```
**************************************************
*  Class Name : VJTech Academy , Maharashtra     *
*  Author Name: Vishal Jadhav Sir                *
*  Mobile No  : 9730087674                       *
**************************************************
```

===

UNIT-I :Basic Syntactical constructs in Java

===

Features of OOP:

================

1) Emphasis is on data rather than procedure.
2) Programs are divided into Objects.
3) Data is hidden and can not accessed from the external functions.
4) Objects may communicate with each other through functions.
5) New data and functions can be added easily whenever required.
6) OOP follow bottom-up approach in program design.
7) Object is a collection of data & functions, function operates on data.
8) More security provided for the data compare to POP.
9) Data cannot move openely from one function to another function.

======================

Basic concepts of OOP:

======================

1) Objects:
- Basic runtime entities known as object.
- They may represent person, table, bank account or any item that program may handle.
- Object is a collection of data and functions.
- Objects are created from class.

2) Classes:
- It is a collection of similiar types of objects.
- Class is a collection of data and functions, functions operate on data.
- You may creates objects from the class.
- When you define the class then memory will not be allocated for the members.
- Class shows data abstraction and data encapsulation features.
- Example: Fruit mango
- In above example, mango is an object which is created from class Fruit.

3) Data Abstraction:
- To show only essential details without background details.

4) Data Encapsulation:
- The wrapping up of data and functions into a single unit is known as Data Encapsulation.

5) Inheritance:
- The process of creating new class by using the concept of old class is known as Inheritance.
- Newly created class is known as Derived class.

- Old class is known as Base class.

6) Polymorphism:
- Polymorphism is a greek word.
- Poly means 'many' and morphism means 'forms'.
- Ability to take more than one forms is known as Polymorphism.
- There are two different types of Polymorphism
-> Compile time Polymorphism
   => Function Overloading
   => Operator Overloading //this is not supported in java
-> Run time Polymorphism
   => Virtual Function

7) Dynamic Binding:
- The linking between calling function and called function is known as binding.
- But that linking is not known until the execution of program is known as dynamic binding.
- Example:

```
        vjtech();   //calling function
        void vjtech() //called function
        {
          //body
        }
```

8) Message Passing:
- In OOP, we can create set of objects that communicate with each other.
- I) Creating classes that define objects.
  II) Creating objects from the class definition.
  III) Establishing communication among objects.

=====================
Benefits of OOP
====================

1) Using concept inheritance, we can achieve reusability.
2) Data hiding
3) Software complexity can be easily managed.
4) Object oriented system can be easily upgraded from small to large systems.
5) It is easy to partition work in the project based on objects.

===================
Application of OOP
===================

1) Real time systems.
2) Simulation and modeling
3) Object-oriented databases
4) Hypertext,hypermedia and experttext.
5) AI and expert systems.
6) Neaural networks and parallel programming.
7) Decision support and office automation System.
8) CIM/CAD System

Java History:

================
- Java is a general purpose, object oriented programming language.
- It was developed by Sun MircoSystem of USA.
- James Gosling is owner of Java language.
- It was developed in year 1991.
- Initially, it was called as Oak (tree name)
- It's name got changed to Java in year 1995.

================
Java Features:
================
1) Compiled and Interpreted:

- Usually computer language is either compiled or interpreted.
- But java combines both these approaches thats via Java is called as two stage compilation process programming language.
- Java compiler takes java source file(.java) as input and generates byte file(.byte).
- Byte file is not a machine code and this file not exists physically in your machine.
- Byte code generated virtually and process virtually.
- Java Interpreter generates machine code from byte code.

2) Platform Independent and Portable:

- Java program can be easily moved from one computer to another computer,anywhere and anytime.
- It means, if we develop Java code on Windows machine then you can easily run that code on other operating systems like Linux,Unix,etc.
- To move java code from one machine to another machine is known as portability.

3) Object Oriented:

- Java is true object-oriented language.
- Almost everything in java is an object.
- All program code and data reside within objects and classes.
- Java is a colletion of rich set of predefined classes and packages, that we can use
in our programs by inheritance.

4) Robust & Secure:

- Java is robust language.
- It provides many safeguards to ensure reliable code.
- It has strict compile time and runtime checking for data types.
- It supports garbage collection feature that would solve memory management problem.
- It supports exception handling which help us to captures many errors.
- Java is more secure programming language which is used for programming on internet.

