Topics to be covered:
https://docs.google.com/document/d/1exK_GQvNhKOQHHtoDdwCVu8MqdMOYg9Rgk6tvIEztp4/edit

Program practice:
https://docs.google.com/document/d/1wdT9pGs8tH_ak_QPuN_pNDPXJTS2hZ8uw8mItPHqe0o/edit?usp=sharing

https://docs.google.com/document/d/1jgRbZW9fjbD_KoWD3W2WI3ZYTBYVwTmKAIWnNf7-9Kw/edit

DAY1

What is a programming language?
A **programming language** is a set of commands, instructions, and other syntax used to create a software program.

Languages that programmers use to write code are called "high-level languages." This code can be compiled into a "low-level language," which is recognized directly by the computer hardware.

The main types of programming languages are:

1. Procedural **Programming Language**.----C
2. Functional **Programming Language**.------Lisp. Haskell
3. Object-oriented **Programming Language**.---JAVA, C++
4. Scripting **Programming Language**.---SQL(Database)
5. Logic **Programming Language**.-------PROLOG

JAVA→ high level, low level

a. **What is JAVA?**

Java is a **programming language** and a **platform**. Java is a high level, robust, object-oriented and secure programming language and developed by *Sun Microsystems* (which is now the subsidiary of Oracle) in the year 1995

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.
API-> application programing interface

b. **History of JAVA**
The principles for creating Java programming were "Simple, Robust, Portable, Platform-independent, Secured, High Performance, Multithreaded, Architecture

Neutral, Object-Oriented, Interpreted, and Dynamic". Java was developed by James Gosling, who is known as the father of Java, in 1995

Currently, Java is used in internet programming, mobile devices, games, e-business solutions, etc. There are given significant points that describe the history of Java.
Initially developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995.

c. JDK, JRE, JVM---- https://youtu.be/7tndaxgk1E8?t=2

JVM

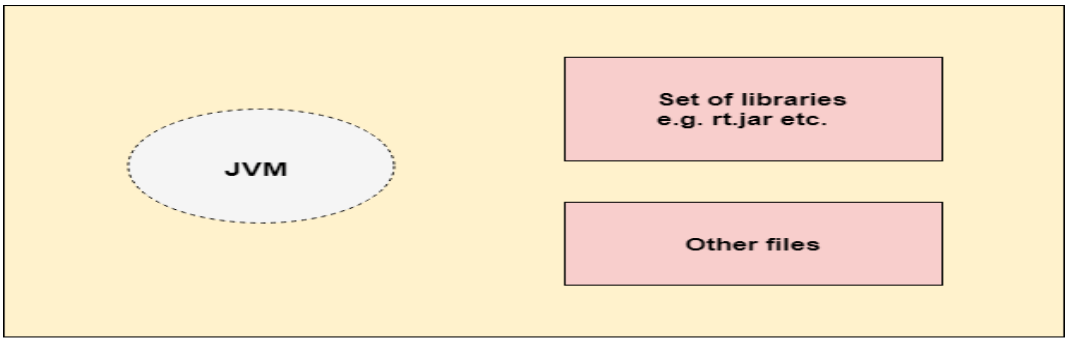
JVM (Java Virtual Machine) is an abstract machine. It is called a virtual machine because it doesn't physically exist. It is a specification that provides a runtime environment in which Java bytecode can be executed. It can also run those programs which are written in other languages and compiled to Java bytecode

The JVM performs the following main tasks:

