1. # Basics of java and overloading

History of Java

* The history of java starts from Green Team.
* Java team members also known as green team.
* They initiated a revolutionary task to develop a language for digital devices such as set-top boxes, televisions etc.
* for the green team members, it was an advance concept at that time. But, it was suited for interret programming.
* Later, Java technology as incorporated by Netscape.
* James Gosling, Mike sheridan and patrick Naughton initiated the java language project in June 1991.
* The small team of sun engineers called Green Team
* Initially it was designed for small, embeded 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.
* oak is a symbol of strength and chosen

as a national tree of many countries like the U.S.A, France, Germany etc.

* In 1995, oak was renamed as "Java" because it was already a trademark by oak Technologies.
* Initially developed by James Gosling at Sun Microsystems and released in 1995.
* In 1995, Time magazine called "Java one of the ten best products of 1995"
* JDK 1.0 was released on January 23 1996
* After the first release of Java, there have been many additional features are added to the language
* Each new version adds new features in Java.

Java Version History

* Many java versions have been released till now.
* The current stable release of Java is "Java SE 18"

1. JDK Alpha and Beta ⟶ 1995
2. JDK 1.0 ⟶ Jan 1996
3. JDK 1.1 ⟶ Feb 1997
4. J2SE 1.2 ⟶ Dec 1998
5. j2SE 1.3 ⟶ May 2000
6. j2SE 1.4 ⟶ Feb 2002
7. j2SE 5.0 ⟶ Sep 2004
8. Java SE 6 ⟶ Dec 2006
9. Java SE 7 ⟶ July 2011

10. Java SE 8 ⟶ Mar 2014
11. Java SE 9 ⟶ Sept 2017
12. Java SE 10 ⟶ Mar 2018
13. Java SE 11 ⟶ Sept 2018
14. Java SE 12 ⟶ Mar 2019
15. Java SE 13 ⟶ Sept 2019
16. Java SE 14 ⟶ Mar 2020
17. Java SE 15 ⟶ Sept 2020
18. Java SE 16 ⟶ Mar 2021
19. Java SE 17 ⟶ Sept 2021
20. Java SE 18 ⟶ Mar 2022

* Since JavaSE 8 release, the oracle corporation follows a pattern in which every even version is release in March month and odd version released in September month.

Importance of Java in the internet programming

* Java is strongly associated with the internet.
* The first application written in Java was "Hot Java".
* Hot Java is a web browser to run applets on internet.
* Internet users can use Java to create applets and run them locally.
* Remote applets can also be downloaded.
* Java applets have made the internet a true extension of the storage system of a local computer.

Java can be used to create [illegible] of programs

1. Applications
2. Applets

Applications

An application is a program that runs on our computer under the operating system of our computer

Types of Java applications

There are mainly 4 types of applications that can be created using java programming

1 Standalone Application
2 Web application
3 Enterprise Application
4 Mobile Application

Stardalone Application

* It is also known as desktop application or Windows based application
* AWT and Swing are used in Java for creating Standalone application
* Ex: Media player, antivirus

Web Application

* An application that runs on the server side and creates dyramic page is called Web application

* Servlet, jsp, Struts, jsf etc.. technologies are used for creating web applications in Java.
* Ex: Facebook, GMail, canva, Trello.

Enterprise Application

* An application that is distributed in nature, such as banking applications etc.
* It has the advantage of high level security, load balancing and clustering. In java, EJB is used for creating enterprise applications
* Ex: Marketing and Sales Management System.

Mobile Applications

* An application that is created for Mobile devices is called Mobile Application
* Android and Java ME are used for creating Mobile applications
* Ex: Educational apps, Lifestyle apps, Social media apps, productivity apps, Entertainment apps, Game apps.

Applets

*An applet is an application designed to be transmitted over the internet and executed by a Java compatible web browser.
* An applet is actually a tiny java program, dynamically downloaded across the network, just like an image.
* But the difference it is an intelligent program, not just a media file. it can react

to the user input and dynamically change.

Difference between C++ and Java

C++	Java
It was introduced in 1985 by Bjarne Straustrup	It was introduced in 1991 by James Gosling at Sun microsystems.
It is a partial object oriented programming language.	It is a pure object oriented programming language
Headerfiles are supported in C++	Headerfiles are not supported in Java
Destructors are used in C++	In Java destructors are replaced with finalize method.
In C++, we can write code without the class and outside the class with scope resolution operator	In Java, we cannot write code without class. Code must be written within the class.
pointers are supported in C++.	pointers are not supported in Java.
Multiple inheritance is supported in C++	In Java, multiple inheritance is not (thro) supported through class.
Template classes are supported in C++.	Template classes are not supported in Java.

compilation and execution will done by compiler	compilation will done by compiler and execution will done by JVM.
compiled source code will be translated into machine code.	compiled source code will be translated into Byte code.
Friend function is supported in C++	Friend function is not supported in Java.
Write one's compile anywhere	Write one's run anywhere/ anytime
It is a platform dependent language	It is a platform independent language
Multhreading is not supported in C++	Multithreading is supported in Java
Database connectivity is not supported in C++	Database connectivity is supported in Java
Storage classes are Supported in C++	Storage classes are not supported in Java.