- Java systems not only verify all memory access but also ensure that no viruses are
communicated with an applet.

5) Distributed:

- Java is designed as distributed langauge for creating applications on networks.
- It has ability to share both data and programs.
- Java application can open and access remote objects on internet.
- This enables multiple programmers at multiple locations to collaborate and work together on a single object.

6) Simple, Small and Familiar:

- Java is a small and simple language.
- Many features of C & C++ which are not relible that was not added in Java.
- Java does not use pointers, preprocessor directive, goto statement and many others.
- Also not included multiple inheritance and operator overloading features.
- Familiarity is another important feature of Java.

7) Multi-threaded and Interactive:

- Thread is a light weight process because it takes small amount of memory space for their execution.
- When mutliple thread executes simultaneously then it is called as Multithreading.
- Java supports multithreaded programs.
- This means that we need not to wait for the application to finish one task before

beginning another.
- Due to this feature, we can create more interactive programs in java.

8) High Performance:

- Java performance is impressive for an interpreted language, mainly due to the use

of byte code.
- Java speed is more faster than C/C++ language.
- Java architecture is also designed to reduce overhead during the runtime.
- Due to multi-threading fetures Java program execution speed is increased.

9) Dynamic & Extensible:

- Java is dynamic language.
- Java is capable of dynamically linking in new class libraries,methods and objects.
- Java programs support functions written in othet languages such as c and C++.
- This facility enables the programmers to use the efficient functions in these languages.
- It is called as native functions/methods.

- Native methods are linked dynamically at runtime.

===
Difference between Java and C
===============================
1) Java is Object oriented programming language and C is Procedure oriented programming.
2) Java does not include c keyword sizeof and typedef.
3) Java does not contain data types struct and union.
4) Java does not define the data type modifiers keyword auto,extern,register,signed and
unsigned.
5) Java does not support pointer concept.
6) Java does not have preprocessor directive and thats via we don't use #define, #include
7) Java adds labeled break and continue statements.
8) Java adds new operators such as instanceof.
9) Java adds many required features of Object Oriented Programming Language.

===
Difference between Java and C++
===============================
1) C++ is object-oriented programming but Java is true object-oriented programming.
2) Java does not support operator overloading.
3) Java does not support pointer concept.
4) Java does not support multiple inheritance. But you can implement it using new feature called 'interface'.
5) Java does not support global variable.
6) Java does not support destructor function but we use finalize() method.
7) There is no header files in Java.
==================================
Java Environment
=================
- Java environment includes development tools(JDK) and classes & method(JSL-API).
- JDK stands for Java development kit.
- JSL stands for Java Standard Library.
- JRE stands for Java Runtime Environment.

1) Java Development kit(JDK)

```
=> javac                : Java compiler (javac filename.java)
=> java                 : Java Interpreter (java filename)
=> jdb                  : Java Debugger
=> appletviewer : for running java applets.
=> javah                : for c header files.
=> javadoc      : for creating HTML documents.
=> javap        : Java disassembler(convert byte file to program)
```

Comments:

- Comments part ignored by the compiler.

1) Single line comment

```
//This is single line comment
```

2) Multi-line comment

```
/*
This is multi-line comment
*/
```

Command line arguments:

- Give input to Java code.
- Example: java filename <list of input values>

```
                   java VJTech 100 200
```

- Program:

```
//Command line arguments
class CommandLineArgsDemo
{
        public static void main(String args[])
        {
                int a,b,c;
                a=Integer.parseInt(args[0]);
                b=Integer.parseInt(args[1]);
                c=a+b;
                System.out.println("Addition of two numbers="+c);
        }
}
```

Scanner Class:

Method	Description
nextInt()	reads an int value from the user
nextFloat()	reads a float value form the user
nextBoolean()	reads a boolean value from the user
nextLine()	reads a line of text from the user
next()	reads a word from the user
nextByte()	reads a byte value from the user
nextDouble()	reads a double value from the user
nextShort()	reads a short value from the user
nextLong()	reads a long value from the user

Mathematical Functions:

- java.lang package contain Math class.
- If you want to access methods of Math class then use below syntax:
  Math.MethodName();

