - This is the platform for developing applications for mobile devices and embedded systems such as set-top boxes.
   - Java ME provides a subset of the functionality of Java SE, but also introduces libraries specific to mobile devices.

# History of Java

- The history of Java starts with the Green Team. Java team members (also known as **Green Team**).
- Java is an Object-Oriented programming language developed by **James Gosling** in the early 1990s.
- The team initiated this project to develop a language for digital devices such as set-top boxes, television, etc. Originally C++ was considered to be used in the project but the idea was rejected for several reasons(For instance C++ required more memory).

- Initially designed for small, embedded systems in electronic appliances like set-top boxes.
- Firstly, it was called **"Greentalk"** by James Gosling, and the file extension was .gt.
- After that, it was called **Oak** and was developed as a part of the Green project

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY'S**

**College of Arts, Science and Commerce**

# Conti….

- **Why Oak?**

  - Oak is a symbol of strength and chosen as a national tree of many countries like the U.S.A., France, Germany, Romania, etc.
  - In 1995, Oak was renamed as **"Java"** because it was already a trademark by Oak Technologies.
- Why Java
  - Gosling and his team did a brainstorm session and after the session, they came up with several names such as **JAVA, DNA, SILK, RUBY, etc.**

  - **Java** name was decided after much discussion since it was so unique. The name Java originates from a sort of **espresso bean**, Java. Gosling came up with this name while having a coffee near his office.

  - Java was created on the principles like **Robust, Portable, Platform Independent, High Performance, Multithread, etc.** and was called one of the **Ten Best Products of 1995** by the **TIME MAGAZINE**.

# Java Version History

Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:

VIVEKANAND EDUCATION SOCIETY’S

# College of Arts, Science and Commerce

For developing java application we need software JDK(Java Development kit)

- Compiler compiles the .java file and then creates .class file
- This .class file is not understand by any OS
- JVM is different for different OS
- JAVA is compiled and interpreted language.

- Java is compiled and interpreted language.

# JDK

- JDK (Java Development Kit) is a software development kit required to develop applications in Java.
- When you download JDK, JRE is also downloaded with it.

- In addition to JRE, JDK also contains a number of development tools (compilers, JavaDoc, Java Debugger, etc).

# JRE

- JRE stands for **JAVA RUNTIME ENVIRONMENT**
- The Java Runtime Environment provides the minimum requirements for executing a Java application; it consists of the *Java Virtual Machine (JVM), core classes*, and *supporting files*.
- . In simple terms, if you want to run Java program you need JRE. If you are not a programmer, you don't need to install JDK, but just JRE to run Java programs.

# JVM

- JVM is an engine that provides a runtime environment to drive the Java Code or applications.
- It converts Java byte code into machine language.
- JVM is a part of Java Run Environment (JRE).

Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:

# VIVEKANAND EDUCATION SOCIETY’S

**College of Arts, Science and Commerce**

# How java is platform independent

# Conti…

conti........

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY'S**

# Conti…

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

conti….

~~conti…~~

# Features of Java

1) Simple

- Java syntax is based on C++ (so easier for programmers to learn it after C++).

- no concept of pointer, dynamic memory allocation, operator overloading etc. 2) Object-oriented

- Object-oriented

- Basic concepts of OOPs are:
  - Object
  - Class
  - Inheritance
  - Polymorphism

- Abstraction
- Encapsulation

# Conti…

## 3) Platform Independent

- Java is platform independent because it is different from other languages like C, C++, etc. which are compiled into platform specific machines while Java is a write once, run anywhere language. A platform is the hardware or software environment in which a program runs.

# Conti…..

## 4)Secured

- Java is best known for its security. With Java, we can develop virus-free systems. Java is secured because:
- Java doesn’t support **explicit pointer**
- JVM provides a virtual machine for running the Java programs.

## 5) Robust

- robust simply means strong. Java is robust because:
- It uses strong memory management.
- There is a lack of pointers that avoids security problems.
- There is automatic garbage collection in java 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. All these

points make Java robust.

# Conti….

## 6)Architecture-neutral

- Java is architecture neutral because there are no implementation dependent features, for example, the size of primitive types is fixed.
- In C programming, int data type occupies 2 bytes of memory for 32-bit architecture and 4 bytes of memory for 64-bit architecture. However, it occupies 4 bytes of memory for both 32 and 64-bit architectures in Java

## 7) Portable

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

## 8) High-performance

- Java is faster than other traditional interpreted programming languages because Java bytecode is "close" to native code. It is still a little bit slower than a compiled language (e.g., C++). Java is an interpreted language that is why it is slower than compiled languages, e.g., C, C++, etc.