- Loads code
- Verifies code
- Executes code
- Provides runtime environment

JRE

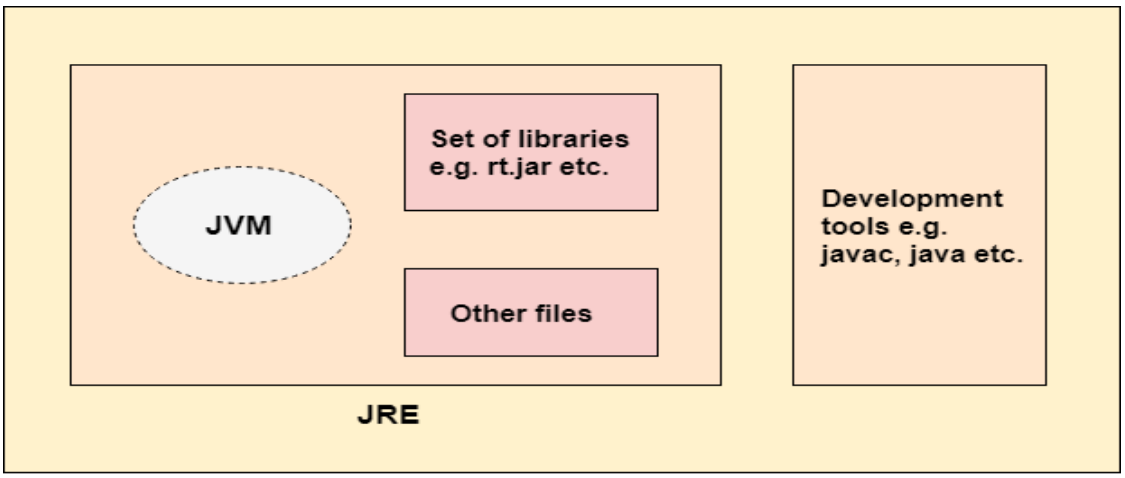
JRE is an acronym for Java Runtime Environment. It is also written as Java RTE. The Java Runtime Environment is a set of software tools which are used for developing Java applications. It is used to provide the runtime environment. It is the implementation of JVM. It physically exists. It contains a set of libraries + other files that JVM uses at runtime.



JRE

JDK

JDK is an acronym for Java Development Kit. The Java Development Kit (JDK) is a software development environment which is used to develop Java applications and applets. It physically exists. It contains JRE + development tools.



JDK

How to check java installed in system: java -version

SETUP JAVA IN SYSTEM ----- System dependent
Install IDE (Integrated development environment)------> {eclipse, Intellij etc..}
Text editors: Notepad, Notepad++, Sublime etc..

d. Hello World

```
class Simple{                                   // creating a class with name
  public static void main(String args[]){       // main method
   System.out.println("Hello Java");            // print statement
   }                                            // method close
}                                               // class close
/*  this is a multi line comment
*  above program is used to print hello world, Simple is a class name
```

```
*  this is a java file, after compiling .class file will be created which have *  byte
*   code in it.
*/
```

Java file contains your Java source code while a . class file contains the Java bytecode produced by the Java compiler. ... class files that run on the JVM to execute a Java application. It is the . class files you will use when you deploy your applet.

Day2

Class and objects:

A **class** is a template for **objects**. ... An **object** is a member or an "instance" of a **class**. An **object** has a state in which all of its properties have values that you either explicitly define or that are defined by default settings.

Refer link: https://www.javatpoint.com/object-and-class-in-java

```
class Book{
static String libarayName;
  String authorName;         // data variable

  void setAuthorName(){  /// function
     String member;
   }
  public static void main(String a[]){
        Book obj1 = new Book();      // creating object for class to tasks ---authorName
        Book obj2 = new Book(); //-----authorName
        Book obj3 = new Book();//----authorName
    }
```

}

Book1: Obj1: authorName, setAuthorName();
Book2: Obj2: authorName, setAuthorName();
Book3: Obj3: authorName, setAuthorName();

e. JAVA Variables

A variable is a container which holds the value while the Java program is executed. A variable is assigned with a data type.

There are three types of variables in java: local, instance and static.

There are two types of data types in Java: primitive and non-primitive.

1) Local Variable

A variable declared inside the body of the method is called local variable. You can use this variable only within that method and the other methods in the class aren't even aware that the variable exists.