Math.min(Variable1,Variable2) - find minimum value

```
Math.max(Variable1,Variable2) - find maximum value
Math.sqrt(VariableName)                 - find square root of given number
Math.pow(Variable1,Variable2) - return power of given number
Math.exp(VariableName)          - to calculate exponential value of given number
Math.round(Variable)                    - it return rounded value.
Math.abs(Variable)                      - It is used to find out absolute value.
```

- Example:

```
class MathMethods
{
        public static void main(String args[])
        {
                int m=12,n=12;
                System.out.println("The minimum Value = "+Math.min(m,n));
                System.out.println("The maximum Value = "+Math.max(m,n));
                System.out.println("Square root of 9  = "+Math.sqrt(9));
                System.out.println("Pow(2,3)          = "+Math.pow(2,4));
                System.out.println("exponential of 709.78222656 is
"+Math.exp(709.78222656));
                System.out.println("round(200.675)    = "+Math.round(200.675));
                System.out.println("round(200.675)    = "+Math.round(200.50));
                System.out.println("round(200.675)    = "+Math.round(200.20));
                System.out.println("Absolute Value    = "+Math.abs(-5944));
        }
}
/*OUTPUT
F:\Academic 2022\JavaBatch2022\UNI-I Official>java MathMethods
The minimum Value = 10
The maximum Value = 12
Square root of 9  = 3.0
Pow(2,3)          = 16.0
exponential of 709.78222656 is 1.7968190692375724E308
round(200.675)    = 201
round(200.675)    = 201
round(200.675)    = 200
Absolute Value    = 5944
*/
```

Data Types:
===========

Constants:

====================
Scope of Variables:
====================

- Scope of the variables is nothig but the life time of variables.
- Its scope is depend on where in the program that variables are declared.
- The area of the program where the variable is accessible is called as scope.
- There are three different types of variables present in java

```
1) Instance Variables:
---------------------
- Instance variable is declared inside the class.
- Instance variables are created when the objects are instantiated.
- Instance variables allocate separate memory space when object is created.
- They take different values for each object.

2) Class Variables:
--------------------
- Class variables are declared inside the class.
- They are the global to the class.
- It common between all objects.
- Only one memory location is created for each class variables.

3) Local Variables:
--------------------
- Local Variables declared and used inside the functions.
- The variables which are declared inside the body of methods in known as local
variables.
- They are not available outside the method.
- Local variables can be declared inside the body of methods which is starting from
opening curly braces ({} and closing braces(}).

Example:
class Student
{
        int rollno;                                             //instance variable
        String name;                                    //instance variable
        float marks;                                    //instance variable
        static int college_code=1010;   //class variable
        void calc_marks()
        {
                        int total;              //local variable
        }

}

==================================
Type casting/ Data Type conversion
==================================
- The process of converting one data type to another data type is known as type
casting.
- To change entity of one data type to another data type is known as data type
conversion.
- Tyep casting occurs when we want to store value of one data type into variable of

another type.
- This type casting is required while developing applications.
- If you store large data type value into small data type then it might be data
loss.
```

```
}
```

- If you will store an int value into byte variable then this will be illegal operation.
- To avoid data loss, you should store smaller data type value into larger data type variable.
- Conversion Table:

From	To
byte	short,char,int,long,float,double
short	int,long,float,double
char	int,long,float,double
int	long,float,double
long	float,double
float	double

- There are two types of casting:

1) Implicit Type casting
- The type casting which is done by the system is known as Implicit type casting.
- Example:

```
//Implicit Type casting
class ImplicitTypeCastingDemo
{
                public static void main(String args[])
                {
                                int a=70;
                                float b;
                                b=a;
                                System.out.println("Value of int variable a="+a);
                                System.out.println("Value of float variable b="+b);
                }
}
```

2) Explicit Type Casting
- The type casting which is done by the programmer is known as Explicit type casting.
- Syntax:

```
                        datatype VariableName1=(datatype)VariableName2;
```

- Example:

```
//Explicit Type casting
class ExplicitTypeCastingDemo
{
                public static void main(String args[])
                {
                                int a=70;
                                float b;
                                b=(float)a;
                                System.out.println("Value of int variable a="+a);
                                System.out.println("Value of float variable b="+b);
                }
}
```

Standard Default Values

- Every variable has default value in JAVA.
- If variable is not initialized then java provides default value to that varible automatically.

