1) Classes ⇒

A Class is like an "object constructor" or a blueprint for creating objects. Eg Public class main { Variables. ← X=5; }

2) Objects ⇒

Everything in java is associated with classes & objects, along with its attributes & methods. For Eg (In a real life Car is an object, the Car has attributes like weight, Color, methods such as drive & brake.

(Eg)
```
Public class Main {
    int x=5;
    Public Static Void main (String args[]){
        Main myobj = new Main();
        System.out.println( myobj.x);
    }
}
```

3) Packages ⇒

* A Package can be defined as a group of related types (Class, Interface, enumeration & annotation) providing access protection & name space management.

Enumeration ⇒ Enum type is a special data type that enables for variable to be a set of predefined constants. (Eg) Compass direction (NORTH, SOUTH, EAST, WEST)

4. Primitive Datatypes =>

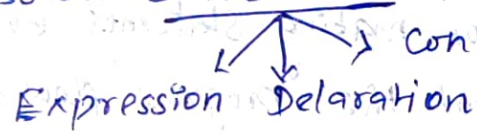
* primitive types are predefined (already defined) in java. [int, byte, Float, Short, long, double & char]

Non-primitive => Non primitive data types created by the programmer and is not defined by Java (except String). This types can be used to call methods to perform certain operations. Non-primitive types can be "null": (String, Arrays)

5) Wrapper classes => provides a way to use primitive data types, as object. It allows null values. [Character, Double, Integer]

Use => The primitive data types in java, are not objects, by default. They need to be converted into objects using wrapper classes.

6) Expression statements =>



* An Expression statements in java is a combination of Variables, operators & methods call constructed according to the language's Syntax's, which evaluate a single values.

(Eg) a = 6;
c = a+b;
++j;

Use => It is used to generate new values. We can also assign a values to a variables. In java expression is the combination of values, variables, operators & method calls.

Declaration Statements ⇒

We declare variables & constants by specifying their data types & names. A variable hold the values

(Eg.) int quantity ;
boolean flag ;

Eg) int quantity = 10;

Eg) int quantity, batch_no, lot_no

Control Statements ⇒

1) Conditional Statement: (or) Selection [IF, IF else, if else if, switch]
2) Loop [For loop, while loop, do while loop, for each loop]
3) Flow Control [return, continue, break]
(Or)
Branching statements.

Statements ⇒ Java provides statements that can be used to control the flow of Java code.

↳ Decision making Statements ⇒ decides which statement to execute and when. Decision making statements evaluate the boolean expression & control the program flow depending upon the result of the condition provided.

2) if else ⇒ The else block is executed if the condition of the if block is evaluated as false.

4) Nested if else ⇒ Multiple if statements, finding else.

5) Switching Statements ⇒ The switch statements contains multiple block of code called cases & a single case is executed based on the variable which is being switched. * [Switch Case in Java is a multiway branch statements. | Statements from multiple conditions. The use of If-else we use Switch case.

2. Looping loop statements are used to execute the set of instruction in a repeated order. The execution of the set of instructions depends upon a particular conditions.

For loop ⇒ It enables at initialize the loop variables, check the conditions, and increment/decrement in a single line.

Use ⇒ We use the for loop only we exactly know the number of times we want to execute the block of code.

⇒ [For each loop ⇒ A for each loop is a loop that can be used on a collection of items. In for-each loop, we don't need to update the loop variables.

```
for( data_type var : array_name / collection_name){
    //Statements.
}
```
]

While_loop ⇒ The while loop also used to iterate over the number of statements multiple times. However, if we don't know the number of iteration in advance, it recommend to use while loop.

Do_while loop ⇒ The do while loop check the conditions at the end of the loop after executing the loop statements. * when the number of iteration is not know & we have to execute the loop at least once.

Syntax ⇒ do {
//statements
} while (conditions)

1) Java Break Statements =>

* The break statement is used to break the current flow of the program & transfer the control to the next statement outside a loop (or) switch statement.

* The break statement cannot be used independently.

2) Continue Statement =>

* The continue statement doesn't break the loop, whereas it skips the specific part of the loop & jump to the next iteration of the loop immediately.