# Conti…..

## 9) Distributed

- Java is distributed because it facilitates users to create distributed applications in Java.
- 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.

## 10) Multi-threaded

- Java is a multithreaded programming language because it allows multiple tasks to be performed simultaneously to share the same memory. 11) Dynamic

- Java can carry lots of information on run time as it provides a link on different classes and objects dynamically.

# Difference between C++ and Java

**Name of**

- Note: Java does not support header files like C++. Java uses the  import keyword to include different classes and methods.

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

# Java Tokens

- Tokens in Java are the small units of code which a Java compiler uses.

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

# VIVEKANAND EDUCATION SOCIETY’S

# Conti…..

- **Keywords-**
  - Keywords in Java are predefined or reserved words that have special meaning to the Java compiler.
  - You cannot use keywords as variables or identifiers as they are a part of Java syntax itself.
  - A keyword should always be written in lowercase as Java is a case sensitive language. Java supports various keywords, some of them are listed below: • Ex:abstract.continue,for new switch etc

**•Identifiers**

- Java Identifiers are the user-defined names of variables, methods, classes, arrays, packages, and interfaces.

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

**College of Arts, Science and Commerce**

# Conti….

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

# Conti…..

- Literals
  - Literals in Java are similar to normal variables but their values cannot be changed once assigned. In other words, literals are constant var~~iables wi~~th fixed values. These are defined by users and can belong to any data type. Java supports five types of literals which are as follows:
    - Integeral Literal
      - For integral data types(byte,short,int,long)
        - Decimal literal(Base 10)
        - Octal Literal(Base 8)
        - Hexa Decimal(nos 0-9 char a-f(we can use both upper case lower case))
    - Floating Point
      - Nos having decimal part eg 10.2
    - Character
      - Enclosed in single quotes.
    - String

    - Sequence of character enclosed in double quotes.
- Boolean
    - Are either true or false.

# Conti…..

- Seprators
    - special symbols are used in Java having some special meaning and thus, cannot be used for some other purpose.[] () {}, ; * =

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

**College of Arts, Science and Commerce**

# Conti……..

- Comments
  - Comments can be used to explain Java code, and to make it more readable. It can also be used to prevent execution when testing alternative code. • Single-line comments start with two forward slashes (//).
  - Multi-line comments start with /* and ends with */.

# Java Data types

# OOPS Concept

- **Object-Oriented Programming** is a methodology or paradigm to design a program using classes and objects.
  - Object
  - Class
  - Inheritance
  - Polymorphism
  - Abstraction
  - Encapsulation

- Object
  - Any entity that has state and behavior is known as an object. For example, a chair, pen, table, keyboard, bike, etc. It can be physical or logical. • An Object can be defined as an instance of a class.
  - **Example:** A dog is an object because it has states like color, name, breed, etc. as well as behaviors like wagging the tail, barking, eating, etc.
- Class
  - A class can also be defined as a blueprint from which you can create an

individual object.

OR

Class is collection of member fields and member variables.

- Inheritance
  - *When one object acquires all the properties and behaviors of a parent object,* it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

- Polymorphism
  - If *one task is performed in different ways*, it is known as polymorphism. For example: to convince the customer differently, to draw something, for example, shape, triangle, rectangle, etc.
  - In Java, we use method overloading and method overriding to achieve polymorphism

- Abstraction
  - *Hiding internal details and showing functionality* is known as abstraction. For example phone call, we don't know the internal processing.
  - Example : .apk .exe
- Encapsulation
  - Binding variables and method under single entity.

# Conti….

Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:

VIVEKANAND EDUCATION SOCIETY’S

# Naming Convention

- Class object variable and method name should not match with keyword
- Class should begin with capital letters
- For multiple word First letter of every word should be

capital. • Example 1: class Student

```
{
        //line of code
}
```

Example 2: class BranchName

```
{
}
```

- Method
  - First letter should begin with lowercase
  - 1st letter of 1st word=>lowercase
  - 1st letter of 2nd word=>uppercase
  - Example: read(),write(),getName(),nextLine()

- Variable
  - Variable name should not be keyword
  - Except only underscore no other special characters are allowed (space alnot allowed.)
  - Variable maximum length <=31
  - Variable name can be alphanumeric.

# Java I/O

- **Java I/O** (Input and Output) is used *to process the input* and *produce* *the output*.