A local variable cannot be defined with "static" keyword.

2) Instance Variable

A variable declared inside the class but outside the body of the method, is called instance variable. It is not declared as static.

It is called instance variable because its value is instance specific and is not shared among instances.

3) Static variable

A variable which is declared as static is called static variable. It cannot be local. You can create a single copy of static variable and share among all the instances of the class. Memory allocation for static variable happens only once when the class is loaded in the memory.

```
class Library{
        static String libraryName = "TCLP";//static variable
```

```
        String memberName="abc";//instance variable
        void printMemberName(){
                int number=90;//local variable
        }

        Public static void main(String a[]){
           Library lib1 = new Library();
           Library lib2 = new Library();
        Lib2.membername = "def"
         }
}//end of class
```

Lib1: libraryName="TCLP", memberName="abc"

Lib2: libraryName="TCLP", membername="def"

Examples:

1. **Type casting: explicit typecasting, implicit type casting**

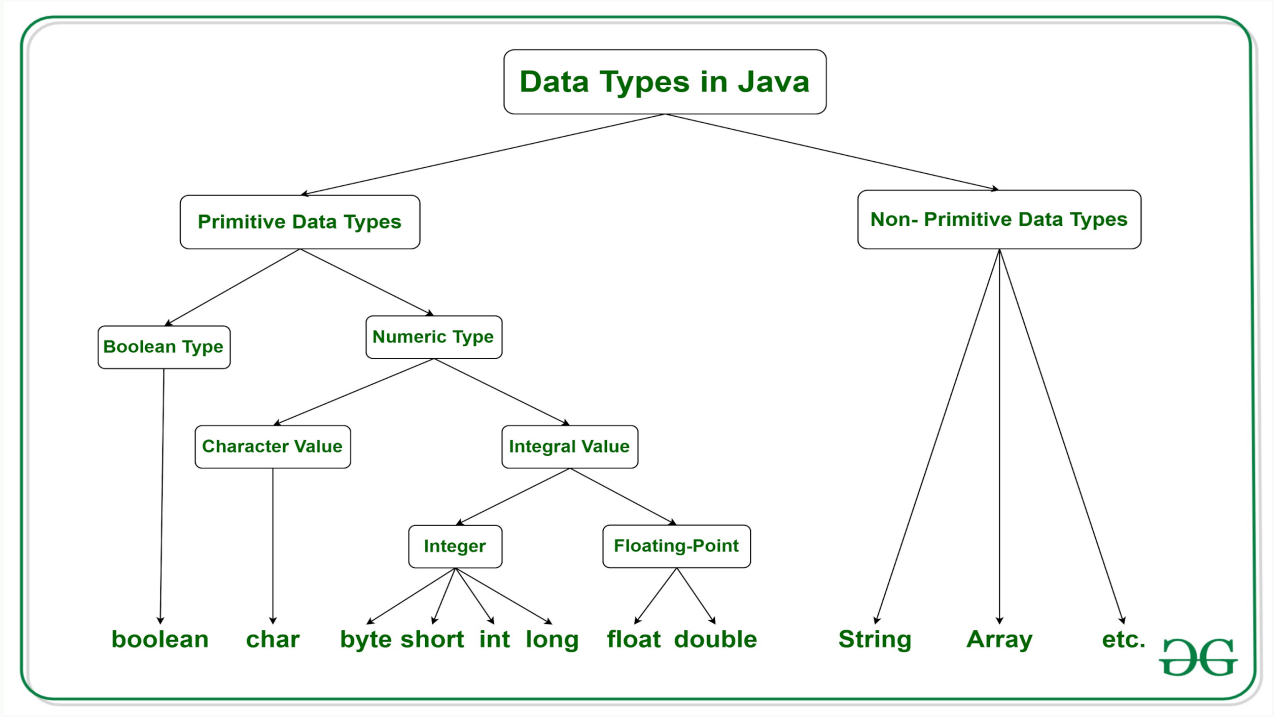
```
 class Simple{

        public static void main(String[] args){
        float f=10.5f;
        //int a=f;//Compile time error
        int a=(int)f;
        System.out.println(f);
        System.out.println(a);
        }
}
```