3) Return Statement =>

* The return statement stops the execution of a method & transfers control back to the calling code. If the param function is void, the return statement does not carry any value. For non void method, the return statement will always have a return values.

Method Invocation =>

* m_1 calls m_2, m_2 calls m_3, m_3 calls m_4.

```
m1(){
    m2();
}
m2(){
    m3();
}
m3(){
    m4();
```

* One method is calling another method.

Recursion ⇒

* Recursion is the technique of making a (or) functions "call itself". (methods)

(Example) Factorial program.

Arrays ⇒ Arrays are used to store multiple values in a single variables, instead of declaring seperate variable for each values.

* Same data type & Repeated elements. We can use Arrays.

* To declare an Array, define the variable type with square brackets.

(Eg) ⇒ String[] cars;

⇒ Methods ⇒

* A Method is a block of code which only runs when it is called. Methods are used to perform certain actions, and they are also known as "function".

Use ⇒ To Reuse code ⇒ Define the code once & use it many times.

⇒ Access Modifiers ⇒

* Access modifiers are keywords that can be used to control the visibility to the fields, methods, Constructor in a Class.

* Public, protected, private, default.

Static keyword =>

* The Static keyword in java is mainly used for memory management.

* The Static keyword in java is used to Share the same variable (or) method of a given class.

* The static keyword is used for a constant variable (or) a method, that is same for every instance of a class.

Static block =>

* This code inside the block is executed only once, the first time the classes is loaded into a memory.

Static class =>

Two types

=> Static Nested class

=> Non Static Nested class. (Inner class)

* Static class in java is a Nested class & it doesn't need reference of the outer class.

* Static class only can access the static members of its outer class.

Final keyword ⇒

* The Final keyword in java used to restrict the user.
* Stop Value changing, Stop method overriding, Stop Inheritance.

1) Final variable ⇒ If you make any variable as final, you cannot change the value of final variable, (It will be constant)

(eg) Final int speedlimit = 90 ;

2) Final method ⇒ you cannot overriding.

3) Final class ⇒ you cannot extend it.

Local Variable	Global Variable
* A variable that is delared inside a function of a computer programs.	* outside a function of a computer programs.
* Accessible only within the function it is declared	* Accessible by all the functions of the program.
* More reliable & secure.	* Accessible by multiple functions ; therefore, its values can be changed.

String class ⇒

*

Oops concepts ⇒

* Inheritance
* Association ⇒ Composition, Aggregation

Interface body ⇒
*abstract method within an interface followed by a ; semi but no braces.

*Default method defined with the default modifier, static method with static keyword.

Interfaces ⇒

* An Interface in java is a blueprint of a class. It has only Method signature, not a body.

*Interface keyword (implements). Interface are used in Java to achieve abstraction.

Abstraction ⇒

* Data abstraction is the process of hiding certain details from the user, & showing only essential information to the user.

* Abstraction can be achieved with either abstract classes (or) Interfaces.

(Eg)

```
abstract class Animal{
    Public abstract void animalSound();
    Public Void sleep(){
        Sysc("Mo");
    }
}.
```

Encapsulation ⇒

* The meaning of Encapsulation, is to make sure that "sensitive" data is hidden from the users.

* To achieve this, ⇒ declare class variable/ attributes as "private"

* provide public get, set methods to access, update the value of private variables.

Polymorphism ⇒

* Poly Means Many & morphs means forms. The word polymorphism having many forms.

* We can define polymorphism as the ability of message to displayed in more than one form.

(Ex) Two methods in same name, to execute two different functionalities. (Different parameters)

Data hiding ⇒ refers to hiding internal object details such as data member from outside the users. Hide & secure the data using access modifiers.

Constructor ⇒

* A Constructor in java is a special method that is used to initialize objects.

* The Constructor is called when an object of a class is created. It can be used to set initial values for object attributes.

Multiple arg Constructors ⇒

* is nothing but, multiple arg with same name, different parameters.

Method overloading ⇒

* If a class have multiple methods in having same name but different in parameters, is known as methodoveloading.

Method overriding ⇒

* If subclass (Childclass) have same method as declared in the parent class, is known as method overriding in java.

One dimensional arrays ⇒

* Is a group of elements having the same data types in different elements.

Two dimensional arrays ⇒

* 2D arrays are stored arrays of arrays.