Type Of Variables	Default Value
byte	zero(0)
short	zero(0)
int	zero(0)
long	zero(0L)
float	0.0f
double	0.0d
char	Null Character
boolean	false

Operators and Expression

- Operator is a symbol which indicate operation to be perform.
- Operands is a variable on which we can perform operation.
- The proper arrangement of operators and operands is known as Expression.
- Following are the classification of operators in JAVA:

1) Arithmetic Operators(+,-,*,/,%)
2) Relational Operators(<,>,<=,>=,==,!=)
3) Logical Operators(&&,||,!)
4) Assignment Operators(=)
5) Increment and decrement Operators(++,--)
6) Conditional Operator(?:) condition?expression1:expression2;
7) Bitwise Operator(&,|,^,<<,>>,~)
8) Special operators(instanceof, dot)

Instanceof Operator:

- This operator return true if the object on the left side is an instance of the class
given on the right side.
- Syntax:

```
        if(object instanceof ClassName)
        {
                //body
        }
```

- Example:

```
//instanceof operator
import java.util.*;
class InstanceOfOpDemo
```

```
{
        public static void main(String args[])
        {
                Scanner sc=new Scanner(System.in);

                if(sc instanceof Scanner)
                {
                                System.out.println("sc is an object of Scanner
class");
                }
        }
}

==============================
***Decision Making Statements:
==============================
1) simple if statement:
-----------------------
- if predefined keyword.
- Syntax:
        if(condition)
        {
                //body of if.
        }

- Example:
        //Write a Java program to check whether two numbers are same or not.
        import java.util.*;
        class IfStatement
        {
                        public static void main(String args[])
                        {
                                        int a,b;

                                        Scanner sc=new Scanner(System.in);
                                        System.out.println("Enter Two Integer
Numbers:");
                                        a=sc.nextInt();
                                        b=sc.nextInt();

                                        if(a==b)
                                        {
                                                System.out.println("Both number are
equal!!!");
                                        }
                        }
        }
        /*
        Enter Two Integer Numbers:
        100
        100
        Both number are equal!!!
        */
```

2) if-else Statement:
=======================
- if and else both are predefined keywords.
- syntax:

```
                        if(condition)
                        {
                                //body of if
                        }
                        else
                        {
                                //body of else
                        }
```

- if condition is true then program controller executes if body otherwise executes else part.
- Example:

```
        //Write a Java program to check whether entered number is even or ODD
        import java.util.*;
        class IfElseStatement
        {
                        public static void main(String args[])
                        {
                                        int no;
                                        Scanner sc=new Scanner(System.in);
                                        System.out.println("Enter Any Integer
Number:");
                                        no=sc.nextInt();

                                        if(no%2==0)
                                        {
                                                System.out.println("Number is
EVEN");
                                        }
                                        else
                                        {
                                                System.out.println("Number is
ODD");
                                        }
                        }
        }
        /*
        Enter Any Integer Number:
        15
        Number is ODD
        */
```

3) Nested if-else statement:

```
-----------------------------
- One if-else within another if is known as nested if-else statement.
- Syntax:
                        if(condition-1)
                        {
                                if(condition-2)
                                {
                                                //body of if
                                }
                                else
                                {
                                                //body of else
                                }
                        }
                        else
                        {
                                //body of else
                        }
- Example:
        //write a Java program to check whether number is positive or negative
        import java.util.*;
        class NestedIfElseStmt
        {
                public static void main(String args[])
                {
                        int no;
                        Scanner sc=new Scanner(System.in);

                        System.out.println("Enter any Integer Number:");
                        no=sc.nextInt();
                        if(no!=0)
                        {
                                if(no>0)
                                {
                                                System.out.println("Number is
Positive!!!");
                                }
                                else
                                {
                                                System.out.println("Number is
Negative!!!");
                                }
                        }
                        else
                        {
                                System.out.println("Zero is neigher Positive nor
Negative");
                        }

                }
        }
        /*
        F:\Academic 2022\JavaBatch2022\UNI-I Official>java NestedIfElseStmt
        Enter any Integer Number:
        143
        Number is Positive!!!

        F:\Academic 2022\JavaBatch2022\UNI-I Official>java NestedIfElseStmt
        Enter any Integer Number:
        -23
        Number is Negative!!!

        F:\Academic 2022\JavaBatch2022\UNI-I Official>java NestedIfElseStmt
        Enter any Integer Number:
        0
        Zero is neigher Positive nor Negative
        */
```