Output:
10.5
10

Data Types in Java

Data types specify the different sizes and values that can be stored in the variable. There are two types of data types in Java:



TYPE	DESCRIPTION	DEFAULT	SIZE	EXAMPLE LITERALS	RANGE OF VALUES
boolean	true or false	false	1 bit	true, false	true, false
byte	twos complement integer	0	8 bits	(none)	-128 to 127
char	unicode character	\u0000	16 bits	'a', '\u0041', '\101', '\\', '\'','\n',' β'	character representation of ASCII values 0 to 255
short	twos complement integer	0	16 bits	(none)	-32,768 to 32,767
int	twos complement integer	0	32 bits	-2, -1, 0, 1, 2	-2,147,483,648 to 2,147,483,647
long	twos complement integer	0	64 bits	-2L, -1L, 0L, 1L, 2L	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	IEEE 754 floating point	0.0	32 bits	1.23e100f, -1.23e-100f, .3f, 3.14F	upto 7 decimal digits
double	IEEE 754 floating point	0.0	64 bits	1.23456e300d, -1.23456e-300d, 1e1d	upto 16 decimal digits

Operators and Keywords, Packages

Operator in Java is a symbol which is used to perform operations. For example: +, -, *, / etc.

https://www.w3schools.com/java/java_operators.asp

Binary number: 01, 10, 11, 100, 101, 110…..1010
10/2 = remainder is 0 and 5/2=1, 2/2 =0, 1

Convert integer to binary and vice-versa.

2*2=4, 2/2=1

Left shift and right shift operator

Left Shift---the Java left shift operator << is used to shift all of the bits in a value to the left side of a specified number of times.

```
class OperatorExample{
    public static void main(String args[]){
    2 raise to power 2 = 4

    System.out.println(10<<2);//10*2^2=10*4=40
```

```
        System.out.println(10<<3);//10*2^3=10*8=80
        System.out.println(20<<2);//20*2^2=20*4=80
        System.out.println(15<<4);//15*2^4=15*16=240
        }
}
```

JAVA RIGHT SHIFT:

The Java right shift operator >> is used to move left operands value to right by the number of bits specified by the right operand

```
class OperatorExample{
        public static void main(String args[]){
        System.out.println(10>>2);//10/2^2=10/4=2
        System.out.println(20>>2);//20/2^2=20/4=5
        System.out.println(20>>3);//20/2^3=20/8=2
        }
}
```

KEYWORDS

Java keywords are also known as reserved words.
Eg: static, int, private, public, class, abstract, final, void, etc….

What are different packages in Java?

A **java package** is a group of similar types of classes, interfaces and sub-**packages**. **Package in java** can be categorized in two form, built-in **package** and user-defined **package**. There are many built-in **packages** such as **java**, lang, awt, javax, swing, net, io, util, sql etc.

Java If-else Statement

The Java *if statement* is used to test the condition. It checks boolean condition: *true* or *false*. There are various types of if statement in Java.