Features of Java

Java language supports the following features

i) Compiled and Interpreted
ii) Platform Independent and portable
iii) Object oriented
iv) Robust and Secure
v) Distributed
vi) Simple Small and familiar

multithreaded and interactive
high performance
dynamic and extensible
compiled and interpreted

* Usually a computer language is either compiled or interpreted.
* Java combines both compiled and Interpreted hence, it is called as two stage system.
* First, source code is translated into byte code.
* And then byte code will be translated into machine code by an interpreter.
* It can be directly executed by the machine

platform independent and portable

* The most significant feature of java is its portability.
* Java programs can be easily moved from one computer to another system.

object oriented

Java is a pure object oriented language. It can support all oops concepts like, data abstraction, encapsulation, Inheritance and polymorphism.

Robust and Secure

Java is a robust language
It uses strong memory management
There is a lack of pointers that avoids security problems.
Hence, it is a secure language

Distributed

Java is designed as a distributed language for creating applications on networks.
It has the ability to share both data and programs.
It enables multiple programmers at multiple locations work together on a single project.

Simple Small and familiar

Java is a small and simple language. Many features of c and c++ are not part of java.
Java doesnot provide pointers, preprocessor headerfiles goto statements, operator overloading, Multiple iheritance.
familarity is another feature of java.
Java is a simplified version of c++.

Multithreaded and Interactive

Multithreaded means handling multiple tasks simultaneously
Java can support Multithreaded programs.
Need not wait for the application to finish one task before beginning another.
Greatly improves the interactive performance of graphical applications.

High performance

Java performance is impressive for an interpreted language.
Due to the use of intermediate byte code.
Java architecture is also designed to reduce overheads during run time.

Dynamic and Extensible

Java is a dynamic language

Java is capable of dynamic linking new

Class libraries

Methods

objects

Java can support function written in other languages such as C and C++.

These functions are known as "Native methods"

Byte Code

Java byte code is the instruction set for the java virtual machine.

As soon as a Java program is compiled, java byte code is generated.

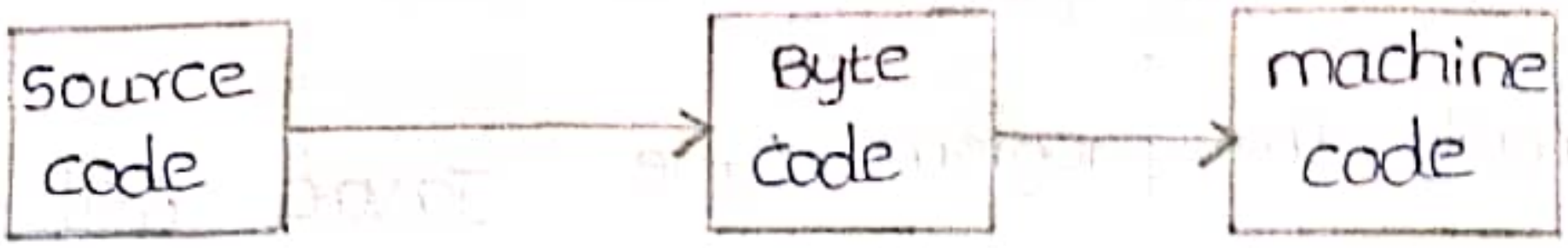
It is defined as an intermediate code which is produced by the compiler during the execution of a Java program.

JVM

A Java virtual Machine (JVM) is an abstract computing machine that enables computer to run a Java program.

JVM is a Platform Independent execution

machine code and executes it.



Steps to Write and execute a Java program

The process of Entering and Executing a Java program.

Entering a Java program

* We can write a Java program using any editor like, Notepad, Notepad++ .. etc.
* We must save the program file with the class name.
* once again ensure that the file name contains the class name.
* This program file is called as source file.
* All Java source files will have the extension .Java
* If a program Contains multiple classes the file name must be the name of the class containing main method.

Executing a Java program

Executing a Java program involves two stages. They are

1. Compiling a Java program

2. Running a Java program

Compiling a Java program

To compile the program type Javac with the name of the source file on the command line.

C:\ Javac Sourcefile.Java (name)

Executing a Java program

To run/execute a Java program we have to type Java along with name of the source file on the command line.

C:\ Java source file name

Ex:

```
class Message
{
   public static void main (String
                            args[])
   {
      System.out.println("Welcome");
   }
}
```

Compiling

C:/ Javac Message.Java

Executing / Running

C:/ Java Message

variables :

variable is a name allocated to the memory location, where the values of datatypes are stored.

It has particular size and value.