4) else-if ladder
====================
- Suppose, we have multiple conditions but in which only one condition will get true then we can use else if ladder.
- Syntax:

```
                        if(condition-1)
                        {
                                //block of statements
                        }
                        else if(condition-2)
                        {
                                //block of statements
                        }
                        else if(condition-N)
                        {
                                //block of statements
                        }
                        else
                        {
                                //block of statements
                        }
```

- Example:

```
        /*Write a Java program to generate student mark grade on the basis of
following conditions.
        marks>=75 - Distinction
        marks>=60 - First Class
        marks>=40 - Pass
        marks<40  - Fail
        */
        import java.util.*;
        class ElseIfLadder
```

```
	{
		public static void main(String args[])
		{
			int marks;
			Scanner sc=new Scanner(System.in);

			System.out.println("Enter Your Marks:");
			marks=sc.nextInt();

			if(marks>=75)
			{
					System.out.println("Congratulations...You 
got Distinction");
			}
			else if(marks>=60)
			{
					System.out.println("Congratulations...You 
got First Class");
			}
			else if(marks>=40)
			{
					System.out.println("Congratulations...You 
are Pass Only");
			}
			else
			{
					System.out.println("You are Fail!!!");

			}
		}
	}
	/*
	F:\Academic 2022\JavaBatch2022\UNI-I Official>java ElseIfLadder
	Enter Your Marks:
	88
	Congratulations...You got Distinction

	F:\Academic 2022\JavaBatch2022\UNI-I Official>java ElseIfLadder
	Enter Your Marks:
	65
	Congratulations...You got First Class

	F:\Academic 2022\JavaBatch2022\UNI-I Official>java ElseIfLadder
	Enter Your Marks:
	58
	Congratulations...You are Pass Only

	F:\Academic 2022\JavaBatch2022\UNI-I Official>java ElseIfLadder
	Enter Your Marks:
	31
```

```
        You are Fail!!!
        */

switch case statement:
=======================
- switch, case, break and defualt keywords.
- Syntax:
                        switch(expression/value)
                        {
                                case value-1: //block of statements
                                                          break;
                                case value-2: //block of statements
                                                          break;
                                case value-N: //block of statements
                                                          break;
                                default: //block of statements
                        }

Looping Statements:
=========================
1) for loop
2) while loop
3) do while loop
4) Enhanced for loop

For Loop:
---------------
- Syntax:
                for(initialization;condition;incre/decre)
                {
                        //body of for loop
                }
- Example:
        //for loop
        import java.util.*;
        class forloopDemo
        {
                public static void main(String args[])
                {
                        int i;
                        for(i=1;i<=5;i++)
                        {
                                System.out.println("VJTech Academy");
                        }
                }
        }
        /*
        VJTech Academy
        VJTech Academy
        VJTech Academy
        VJTech Academy
        VJTech Academy
        */
```

while loop:
==============
- while is a predefined keyword
- Syntax:

```
                while(Condition)
                {
                        //body of while loop
                }
```

- Example:

```
        //for loop
        import java.util.*;
        class whileloopDemo
        {
                public static void main(String args[])
                {
                        int i=1;
                        while(i<=5)
                        {
                                System.out.println("VJTech Academy");
                                i++;
                        }
                }
        }
        /*
        VJTech Academy
        VJTech Academy
        VJTech Academy
        VJTech Academy
        VJTech Academy
        */
```

do-while loop:
==================
- do & while both are predefined keywords.
- Syntax:

```
                do
                {
                        //body
                }while(condition);
```

- Example

Enhanced for loop/For each loop:
================================
- It is also called as for each loop.
- Using this loop, we can easily retrieve the value of array without using indexes.
- Using for each loop, we can easily iterate over the array.

- Syntax:

```
                for(DataType VariableName:Expression)
                {
                                //statements
                }
```

- Example:

```
//for each loop
class ForEachLoopDemo
{
        public static void main(String args[])
        {
                int num[]={10,20,30,40,50};

                System.out.println("Your Array Elements are:");

                for(int x:num)
                {
                        System.out.println("Value of x :"+x);
                }
        }
}
/*
Your Array Elements are:
Value of x :10
Value of x :20
Value of x :30
Value of x :40
Value of x :50
*/
```