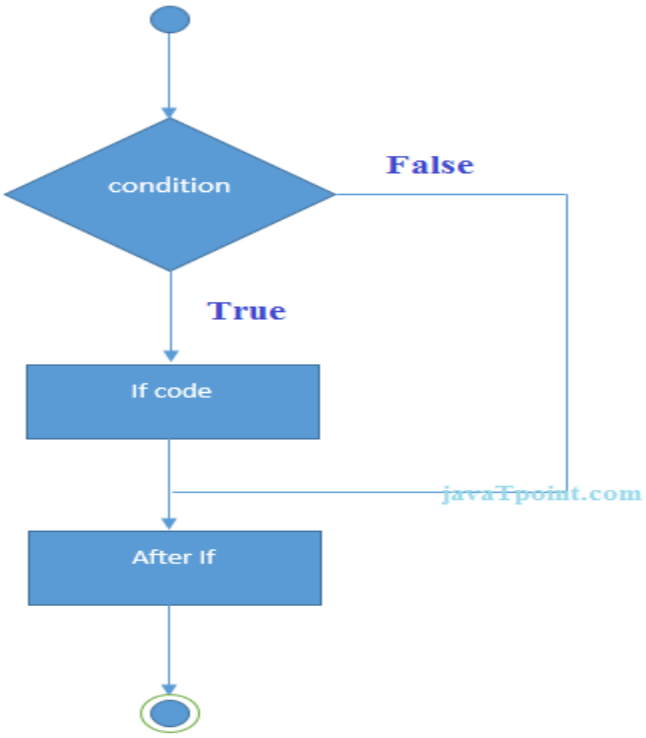
- if statement
- if-else statement
- if-else-if ladder

- nested if statement

Java if Statement

The Java if statement tests the condition. It executes the *if block* if condition is true.

```
if(condition){
//code to be executed
}
```



Or (||) and (&&) xor(^)

Or operator

A	B	A||B
1	0	1
0	0	0
1	1	1

And operator

A	B	A&&B
1	0	0
0	0	0
1	1	1

```
public class IfElseIfExample {
public static void main(String[] args) {
    int marks=65;
      //type1    if-else block
         if(marks<75){

         }else{

         }
//type2      if-else ladder
    if(marks<50){
        System.out.println("fail");
    }
    else if(marks>=50 && marks<60){
        System.out.println("D grade");
    }
    else if(marks>=60 && marks<70){
        System.out.println("C grade");
    }
    else if(marks>=70 && marks<80){
        System.out.println("B grade");
    }
    else if(marks>=80 && marks<90){
        System.out.println("A grade");
    }else if(marks>=90 && marks<100){
        System.out.println("A+ grade");
    }else{
        System.out.println("Invalid!");
    }
}
}
```

DAY 3:

1. Creating a sample class book with details -- bookName, authorName, pageNumber;
2. Naming conventions
3. Constructors logic
4. How to take input from the user.

```
int, float, String
```

```
1. class name is alaways starting with capital letter
2. name: camel casing--> variable name and method name;
3.
```

1. **Constructors: 3 types**
   Simple, parameterized

```
public class Book {
    String bookName;
    int pageNumber;
    String authorName;

    //constructors

    Book(){

        //empty

    }
public void test(Book bookObj1, Book obj2){
    this.bookName = "new book";
    System.out.println(this.bookName);
    System.out.println(this);

    if(this.bookName == bookObj1.bookName){
```

```
    }

    if(obj2.bookName == bookObj1.bookName){

    }
}

    Book(String newBookName, String newAuthorName, int newPageNumber){
        this.bookName = newBookName;
        this.authorName = newAuthorName;
        this.pageNumber = newPageNumber;
    }


    public static void main(String a[]){
        Book book1 = new Book();
        Book book2 = new Book("Harry Potter ", "J k Rolling " , 200);
        Book book3 = new Book("Harry Potter1 ", "J k Rolling1 ", 400);
        System.out.print(book2.bookName);
        System.out.print(book2.authorName);
        System.out.print(book2.pageNumber);

        System.out.println();

        System.out.print(book3.bookName);
        System.out.print(book3.authorName);
        System.out.print(book3.pageNumber);
       System.out.println();

       System.out.println(book3);

       book3.test(book1, book2);

    }
}
```

https://www.youtube.com/watch?v=HD5lyaOgdPQ

SCANNER INPUT:

Scanner class in Java is found in the java.util package. Java provides various ways to read input from the keyboard, the java.util.Scanner class is one of them.