Size and value is deperding upon datatype of a variable.

variable declaration

Variable Declaration tells to the compiler about the following three things.

a. variable name

b. Scope of a variable

c. Type of value to be stored in a variable

Syntax :

```
data-type variable-name;
```

variable initialization

It is the process of assigning the values to a variable.

The variable initialization can takes place at two stages.

a. While declaring a variable

b. After the declaration

Syntax :

a. While declaring a variable

```
data-type variable-name = value;
```

b. After the declaration

```
data-type variable_name;
variable_name = value;
```

Scope of a variable

It defines the visibility and accessibility of a variable in a program

In simple words it tells about how long the variable is active in a program.

Types of variables

There are three types of variables in java.

a. Instance variable
b. class variable
c. local variable

Instance variable

An instance variable is a variable that are created when an object is instantiated Instance variables are accessed by creating the object to its class

class variable

A class variable is a variable that is defined in a class of which a single copy exists, regardless of how many instances of the class exist.

Local variable

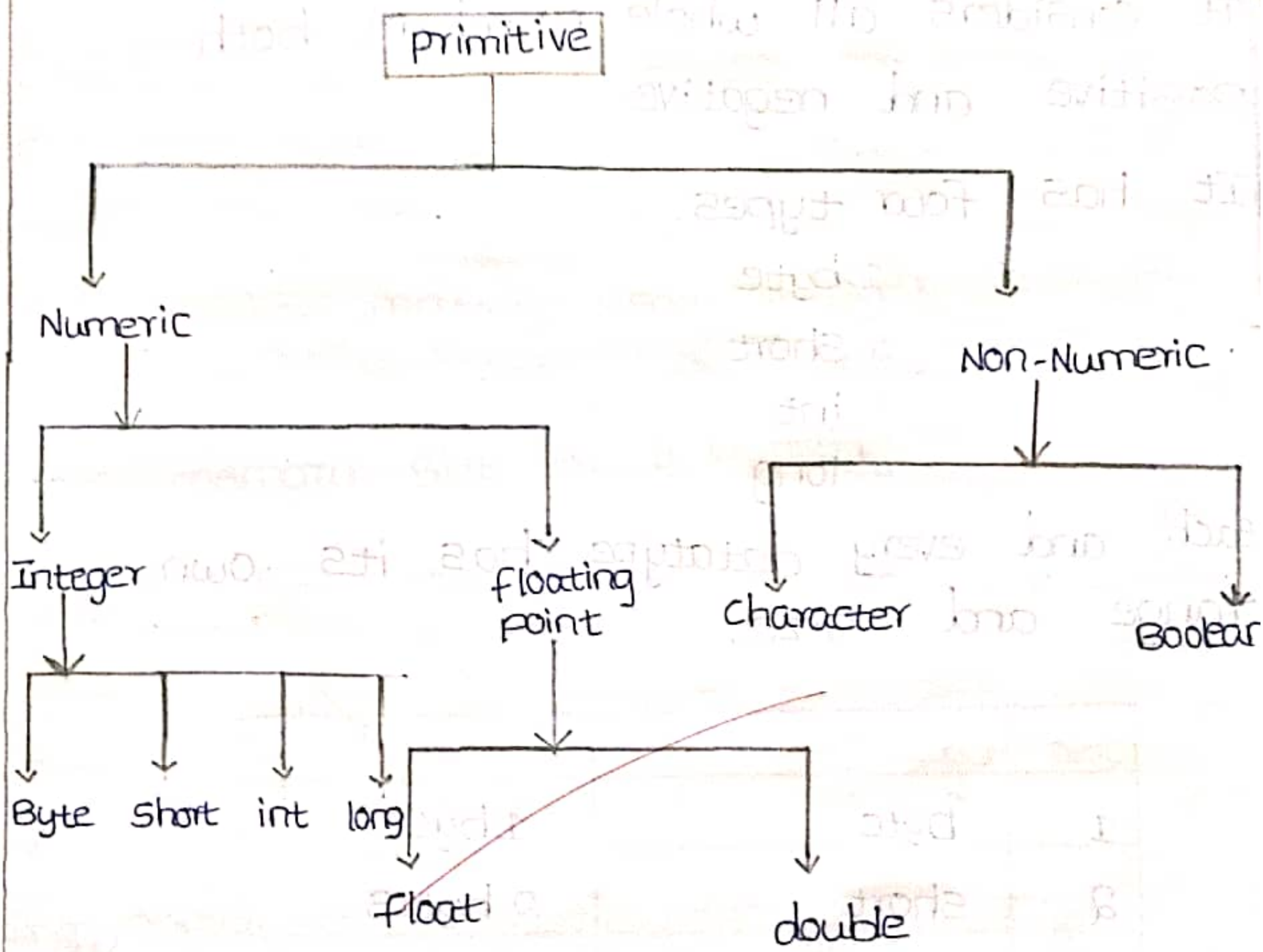
Local variable is a variable that is given

local scope.

Data types

Data type is used to specify the size and type of value to be stored in a variable.

