**Java Fullstack Development Program**

Trainer: Shekher

(Working Professional)

Duration: 6 Months

Time: 9AM to 11AM

Fee: 30k

( Live classes + Class Notes + Material + Backup videos(1 year validity) + Mock interviews + Placement Assistance)

FullStack Development = Front-end development + Back-end development + Database + Cloud(AWS)

Front-end development: HTML(Hyper Text Markup Language)

CSS ( Cascading Style Sheet)

Javascript

Bootstrap

React JS

Back-end development: Core JAVA

Adv JAVA (JDBC, Servlet, JSP)

Spring Framework

Hibernate

Spring Boot

Microservices

Database: Oracle, MySQL, MongoDB

Front-end : Front-end development means, designing and developing the web

pages.

A web page is used to accept the input/data from the users and also

used to display the information to the users.

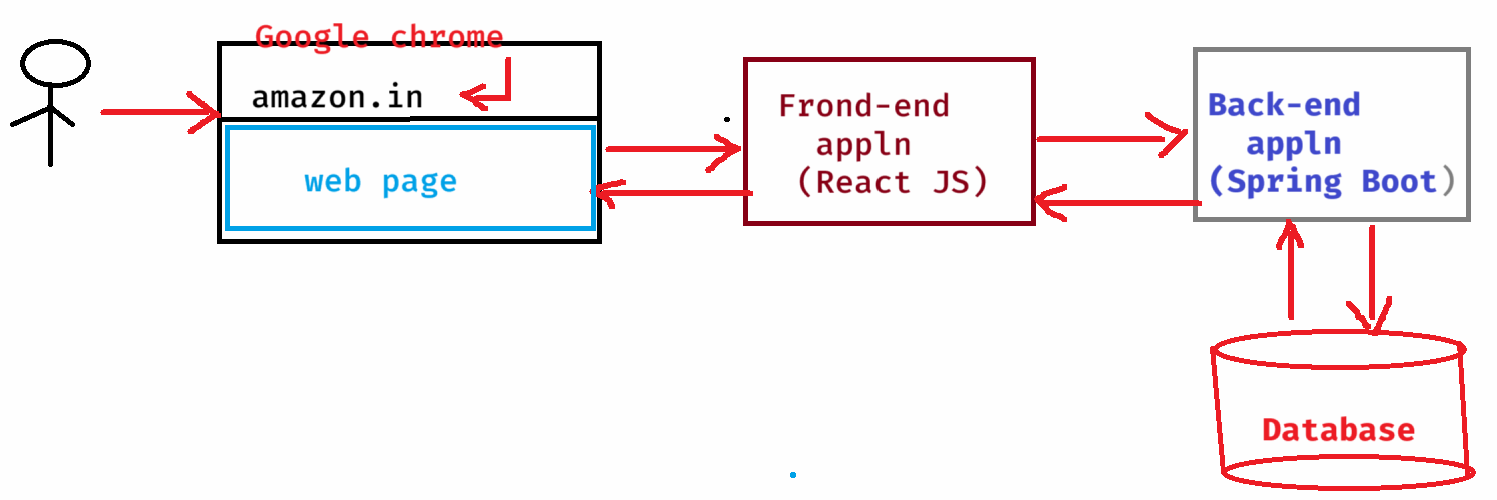
Back-end: Back-end development means, implementing business logics of an

organization. For example, calculationg order total, taxes, final price,

discounts, etc..

Database: Database is a storage area for the data of the project.

For example, users data, products data, payments data, etc..



CORE JAVA

------------------------

programming language?

* programming language is used to define programs, which can instruct a computer to do certain task.
* computer can understand only Binary language(0’s and 1’s), and the programming languages have translators, which can convert the program into binary language.
* Java is also a programming language and it has 3 modules.