* 2D array are declared by defining a datatype followed by two set of square brackets.

Exception Handling => * The Exception Handling in Java is one of the powerfull mechanism "to handle the runtime errors", So that the normal flow of the application can be maintained.

File handling =>

* File Handling in Java defined as reading and writing data to a file. The particular File class From the package java.io allows us to handle & work with different Format of Files.

Multithreading =>

* Multithreading is a programming concept in which the application can create a small unit of tasks to execute in parallel.

* If a programming language supports creating multiple threads & passes them to the operating system to run in parallel, it is called multithreading.

Java Concurrency => IS the capability of the Java platform to run multiple operation simultaneously. The operation could be multiple Java program (or) parts of a single Java program.

Java Reflection => Reflection allows us to inspect & manipulate classes, Interfaces, Constructor, methods & Fields at a time.

⇒ JVM

* Java virtual Machine acts as run time engine to run java applications. JVM is one that actually calls the main method present in a java code. JVM is a part of JRE (Java Run time Environment).

* JAVA Application called (WORA) write onces Run anywhere.

⇒ Garbage collection ⇒

* In Java is the "automated process of deleting code that's no longer needed (or) used".

* This automatically frees up memory space and ideally makes coding java apps easier for developer.

⇒ Memory-Mapping Files ⇒

* Memory mapping files are causual special files in Java that helps to access content directly from memory.

* Java programming supports memory-mapped file with java.nio.package. Memory-mapped I/o uses the filesystem to establish a virtual memory mapping from the user directly to the filessystem. pages.

1) JVM =>

* Java Virtual Machine (JVM) act as a run time engine to run Java application. JVM is the one that actually calls main method present in java code. JVM is a part of Java Runtime Environment).

* Java application are called WORA (Write once Run Anywhere). This means a programmer can develop a Java code on one system & can expect it to run on any other Java-enabled system without any adjustment. This is all possible because of JVM.

Source code (Java code)
↓
Compiler
↓
Bytecode
↓ JVM (Windows, mac) → Machine code

2) Byte code =>

* Byte code can be defined as an intermediate code generated by the compiler after the compilation of Source code (Java program). This intermediate code makes Java a platform independent language.

3) Java Compiler =>

* A Java Compiler is a program that takes the text file work of a developer and compiles it into a platform independent Java files. Java compilers include the Java programming language compiler (javac) and the Eclipse Compiler of Java (ECJ)

4. Command to Compile java file =>

1. Open a Command prompt
2. Type javac FirstProgram.java & press enter to Compile java code. If there is no error in your code, the command prompt will take you to the next line.

=> In Eclipse, by default it will compile source code, If Save the file

5. Command to run (or) execute java file =>

* After the compilation, type java and program name to run your program.

=> java. Fistprogram.

=> In Eclipse, Right Click and Run as -> Java project.

6. Class Variable nothing but "Static Variable"

21/11/2023

1. ⇒ Parameters ⇒ Information can be passed to method as parameter. Parameters act as variable inside the method.

* Parameters are specified after the method name, inside the parantheses. You can add as many parameters as you want, just seperate them with comma.

(Eg) Static void MyMethod(String fname){
}

2) Operator precedence ⇒

* The operator precedence represents how two expression are bind together. In one expression, it determine the grouping of operators with operands and decides how an expression will evaluate.

* While solving an expression two things must be kept in mind, the fist is "precedence" & the second is "associativity".

a+b*c ⇒ 4+6*2

Precedence is the priority for grouping different types of operators with their operand.

operators

Assignment ⇒ Assignment operators are used to assign values to Variables. (=), (+=) addition assign

↳ operators are used to perform operation on Variables & Values. ⇒ operator (+), −, *)

Arithmetic operation ⇒

* Arithmetic operators are used to perform Common mathematic operations. (Eg) (+, −, *, /, %, ++, −−)

Comparison operators ⇒

* Comparison operators are used to compare two values. Eg (==, <, >, <=, >=, !=)

Unary operators ⇒

* Unary operator is an operator that can be used only with an operand. It is used to represent the positive & negative values, increment, decrement values by 1, Complement a boolean values.

⇒ Unary plus, Unary minus.

Relational operation Two category 1. Comparision, 2. Equality.