Each (variable data type) is assigned to a Datatype so it can store values according to its own Datatype.



Primitive datatypes are the basic datatypes in Java language.

These datatypes are classified into two types.

1. Numeric
2. Non-numeric,

Numeric

Numeric is used to store numerical values such as Integers and floating point numbers.

It is again divided into two types

1. Integer
2. floating point

Integer

It considers all whole numbers both positive and negative

It has four types.

1. byte
2. short
3. int
4. long

Each and every datatype has its own range and size.

S.NO	Data type	size
1	byte	1 byte
2	short	2 bytes
3	int	4 bytes
4	long	8 bytes

Byte :

The smallest type is byte

It's range from -32,768 to +32,767.

Byte variables are declared by using the

keyword "byte"

Ex: byte x, y

Short :

short is a signed 16 bit type.

It's range from -32,768 to +32,767.

It is probably the least used datatype in Java.

short variables are declared by using the keyword "short"

Ex: short x, y

int :

The most commonly used integer datatype is "int"

It's memory size is 4 bytes

int variables are declared by using the keyword "int"

Ex: int x, y

Long :

Long is a signed 64 bit type.

It is used to store large integer values.

Long variables are declared by using the datatype "long"

Ex: long x, y

floating point

Floating point numbers are also known as real numbers

It is used to store realnumbers.

Real numbers are the numbers that can have a fractional part.

These are subdivided into two types.

1. float
2. double

float :

It is used to store floating point numbers like, 5.4, 7.6 etc--

It's memory size is 4 bytes.

float variables are declared by using the keyword "float"

Ex: float x

Double :

It is used to store large floating point numbers like, 172.27648792 -- etc.

It's memory size of 8 bytes.

double variables are declared by using the keyword "double"

Ex: double interest

S.NO	Datatype	Size
1	float	4 byte
2	Double	8 byte

Non-Numeric

It is used to store Non-numerical values.

It is sub-divided into two types.

1. char
2. Boolean

char

It is used to store character values.

char in Java is different from char in C and C++.

Java uses singlecodes to represent character values.

It's memory size is 2bytes.

char variables are declared by using the keyword "char"

Ex: char name

Boolean

It is used to represent Logical values that can be either true/false.

Relational, conditional and Logical (values) operators return boolean values.

It's memory size is 1bit

Ex: Boolean flag

S.No	Datatype	Size
1	char	2bytes
2	Boolean	1bit

operators in Java

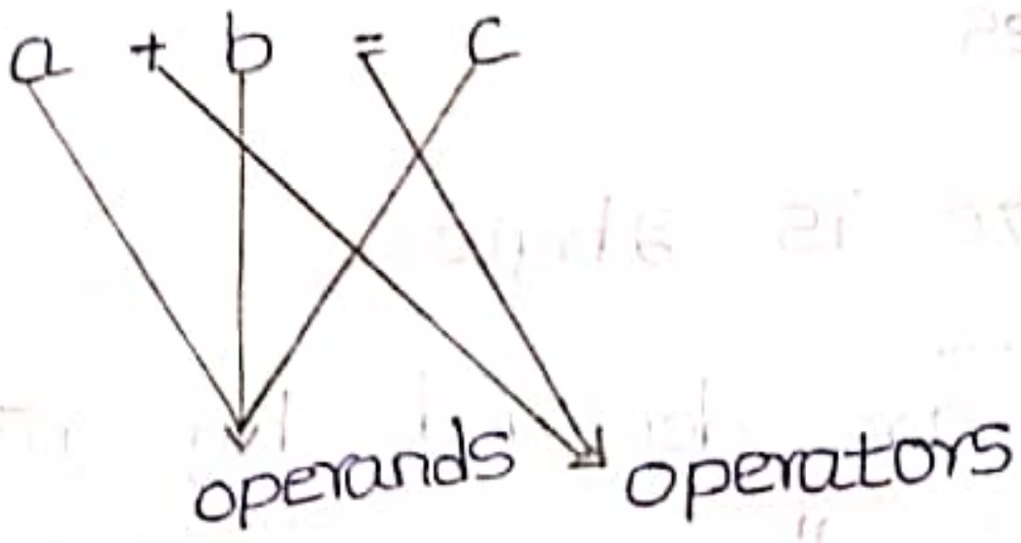
operators are the symbols that performs the Logical and Mathematical operations.

It is used to manipulate the data.

variables are called as "operands"

It tells to the compiler what type of operation should be performed on operands.

Ex :



Types of operators

There are eight types of operators in Java.

1. Arithmetic operators
2. Relational operators
3. Logical operators
4. Assignment operators
5. Increment and Decrement operators
6. Bitwise operators
7. conditional operators
8. special operators

Arithmetic operators

Arithmetic operators are the symbols used to solve the alzebraic expressions

S.NO	operator	use	example
1	+	Adding of 2 operands	a+b
2	-	Subtraction of 2 operands	a-b
3	*	Multiplication of 2 operands	a*b
4	/	Division of 2 operands	a/b
5	%	Modulo-division of 2 operands	a%b

Relational operators

The operator that performs comparision between two operands.