1. Java SE(JSE) : Java Platform, Standard Edition
2. Java EE(JEE): Java Platform, Enterprise Edition
3. Java ME(JME): Java Platform, Micro Edition

* Java SE module, we call it as Core Java.
* Java SE module is the base for Java EE module.
* Using Java SE module, you can develop stand-alone applications.
* stand-alone applications means, the applications which you are downloading it, then installing it and then using it.
* For example, Anti-virus software application, you are downloding it from a website, then installing it and then using it to scan for virus. So, it is a stand-alone application.
* Stand-alone applications are also called desktop applications.
* Java EE module can be used for developing web applications and enterprise applications.
* small scale applications are termed as web applications and large scale applications are termed as enterprise applications.
* For example, restaurent websites, hotel booking websites, IT trainings websites, etc.. are small scale applications. So, we can call them as webapplications.
* For example, banking websites, stock trading websites, etc.. are large scale applications. So, we can call them as enterprise applications.
* To develop a Back end application, You need to learn both Java SE module and Java EE module.
* The first name of Java was “OAK”.
* OAK language development was started in 1991, and it was renamed to Java in 1995.
* The first release of Java was on Jan 1996.
* Java was introducted with a solgan “Write Once, Run Anywhere(WORA)”.
* WORA means, You can develop the application on one Operating System and you can run it on any other operationg Systems.
* Java was developed by a team of 5 people, headed by “James Gosling”.

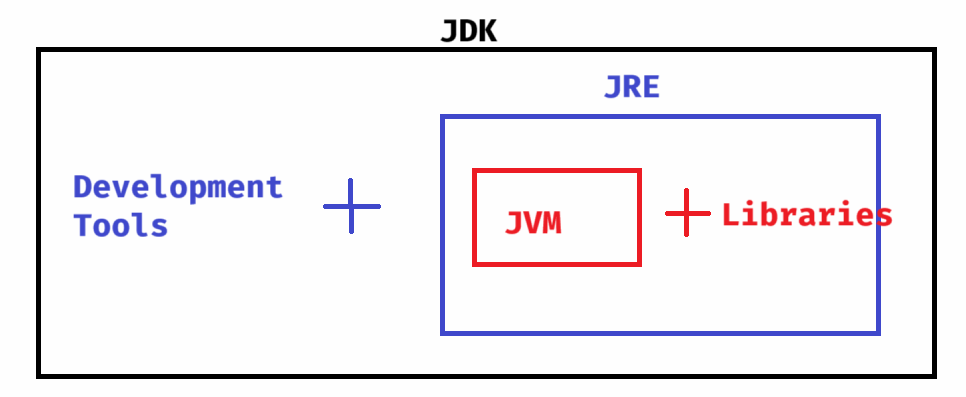
JDK, JRE and JVM:

JDK -- Java Development Kit

JRE -- Java Runtime Environment

JVM – Java Virtual Machine

* **JDK is a complete package which can be used for writing the programs, compiling the programs and executing the programs.**
* **The current version of JDK is Java SE 24(March 2025).**
* **JDK = Development Tools + JRE**
* **JRE = JVM + Libraries**
* **Development tools are like compiler, debugger, documentation tool, profiler etc..**
* **JRE is only responsible for executing the program, not for compiling the program.**
* **JVM is a part of JRE, which is responsible for converting Java’s byte code into Machine code and also for memory allocations.**
* **Libraries are nothing but Java API.**
* **API : Application Programming Interface.**
* **Every language provides some API for developing the programs by the programmers.**

****

**Installing Java(JDK 21):**

**step1: visit** [**https://www.oracle.com/java/technologies/javase/jdk21-archive-downloads.html**](https://www.oracle.com/java/technologies/javase/jdk21-archive-downloads.html)