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

```
import java.util.Scanner;  // Import the Scanner class
```

```
class Main {
  public static void main(String[] args) {
Scanner myObj = new Scanner(System.in);  // Create a Scanner object
    System.out.println("Enter username");

    String userName = myObj.nextLine();  // Read user input
    System.out.println("Username is: " + userName);  // Output
user input
  }
}
```

```
import java.util.*;
public class ScannerClassExample1 {
    public static void main(String args[]){
        String s = "Hello, This is JavaTpoint.";
        //Create scanner Object and pass string in it
        Scanner scan = new Scanner(s);
        //Check if the scanner has a token
        System.out.println("Boolean Result: " + scan.hasNext());
        //Print the string
        System.out.println("String: " +scan.nextLine());
        scan.close();
        System.out.println("--------Enter Your Details-------- ");
        Scanner in = new Scanner(System.in);
        System.out.print("Enter your name: ");
        String name = in.next();
        System.out.println("Name: " + name);
        System.out.print("Enter your age: ");
        int i = in.nextInt();
        System.out.println("Age: " + i);
        System.out.print("Enter your salary: ");
        double d = in.nextDouble();
        System.out.println("Salary: " + d);
        in.close();
        }
}
```

LOOPS:

For, while, do-while

Comparison	for loop	while loop	do while loop
Introduction	**The Java for loop is a control flow statement that iterates a part of the programs multiple times.**	**The Java while loop is a control flow statement that executes a part of the programs repeatedly on the basis of given boolean condition.**	**The Java do while loop is a control flow statement that executes a part of the programs at least once and the further execution depends upon the given boolean condition.**
When to use	**If the number of iteration is fixed, it is recommend**	**If the number of iteration is not fixed, it is recommended**	**If the number of iteration is not fixed and you must have to execute the loop at least once, it is**

	ed to use for loop.	**to use while loop.**	**recommended to use the do-while loop.**

Syntax	**for(init;condition;incr/decr){
	// code to be executed
	}**
		**while(condition){
		//code to be executed
		}**
			**do{
			//code to be executed
			}while(condition);**

Example	**//for loop
	for(int i=1;i<=10;i++){
	System.out.println(i);
	}**
		**//while loop
		int i=1;
		while(i<=10){
		System.out.println(i);
		i++;
		}**
			**//do-while loop
			int i=1;
			do{
			System.out.println(i);
			i++;
			}while(i<=10);**

Syntax for infinite loop	**for(;;){
	//code to be executed
	}**
		**while(true){
		//code to be executed
		}**
			**do{
			//code to be executed
			}while(true);**

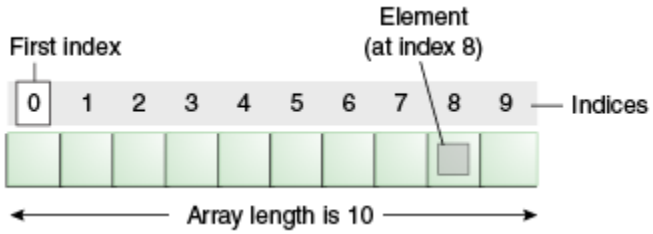
DAY4:

Arrays, continue, break

Normally, an array is a collection of similar type of elements which has contiguous memory location.

Java array is an object which contains **elements of a similar data type.** Additionally, The elements of an array are stored in a contiguous memory location. It is a data structure where we store similar elements. We can store only a fixed set of elements in a Java array.

Array in Java is index-based, the first element of the array is stored at the 0th index, 2nd element is stored on 1st index and so on.



Types of Array:

There are two types of array.