Equality ⇒ Relational operators are used to check between two operands.

* == Operator is a type of "Relations operator in Java" used to check for relations of equality. It returns a boolean Result After the comparision & is extensively used in looping Statements & Conditional if else Statements.

Conditional operators in java =>

* Conditional operators check the conditions and decides the desired result on the basic of both conditions.

=> AND &&, OR -||, Ternary ?:

Blocks

* A Blocks is a set of statements enclosed in set braces.

```
if( ) {

}
```

Class declaration =>

* A class is blueprint for creating an object. We may delare a class using the class keyword. We can declare a class with access modifiers, class name, & curly braces { }.

Member Variables => also known as Instance variable.

*

Variable names =>

* The name of the variable should begin with alphabet (or) underscore (or) a dollar sign ($)

* no space (or) special characters are not allowed in the variable names.

Naming a Method =>

* While writing a method name we should follow the camel case ie, first letter of the first word should be small & the first letter of the remaing words should be capital.

Public void demoMethod()

{

}

Providing Constructor for your class =>

* A class contains constructors that are invoked to create objects from the class blueprint. Constructor declartion look like method declaration - except that they use the name of the class & have no return type.

Passing information to a method on a constructor =>

*

Oops Concepts ⇒

Inheritance → In Java, it is possible to inherit attributes, methods from one class to another. Inheritance is used to implement is a relationship.

* Two Categories ⇒ Subclass (child)
  Superclass (parent)

* To inherit from a class, use the "extends" keyword.

* Attributes ⇒ X, Y
  Methods ⇒ (eg) Public void displayPlayerInfo()

Public int multiply (int num1, int num2) {

Public → access Modifier
int → datatype
multiply → methodname
int → parameter type
num1, num2 → Parameter name

The Final keyword ⇒

* If you don't want other classes to inherit from a class, use the final keyword.

Composition

* Composition is a way to design (or) implement has-a relationship. (Eg) Person has an address.

* It is code reusability. "Tightly coupled" delete Parent Child also delete.

* A composition in Java, between two objects associated with each other exist, when there is a strong relationship between one class and another. A "Human" class is a composition of hearts & lungs.

Instance Methods → The Method defines in a class, which is only accessible through the object of the class.

Object creation ⇒ Puppy P = new Puppy("TOM");

Calling the method ⇒ P.setAge(2);

Instance Variable ⇒ (Eg) datatype dataname;

Types of Variable ⇒ Local Variable, Instance Variable, Static

↓

A Variable defined the within a block (or) methods (or) Constructor is called Local Variable.

Instance Variable ⇒ An Instance variable are declared in a class, these variables are created when the Object of the class is created, then destroyed when the Object is destroyed. (Eg) Public String name;
Private int age;

Static Variable ⇒ (↑ global) When ever a variable declared as Static, outside a method, constructor or a block. this means there is only one copy of it entire class, rather then each Instance having its own copy.

(Eg) Static String CollegeName = "ITS";

Static Method ⇒ A Static Method mean it can ~~created be called~~ without creating instance of the class.

⇒ Object ⇒ Attributes & methods [Eg (Car)]
↓ ↓
weight drive, brake

Class ⇒ Blue Print for creating object.

Aggregation ⇒ When an object A contains reference to another object B (or) we can say Object A has relationship with object B. (not tightly coupled)

Association Two forms ⇒ Aggregation, Composition.

is Relation Between two seperate classes, which establish through the object. (Association between the two classes)

Association Manages one to one, one to many, many to many,

Encapsulation ⇒ Data hiding, because other class will not be able to access data through the private data members.

* It is the process of wrapping code, & data together into a single unit. Variables (or) data of a class hidden from any other class.

Interface ⇒ Only have signature, not a body.

keyword → Implement

* The interface in Java is a mechanism to achieve abstraction.

* Default, static methods, Private Methods

* Loose coupling, Multiple Inheritance

Access Modifiers ⇒

	default	Private	protected	Public
Same class	yes	yes	yes	yes
Same class-Subclass	yes	No	yes	yes
Same Package-non Sub class (different class)	Yes	No	Yes	Yes
Different package-Subclass	No	No	Yes	Yes
Different Package non Subclass	No	No	No	Yes.