**step2: click on the link of Windows x64 msi installer.**

**step3: jdk-21.0.5\_Windows-x64-bin.msi file will be downloaded.**

**step4: double click on the downaloded msi file, then click on Next 🡪 Next 🡪 Close.**

**step5: The Java software is installed at,**

**C:\Program Files\Java\jdk-21**

**Java Path settings:**

**------------------**

**step1: windows search 🡪 type environment**

**step2: click on edit System environment variables**

**step3: click on Environment variables…**

**step4: Select Path variable(under system variables), then**

**click on edit button.**

**step5: click on New button, and enter**

**C:\Program Files\Java\jdk-21\bin**

**step6: Move up this value to the top place.**

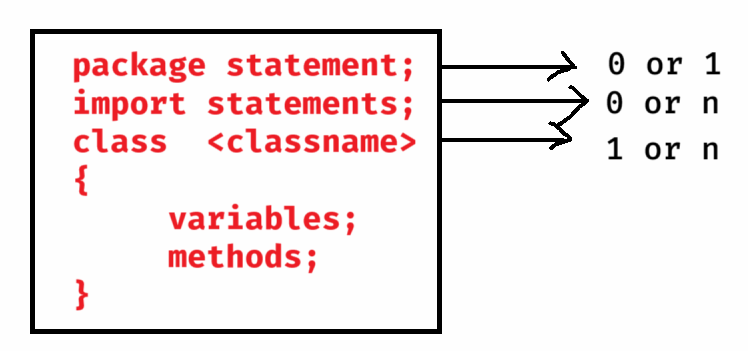
**step7: ok 🡪 ok 🡪 ok**

**. To verify the Java Path is set properly or not,**

1. **open cmd**
2. **run the below command.**

**java -version**

**Basic structure of a Java program:**

****

**main method:**

**------------**

**. Every Java program execution starts from the main method.**

**. JVM calls this main method to execute the program.**

**. The syntax of main method is,**

**public static void main(String[] args)**

**output statement:**

**---------------**

**System.out.println(“HELLO”);**

**First Java program:**

1. **open Notepad**
2. **write the Java code.**

**class Hello**

**{**

**public static void main(String[] args)**

**{**

**System.out.println("Welcome to JAVA");**

**}**

**}**

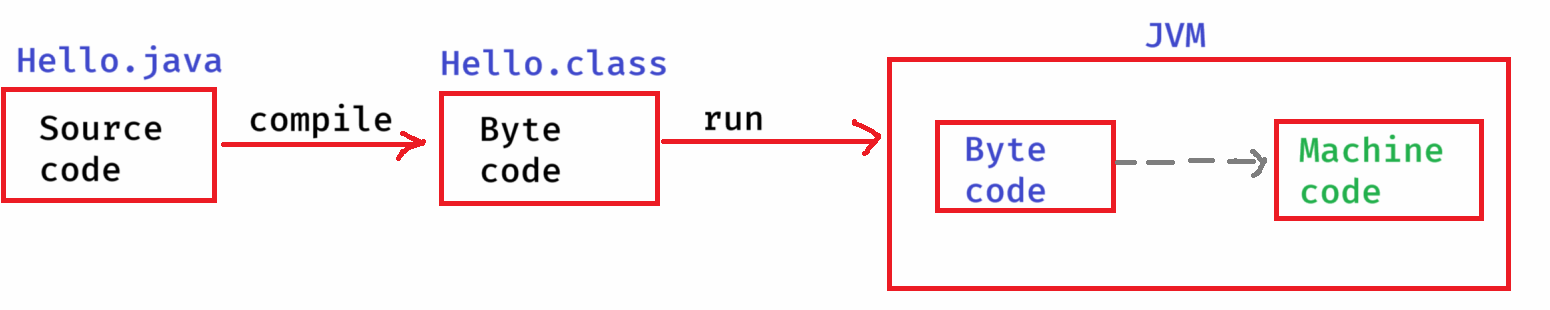
1. **Save the program with file name Hello.java**
2. **open cmd**
3. **move to the location, where the program is saved.**
4. **compile the program.**