- Stream
  - A stream is a sequence of data. In Java, a stream is composed of bytes. It's called a stream because it is like a stream of water that continues to flow. • In Java, 3 streams are created for us automatically. All these streams are attached with the console.
  - **1) System.out:** standard output stream
  - **2) System.in:** standard input stream
  - **3) System.err:** standard error stream

- Output
  - System.out.print()=>Everything will be printed on single line
  - System.out.println()=>o/p will be printed on new line
  - System.out.prntf()=> similar to the *printf()* function in C. • Example System.out.printf("Hello %s!%n", "World");This produces the following output:Hello World!
- Input
  - Scanner
  - BufferedReader

```
String myStr = "Hello";
char result = myStr.charAt(0);
System.out.println(result);
```

- Definition and Usage
  - The charAt() method returns the character at the specified index in a string.
  - The index of the first character is 0, the second character is 1, and so on.

**VIVEKANAND EDUCATION SOCIETY’S**

**College of Arts, Science and Commerce**

# Conditional Statements In Java

- 1. if statement
  2. nested if statement
  3. if-else statement
  4. if-else-if statement
  5. Switch Case Statement

# if statement:

- **if statement:**
  - The if statement is the most basic of all the control flow statements. • The if statement tells our program to execute a certain section of code only if a particular test evaluates to true.

**Syntax:**

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

# VIVEKANAND EDUCATION SOCIETY’S

## College of Arts, Science and Commerce

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

```
public class sample
{
 public static void main(String[] args)
{
 int num = 100;
 if (num<=100)
  {
 System.out.println("Value of num is "+num);  }
}

}
```

# Nested if statement:

- An if statement inside another the statement. If the outer if condition is true then the section of code under outer if condition would execute and it goes to the inner if condition. If inner if condition is true then the section of code under inner if condition would execute. **Syntax:**

```
if(condition_1)
 {
 Statement1(s);

 if(condition_2)
 {
 Statement2(s);
 }
}
```

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

## VIVEKANAND EDUCATION SOCIETY’S

College of Arts, Science and Commerce

# if-else statement:

- If a condition is true then the section of code under if would execute else the section of code under else would execute.

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

# Flow chart

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

```
public class IfElseStatement
{
 public static void main(String[] args) {
 int num = 100;
 if (num>100)
 {
 System.out.println("Value is greater than 100");  }
 else
 {
 System.out.println("Value is less than 100");  }
 }

}
```

# if-else-if ladder:

- As soon as one of the conditions controlling the if is true, the statement associated with that if is executed, and the rest of the ladder is bypassed.

```
if (condition)
statement;
else if (condition)
statement;
```

```
. .
else statement;
```

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

# nested if

```
 if(condn1)
if(condn 2)
if(condnt 3)
    if(condnt 4) stmt
   else
stmt4
else
stmt3
else
stmt 2
else
```

stmt1

# Example

Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:

# switch-case

- If a condition is true then the section of code under if would execute else the section of code under else would execute.

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

```
switch (expression) {
    case value1:
    statement1;
    break;
    case value2:
    statement2;
    break;
    .
    .
    case valueN:
    statementN;
    break;
    default:
    statementDefault; }
```

# if-else statement:

- If a condition is true then the section of code under if would execute else the section of code under else would execute.

```
if(condition)
{
```

```
Statement(s);
}
else
{
Statement(s);
}
```

# Conti....

- Expression can be of type byte, short, int char or an enumeration.

Beginning with JDK7, *expression* can also be of type String.

- Dulplicate case values are not allowed.
- The default statement is optional.
- The break statement is used inside the switch to terminate a statement sequence.
- The break statement is optional. If omitted, execution will continue on into the next case.

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

# College of Arts, Science and Commerce

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

# Loops in Java

- Looping in programming languages is a feature which facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true.
- **while loop:** A while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The while loop can be thought of as a repeating if statement.
- **Syntax :**

```
while (boolean condition)
{
loop statements...
```

```
    }
```

```
class whileLoopDemo
{
 public static void main(String args[])  {
 int x = 1;

 // Exit when x becomes greater than 4
while (x <= 4)
 {
 System.out.println("Value of x:" + x);
 // Increment the value of x for  //
next iteration
x++;
}
}
}
```

- **for loop:**
  - for loop provides a concise way of writing the loop structure. Unlike a while loop, a for statement consumes the initialization, condition and increment/decrement in one line thereby providing a shorter, easy to debug structure of looping.