S.NO	operator	use	example
1	<	checks 1st operand < 2nd operand	a<b
2	<=	checks 1st operand <= 2nd operand	a<=b
3	>	checks 1st operand > 2nd operand	a>b
4	>=	checks 1st operand >= 2nd operand	a>=b
5	==	checks 1st operand == 2nd operand	a==b
6	!=	checks 1st operand != 2nd operand	a!=b

Logical operators

The operator that is used to perform Logical operations on two operands

S.NO	operator	use	example
1	&&	& operation on 2 operands	(a<b)&&(b<c)
2	||	|| operation on 2 operands	(a<b)||(b<c)
3	!	Returns T/F based on condition or expression	a=!(b=10)

Assignment operator

The operator that is used to assign some values to the operands.

S.No	operator	shorthand operator
1	x=x+1	x++=1
2	x=x-1	x--=1
3	x=x*(k+1)	x*=(k+1)
4	x=x/(k+1)	x/=(k+1)
5	x=x%(k+1)	x%=(k+1)

Increment / decrement operators

The operator that is (increa) used to increase the operand value by 1 is called increment operator

The operator that is used to decrease the operand value by 1 is called decrement operator.

S.No	operator	use	Example
1	Pre-inc	Increment first	++a
2	Post-inc	stores first Increment next	a++
3	Pre-dec	Decrement first	--a
4	Post-dec	stores first decrement next	a--

conditional operator

conditional operator are also known as "Ternary operator"

It will check the

condition is True then it will return True instruction.

If the condition is false then it will return the false instruction.

Bitwise operators

Bitwise operators are used to perform operations on bits.

S.NO	OPerator	Name
1	&	Bitwise AND
2	|	Bitwise OR
3	^	Bitwise Ex-clusive OR
4	<<	Shift left
5	>>	Shift right
6	~	one's complement

Special operators

It is the combination of all operators.

++ pre, post increment

-- pre, post decrement.

Conversion and casting features

The process of assigning a value of one type to a variable of another type is known as "type casting"

In Java, type casting is classified into two types.

1. Widening
2. Narrowing

Widening casting

Converting a smaller type to a larger type size.

It is also known as Implicit Casting.

This type of casting is takes place when the two types are compatible and the target type is larger than the source type.

byte → short → int → long → float → double

Example

```
class Widening
{
   public static void main(String args
    {
        int i=9;
        double d = i;
        System.out.println(i);
        System.out.println(d);
    }
}
```

Narrowing casting

Converting a Larger type to Smaller type size.

It is also known as explicit type casting

This type of casting must be done manually by placing the type in paranthesis in front of value.

double → float → long → int → short → byte

Example

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

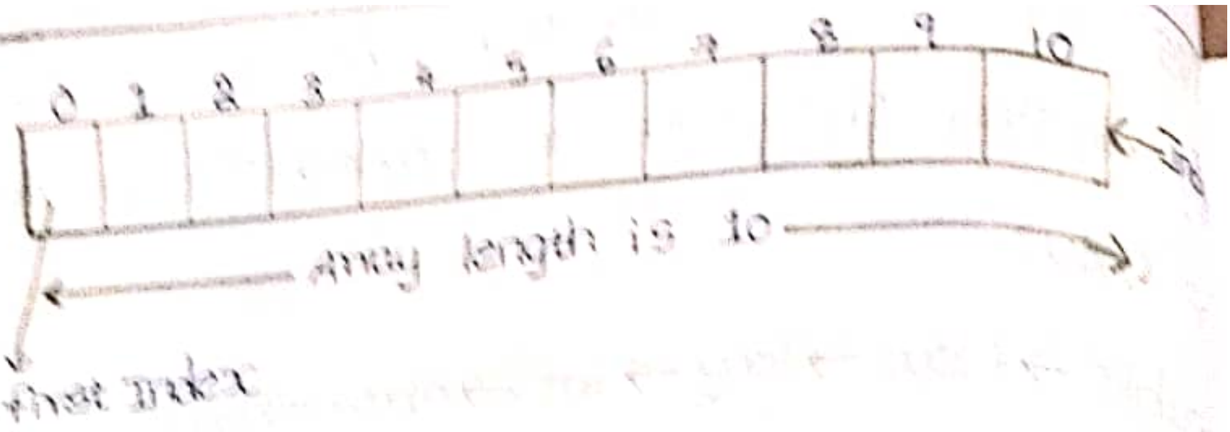
Java Arrays

Array is a collection of elements that have contiguous memory location.

Java array is an object that contains elements of similar datatype.

We can store only fixed set of elements in a Java array

Array in Java is index based, first element of the array is stored at 0 index.