- Single Dimensional Array--- just a single row
- Multidimensional Array ----- column and row

Single dimension array

Syntax to Declare an Array in Java

1. dataType[] arr; (or)
2. dataType []arr; (or)
3. dataType arr[];

Instantiation of an Array in Java

example:

int ref=**new** int[size];

```
int name=new int[10];
String array=new String[10];

Book bookList = new Book[10];
```

Example:

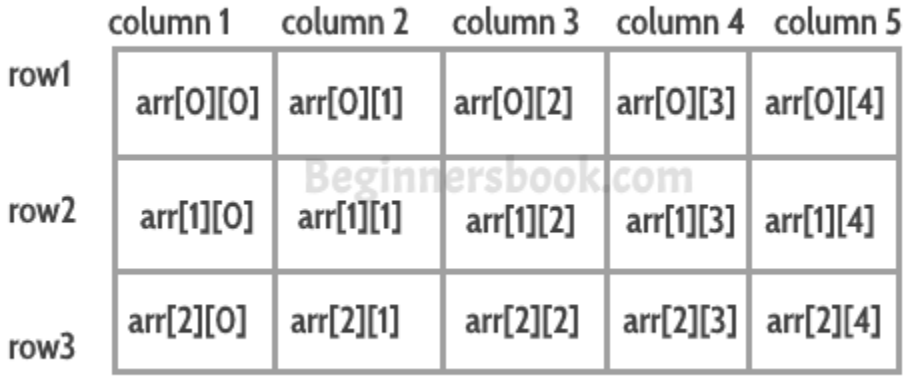
```
class Testarray{
        public static void main(String args[]){
        int a[]=new int[5];//declaration and instantiation
        a[0]=10;//initialization
        a[1]=20;
        a[2]=70;
        a[3]=40;
        a[4]=50;
        //traversing array
        for(int i=0;i<a.length;i++)//length is the property of array
        System.out.println(a[i]);
}}
```

a: 10, 20,70,40,50

```
System.out.println(a[3]);
```

Pre defined values in array: `int a[]={33,3,4,5};`

Multi dimensional array:



In such case, data is stored in row and column based index (also known as matrix form).

Syntax to Declare Multidimensional Array in Java

1. dataType[][] arrayRefVar; (or)
2. dataType [][]arrayRefVar; (or)
3. dataType arrayRefVar[][]; (or)
4. dataType []arrayRefVar[];

Instantiation

```
int[][] arr=new int[3][3];//3 row and 3 column
String[][] stringArr =  new String[2][2];
```

Predefining values in 2d array:

1. `int a[][]={{1,3,4},{3,4,5}};`

	Col1	Col2	Col3
Row0:	1(0,0)	3(0,1)	4(0,2)
Row1:	3(1,0)	4(1,1)	5(1,2)

2. `int b[][]={{1,3,4},{3,4,5}};`

Example

```
public class ArrayClass {

public static void main(String args[]) {
//declaring and initializing 2D array
       int arr[][] = {{1, 2, 3}, {2, 4, 5}, {4, 4, 5}};
//printing 2D array
       for (int i = 0; i < 3; i++) {
           for (int j = 0; j < 3; j++) {
               System.out.print("arr[" + i+"]["+j+"] "+arr[i][j] + " ");
           }
           System.out.println();
       }

       System.out.println();
       for (int i = 0; i < 3; i++) {
           for (int j = 0; j < 3; j++) {
               System.out.print("arr[" + j+"]["+i+"] "+arr[j][i] + " ");
           }
           System.out.println();
       }
   }
}
```

DAY 5

OOPS CONCEPT

Object means a real-world entity such as a pen, chair, table, computer, watch, etc. Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects. It simplifies software development and maintenance by providing some concepts:

- **Object**
- **Class**
- **Inheritance**
- **Polymorphism**
- **Abstraction**
- **Encapsulation**