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

- **Initialization condition:** Here, we initialize the variable in use. It marks the start of a for loop. An already declared variable can be used or a variable can be declared, local to loop only.
- **Testing Condition:** It is used for testing the exit condition for a loop. It

must return a boolean value. It is also an **Entry Control Loop** as the condition is checked prior to the execution of the loop statements.

- **Statement execution:** Once the condition is evaluated to true, the statements in the loop body are executed.
- **Increment/ Decrement:** It is used for updating the variable for next iteration.
- **Loop termination:**When the condition becomes false, the loop terminates marking the end of its life cycle.

```
class forLoopDemo
{
public static void main(String args[])   {
// for loop begins when x=2
// and runs till x <=4
for (int x = 2; x <= 4; x++)
System.out.println("Value of x:" + x);   }
}
```

# do while:

- do while loop is similar to while loop with only difference that it checks for condition after executing the statements, and therefore is an example of **Exit Control Loop.**
- **Syntax:**

```
do
{
        statements..
}
while (condition);
```

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

**College of Arts, Science and Commerce**

```
class dowhileloopDemo
{
 public static void main(String args[])   {
 int x = 21;
 do
 {
```

```
 // The line will be printed even   // if
the condition is false
 System.out.println("Value of x:" + x);   x++;
 }
 while (x < 20);
 }
}
```

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY’S**

**College of Arts, Science and Commerce**

# Typecasting in java

- Typecasting is used to convert an object variable
- Type casting is when you assign a value of one primitive data type to another type.
- In Java, there are two types of casting:
- **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

# Widening Casting

- Widening casting is done automatically when passing a smaller size type to a larger size type:
  - public class MyClass {
  - public static void main(String[] args) {
  - int myInt = 9;
  - double myDouble = myInt; // Automatic casting: int to double

  - System.out.println(myInt); // Outputs 9

- System.out.println(myDouble); // Outputs 9.0
- }
- }

# Narrowing Casting

- Narrowing casting must be done manually by placing the type in parentheses in front of the value:
  - public class MyClass {
  - public static void main(String[] args) {
  - double myDouble = 9.78;

- int myInt = (int) myDouble; // Manual casting: double to int

- System.out.println(myDouble); // Outputs 9.78
- System.out.println(myInt); // Outputs 9
- }
- }

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY'S**

**College of Arts, Science and Commerce**

# Arrays

- Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.

- n Java all arrays are dynamically allocated.
- The variables in the array are ordered and each have an index beginning from 0.
- Index number is also called as subscript.
- Array can contains primitives (int, char, etc) as well as object (or non-primitives) references of a class depending on the definition of array. In case of primitives data types, the actual values are stored in contiguous memory locations

Advantages

**Code Optimization:** It makes the code optimized, we can retrieve or sort the data efficiently.

**Random access:** We can get any data located at an index position.

Disadvantages

**Size Limit:** We can store only the fixed size of elements in the array. It doesn't grow its size at runtime. To solve this problem, collection framework is used in Java which grows automatically.

# Single Dimensional Array in Java

- **Syntax to Declare an Array in Java**
  - dataType[] arr; (or)
  - dataType []arr; (or)
  - dataType arr[];
- Example
- The following code snippets are examples of this syntax −
  - Double [] myList; // preferred way. or
  - double myList[]; // works but not preferred way.

- **Instantiation of an Array in Java**
  - arrayRefVar=**new** datatype[size];
- **Example:**
  - int intArray[]; //declaring array
  - intArray = new int[20]; // allocating memory to array OR
  - int[] intArray = new int[20]; // combining both statements in one

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY'S**

**College of Arts, Science and Commerce**

- **Declaration, Instantiation and Initialization of Java Array** • We can declare, instantiate and initialize the java array together by: • **int** a[]={33,3,4,5};//declaration, instantiation and initialization • Let's see the simple example to print this array.

- //Java Program to illustrate the use of declaration, instantiation
- //and initialization of Java array in a single line

```
class Testarray1{
public static void main(String args[]){
int a[]={33,3,4,5};//declaration, instantiation and initialization
//printing array
for(int i=0;i<a.length;i++)//length is the property of array
System.out.println(a[i]);
}
}
```

**Name of the Teacher: Laxmi Tiwari Class: SYB.Sc CS Subject Core Java Paper:**

**VIVEKANAND EDUCATION SOCIETY'S**

**College of Arts, Science and Commerce**

- Example

- Following statement declares an array variable, myList, creates an array of 10 elements of double type and assigns its reference to myList −
- double[] myList = new double[10];Following picture represents array myList. Here, myList holds ten double values and the indices are from 0 to 9.