Types of Array

There are two types of arrays in Java

1. Single Dimensional array
2. Two Dimensional array

Single Dimensional array:

* Single Dimensional array is also known as "One Dimensional array"
* One Dimensional array in java is a special type of variable that can store multiple values of only a single datatype such as int, float, double, char etc
* It can store values "contiguous" locations in computer memory.

Syntax:

```
Datatype arrayname [arraysize]
```

Example

```
class onedarray
{
 public static void main (String args[])
 {
    int a[] = { 2, 3, 4, 5};
    int i;
   for (i=0; i<a.length; i++)
```

```
        System.out.println(a[i]);
    }
}
```

Two Dimensional array

* In two dimensional array data is stored in "row" and "column" based index
* It is also known as "Matrix form" of data storage.

Syntax

```
Datatype arrayname [x][y]
```

Example:

```
class Tdarray
{
  public static void main(String args[])
  {
    int arr[][] = {{1,3,5},{2,4,6}};
    int i,j;
    System.out.println("Two dimensional array
                        elements are");
    for(i=0; i<=2; i++)
     {
       for(j=0; j<3; j++)
        {
          System.out.println(arr[i][j]);
        }
          System.out.println();
    }
  }
}
```

Class

* A class is a "group of objects" which have common property
* A collection of objects is called "class"
* It is a logical entity
* A class in java can contain
  1. Fields
  2. Methods
  3. constructors
  4. Blocks
  5. class and Interface

Syntax

```
class classname
{
   fields;
   Methods;
}
```

Example

```
class cls
{
  int id;
 String name;
 public static void main (String args[])
 {
    cls  cs = new cls();
   System.out.println( cs.id);
   System.out.println (cs.name);
 }
}
```

object :

An object is a "real world entity"

An entity that has state and behaviour is called "object"

An object has three characteristics

1. State
2. Behaviour
3. Identity

Syntax

```
classname objectname = new classname();
```

Example

```
class cls
{
 int id;
 String name;
 public static void main (String args[])
 {
     cls cs = new cls(); // Creation of an object
    System.out.println(cs.id); // accessing id with the help of object.
    System.out.println(cs.name); // accessing name with the help of object
 }
}
```

New operator

The new keyword is used to create an "instance of the class"

It instantiates a class by allocating memory

object :

An object is a "real world entity"

An entity that has state and behaviour is called "object"

An object has three characteristics

1. State
2. Behaviour
3. Identity

Syntax

```
classname objectname = new classname();
```

Example

```
class cls
{
 int id;
 String name;
 public static void main (String args[])
 {
      cls cs = new cls(); // Creation of an object
    System.out.println(cs.id); // accessing id with the help of object.
    System.out.println(cs.name); // accessing name with the help of object
 }
}
```

New operator

The new keyword is used to create an "instance of the class"

It instantiates a class by allocating memory

for a new object and returning a reference to that memory.

We can also use the new keyword to create the array object.

Syntax

```
classname obj = new classname();
```

Example

```
class Newoperator
{
  void display()
  {
    System.out.println("Body of the display
                        Method");
  }
  Public static void main(String args[])
  {
   Newoperater op = new Newoperator.();
  (obj) op.display();
  }
}
```

Methods

The method in java is a collection of instructions that performs a specific task It provides reusability of code. We write a method once and use it many times.

We do not require to write code again and again

We can also easily modify code using methods.

A Method must contain the following things

1. return type
2. Method name
3. parameter list
4. Method Body

Example

```
class Mymethod
{
  int add (int a, int b)
   {
      return a+b;
   }
 public static void main (String args[])
   {
      Mymethod Mm = new Mymethod();
      System.out.println(Mm.add( 10, 20));
   }
}
```

Method overloading

The process of having two or more methods with the same name but different in parameter is called "Method overloading"

The method declaration, definition and calling of the function are done with the same name but with different arguments

The advantage method overloading is that it increases "readability" of the program. because you don't need to use different

names for the same action

Types of method overloading

We can overload a method by 3 different ways.

1. By changing parameter
2. By changing Datatype
3. By changing both datatype and parameter

1. By chaning parameter

In this example we created two metho first method performs addition of two num ers, second method performs addition of three (methods) numbers.

Example

```
Class Methodovr
{
  int add (int a, int b)
  {
    return a+b;
  }
  int add (int a, int b, int c)
  {
    return a+b+c;
  }
Public static void main (String args[])
{
  Methodovr MO = new Methodovr ();
System.out.println (MO.add (10, 20));
System.out.println (MO.add (10, 20, 30));
```

```
    }
}
```

By changing Datatype

In this example we created two methods called add first method is created with integer datatype and the second method is created with Double datatype

Example

```
class Methodovr
{
   int add (int a, int b)
    {
       return a+b;
    }
  double add (double a, double b)
    {
       return a+b;
    }
public static void main (String args[])
{
   Methodovr MO = new Methodovr();
  System.out.println (MO.add (10, 20));
  System.out.println (MO.add (10.5, 20.5));
}
}
```

By changing both Datatype and parameter

In this example we created two methods called add. first method has three parameters

and the value of type integer. Second method has two parameter and the value of type double.