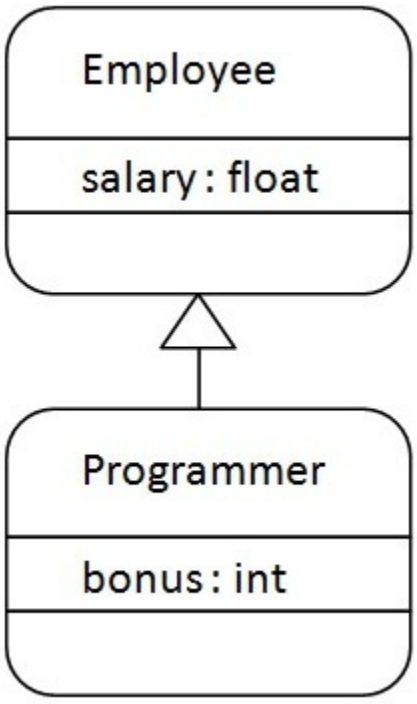
Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object. It is an important part of OOPs (Object Oriented programming system).

The idea behind inheritance in Java is that you can create new classes that are built upon existing classes. When you inherit from an existing class, you can reuse methods and fields of the parent class. Moreover, you can add new methods and fields in your current class also.

Inheritance represents the IS-A relationship which is also known as a *parent-child* relationship.

```
class Subclass-name extends Superclass-name
{
  //methods and fields
}
```

The extends keyword indicates that you are making a new class that derives from an existing class. The meaning of "extends" is to increase the functionality.




```
class Employee{
 float salary=40000;
}
class Programmer extends Employee{
```

```
 int bonus=10000;
 public static void main(String args[]){
   Programmer p=new Programmer();
   System.out.println("Programmer salary is:"+p.salary);
   System.out.println("Bonus of Programmer is:"+p.bonus);
}
}
```

Polymorphism in Java is a concept by which we can perform a *single action in different ways*. Polymorphism is derived from 2 Greek words: poly and morphs. The word "poly" means many and "morphs" means forms. So polymorphism means many forms.

There are two types of polymorphism in Java: compile-time polymorphism and runtime polymorphism. We can perform polymorphism in java by method overloading and method overriding.

https://www.javatpoint.com/runtime-polymorphism-in-java

Abstraction is a process of hiding the implementation details and showing only functionality to the user.

There are two ways to achieve abstraction in java

1. Abstract class (0 to 100%)
2. Interface (100%)

Abstract class in Java

A class which is declared as abstract is known as an **abstract class**. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.

Points to Remember

- An abstract class must be declared with an abstract keyword.

- It can have abstract and non-abstract methods.
- It cannot be instantiated.
- It can have constructors and static methods also.
- It can have final methods which will force the subclass not to change the body of the method.

```
abstract class Bike{
  abstract void run();
}
class Honda4 extends Bike{
void run(){System.out.println("running safely");}
public static void main(String args[]){
 Bike obj = new Honda4();
 obj.run();
}
}
```

Encapsulation in Java is a *process of wrapping code and data together into a single unit*, for example, a capsule which is mixed of several medicines.

We can create a fully encapsulated class in Java by making all the data members of the class private. Now we can use setter and getter methods to set and get the data in it.

READ ONLY

```
public class Student{
//private data member
private String college="AKG";
//getter method for college
public String getCollege(){
return college;
}
}
```

WRITE ONLY

```
public class Student{
//private data member
private String college;
//getter method for college
```

```
public void setCollege(String college){
this.college=college;
}
}
```

BOTH

```
class Account {
//private data members
private long acc_no;
private String name,email;
private float amount;
//public getter and setter methods
public long getAcc_no() {
    return acc_no;
}
public void setAcc_no(long acc_no) {
    this.acc_no = acc_no;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getEmail() {
    return email;
}
```

```
public void setEmail(String email) {
    this.email = email;
}
public float getAmount() {
    return amount;
}
public void setAmount(float amount) {
    this.amount = amount;
}}
```

YOUTUBE LINK:
https://www.youtube.com/watch?v=DaAJIMakWKU&list=PLd3UqWTnYXOkwIuXExifmJWKkvGa1ywWp