Example

```
class Methodovr
{
  int add (int a, int b, int c)
   {
     return a+b+c;
   }
  double add ( double a, double b)
   {
     return a+b;
   }
  Public static void main (String args[])
  {
    Methodovr MO = new Methodovr ();
  System.out.println(MO.add( 10, 20, 30));
  System.out.println(MO.add (10.25, 10.25));
  }
}
```

Constructor

Constructor in java is a special type of method that is used to initialize the object. Constructor has the same name as that of class name.

Java constructor is invoked at the time of object creation.

It should not return a value not even void.

If no constructor is provided for a class compiler invokes default constructor for our program.

Types of constructors

There are two types of constructors in java. They are

1. Default Constructor
2. parameterized constructor

Default Constructor

A Constructor that have no parameter is known as "Default constructor"

Default constructor is provided by the "compiler"

Example

```
class Dconstruct
{
  int id;
  String name;
void display()
 {
  System.out.println(id+" "+name);
 }
 public static void main(String args[])
 {
  Dconstruct dc = new Dconstruct();
  dc.display();
 }
}
```

Parameterized Constructor

A Constructor that have parameters is known as "parameterized constructor"

Parameterized constructor is used to provide different values to the distinct objects.

Example

```
class pconstruct
{
  int id;
  string name;
  pconstruct ( int i, string n)
  {
    id = i;
    name = n;
  }
  void display()
  {
    System.out.println(id + " " + name);
  }
  public static void main (string args[])
  {
    pconstruct pc = new pconstruct ();
    pc.display ();
  }
}
```

Constructor overloading

* Similar to methods it is also possible to overload constructors.

* A class can contain more than one constructor in known as "constructor overloading"
* All constructors are defined with same name as they belong to the class
* All constructors contain different number of arguments
* Depending upon the number of arguments the compiler executes appropriate constructor.

Ex:-

```
class constructor
{
   int id;
   String name;
   int age;
 constructor(int i, String n)
 {
    id = i;
   name = n;
 }
 constructor(int i, String n, int a)
 {
    id = i;
   name = n;
   age = a;
 }
void display()
 {
 System.out.println(id+" "+name+" "+age);
```

```
public static void main (String args[])
{
  Constructor1 c0 = new Constructor1
                    (111, "karan");
  Constructor1 c01 = new constructor1
                    (222, "Aryan", 25);
  c0. display();
  c01. display();
}
}
```

Differences between the constructor and Method.

	constructor	Method
1.	constructor is used to initialize the state of an object	Method is used to expose the behaviour of an object.
2.	constructor must not have any return type	Method must have return type
3.	constructor is invoked implicitly	Method is invoked implicitly
4.	The java compiler provides a default constructor	The java compiler doesnot provides a default method.
5.	constructor name is same as the class name	Method name may or may not be same as class name.

This pointer

In java, "this" is a "reference variable" that refers to the current object.

This keyword can be used to refer current class instance variable.

"This" can be used to invoke current class constructor.

This keyword can be used to invoke current class method.

It can be passed as an argument in the "Method call"

It can be passed as an argument in the "constructor call"

Example

```
Class Thispoint
{
   int id;
   String name;
Thispoint( int id, String name)
 {
   this.id = id;
  this.name = name;
}
 void display()
  {
   System.out.println(id+" "+name);
  }
   Public Static void main(String args[])
```

```
{
  Thispoint tp = new Thispointer(111, "karan");
  tp.display();
}
}
```

Static keyword

The "static keyword" in java is used for memory management mainly

We can apply "static" keyword with variables, methods, blocks, and nested class.

Static variable

If we declare any variable as "static" it is known as static variables

The static variable can be used to refer the common property of all objects.

Static variable is accessed without Creating any object.

Example

```
class StaticVbl
{
  int rollno;
  String name;
 Static String College = "ITS";
StaticVbl (int r, String n)
{
```

```
    rollno = r;
    name = n;
}
void display()
 {
   System.out.println(rollno+" "+name+" "+
                      college);
 }
public static void main(String args[])
 {
   StaticVbl sv = new StaticVbl(111,
                                "karan");
   sv.display();
 }
}
```

Static Method

If we apply "Static" keyword with any method it is known as "Static Method"

Static method can be accessed without Creating the object.

Example

```
class StaticMrd
{
   int rollno;
   String name;
  static String college = "ITS";
static void change()
 {
   college = "GPT";
```

```
}
Static md (int r, String n)
{
   rollno = r;
   name = n;
}
void display ()
{ System.out.println (rollno + " " + name
                      + " " + college);
}
public static void main (String args[])
{
 Staticmd sm = new staticmd ( 222,
                              "Aryan");
 sm.display();
 }
}
```

Static Block

Static block is used to initialize the static data member.

It is executed before the main method

It is executed at the time of class loading.

Example

```
Class staticblk
{
 static
```

```
{
 System.out.println(" static block");
}
public static void main (String args[])
{
  System.out.println(" Main method");
}
}
```

Strings

String is the most commonly used class in java library.

String is encapsulated under java.lang package. The java platform provides the string class to create and manipulate strings.

"strings" are used to store group of characters.

Example

```
class stringexmp
{
 public static void main(String args[])
 {
  String s1 = " java";
  char ch[] = {'s', 't', 'r', 'i', 'n', 'g', 's'}
 String s2 = new String(ch);
 String s3 = new String("example");
 System.out.println(s1);
```

```
System.out.println(s2);
System.out.println(s3);
 }
}
```

String Methods

1. charAt(int index):

It returns a char value for the particular index.

```
String s="Java"
 0123
s.charAt(2)
```

It returns v

2. length():

It returns length of the String

```
String s="Good"
s.length()
```

It returns 4

Note:

While finding the string length it starts from 1st index not from 0th index.

3. toUpperCase():

It returns a string in uppercase

4. toLowerCase():

It returns a string in Lowercase

5. Index of():

It returns a index value for the

specified character.

6. lastIndex of():

It returns a last index value for the specified character that means if a character is two or more times repeated in a string in that situation this method return the index of last repeated character.

```
S1 = "Java"
S1.lastindex of("a")
```

It returns 3

7. Contains():

It checks whether the specified character is present in a string or not if present it returns a boolean value "True" else it returns "false"

8. endswith():

It checks whether the string is ended with a specified character or not.

If it is ended with a specified character it returns boolean value "True" else it returns "false"

9. Concat():

It is used to combine two strings.

10. equals():

It Compare two strings and checks it is equal or not and it return boolean

value.

11. equalsIgnoreCase():

It checks two strings is equal or not by Ignoring the case.Sensitive.

12. trim():

It removes spaces from a String

13. Empty():

It checks whether the string is empty or not.

String Buffer

A string buffer is like a string, but it can be modified

At any point in time it contains some particular sequence of characters but the length and content of the sequence can be changed through certain Method calls.

StringBuffer methods are:

1. append
2. Substring
3. insert
4. delete
5. replace
6. reverse

1. append:

It is used to combine some extra character or string to the existing string

2. Substring:

It is used to get some required

characters from the existing String

3. insert :
It is used to insert a character or string in the font of the existing string, middle or end of an existing string

4. delete :
It is used to delete a character or string.

5. replace :
It is used to replace a character or string.

6. reverse :
It is used to reverse a string.

StringBuilder

StringBuilder in java is a class used to create a mutuable or in otherwords a modifiable succession of character.

It is more efficient than StringBuffer

It is non-synchronized it means two threads can call the method of String Builder, Simultaneously.

Command Line arguments

The java command-line argument is an argument Passed at the time of running the java

The argument passed from the console can be recieved in the java program and it can be used as an input.

It provides a convenient way to check the behavior of the program for the different values.

Example

```
class CommandLine1
{
  public static void main(String args
    {
      System.out.println("Your first
                          argument is:" +
                          args[0]);
    }
}
```

Final keyword

The final keyword is used to "restrict" the user.

It is used to make a variable as constant, Restrict method overriding, Restrict inheritance

The final keyword can be used in the following ways:

1. Final variable
2. Final Method
3. Final class

Final variable

A variable is declared with the final keyword is known as "final variable"

It may be member variable or local variable.

Final keyword is used to make a variable as "constant"

A variable declared with "final" keyword cannot be modified by the program after initialization.

Example

```
Class Circle
{
  Public  static final double
          Pi = 3.14159
 Public   static void main (String args[])
 {
    System.out.println(pi);
 }

}
```

Final Method

When a method is declared with final keyword, it is called "final Method"

All final methods cannot be overriden.

When we want to restrict overriding then make a method as final.

Example

```
class p
 {
  final void demo()
   {
    System.out.println("class p Method");
   }
}
class q extends p
 {
  void demo()
   {
    System.out.println("class q method");
   }
}
Public class fm extends q
{
  public static void main(String args[])
  {
   q obj = new q();
   obj.demo();
  }
```

nal class

When a class is declared with the final keyword then it is called "final class".

final class can not be inherited by other classes.

We cannot extend a final class.

When we want to restrict inheritance then make the class as final.

Example

```
final class Car
{
  System.out.println("Travelling in Car");
}

class Benz extends Car
{
  void run()
   {
     System.out.println("running safely with
                         100kmph in benz car");
   }
  public static void main(String args[])
   {
     Benz ben = new Benz();
     ben.run();
   }
}
```

14/10/22

Creation of String object

There are two ways to Create String object:

1. By String literal
2. By new keyword

String literal

Java string literal is created by using double quotes.

Example

```
String s = "Welcome";
```

By new keyword

We can also create the string object by using new keyword

Example

```
String s = new String ("Welcome");
Char ch[] = {'H', 'E', 'L', 'L', 'O'}
String s = new String (ch);
```