**D:\>javac Hello.java**

1. **execute the program.**

**D:\>java Hello**

**output: Welcome to JAVA**

****

* **Every programming language contains 4 types tokens.**

1. **keywords**
2. **identifiers**
3. **data types**
4. **operators**

* **Every programming language contains 5 types of statements.**

1. **input statement**
2. **output statement**
3. **memory statement**
4. **arithmetic and logic statement**
5. **control statement.**

**keyword:**

**-------**

* **It is a word which has a pre-defined meaning given by the programming language.**
* **These are also called as reserved words.**
* **In Java, we have more than 55 keywords.**
* **ex:**
* **class, public, private, interface, final, extends, implements, volatile, transient, etc..**

**identifiers:**

* **The names assigned to the variables or methods or classes are called as identifiers.**
* **identifier rules,**
* **1. white space is not allowed in identifier**
* **2. Identifier can have letters(upper&lower), digits(0-9), underscore( \_ ) and dollar($). Other special characters are not allowed.**
* **3. Don’t start identifer with a digit.**
* **4. Don’t use a keyword as identifier.**
* **5. identifiers are case-sensitive.**

**ex:**

**int @cost = 200; // invalid identifier**

**int cost@today = 300; //invalid identifier**

**int cost\_today = 540; //valid**

**int cost today = 600; //invalid identifier**

**int final$Price = 1250; //valid**

**int private = 20; // invalid identifier**

**int \_$cost = 100; // valid**

**int 1a = 10; // invalid identifier**

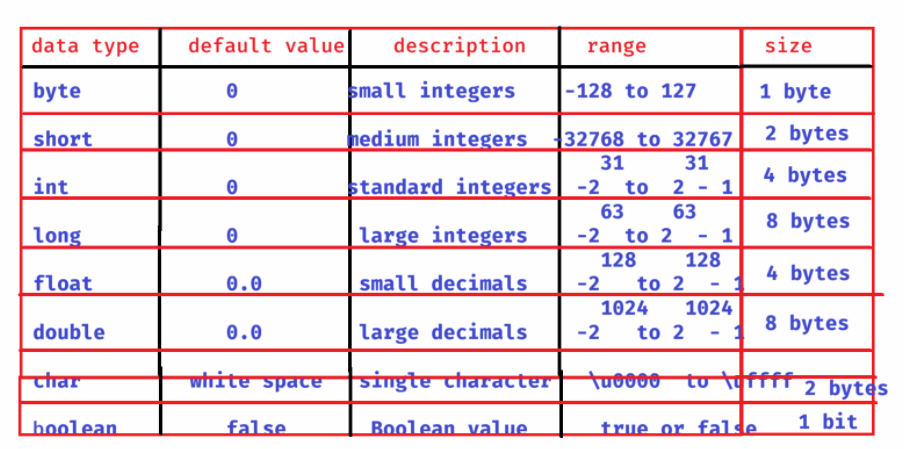
**data types:**

* **A data type represents the type of data that a variable can hold/store and also tells the JVM about how much memory should be created for the variable.**
* **Data types are 2 types.**

1. **primitive data types/ value types**
2. **non-primitive data types / reference types.**

* **value types can store the values like integers, decimal point values, character values, boolean values.**
* **reference types can store the objects.**

**primitive data types table:**

****

**declaring a variable:**

**-------------------**

**datatype variablename; //variable declaration**

**or**

**datatype variablename = value; //declaration + initialization**

**ex1:**

**byte b1 = 110;**

**byte b2 = 10;**

**byte b3 = b1 + b2;**

**Sop(b3); //ouput: 120**

**ex2:**

**byte b1 = 110;**

**byte b2 = 100;**

**byte b3 = b1 + b2; //error**

**Sop(b3);**

**ex3:**

**long x = 21989L;**

**. A long value should contain a suffix L or l.**

**ex4:**

**float price = 3.50f;**

**. A float value should contain a suffix F or f.**

**ex5:**

**boolean married = false; //correct**

**boolean married = 0; //error**

**. In Java, boolean values accepted are true/false. But**

**not 0 or 1.**

**. Internally JVM stores false as 0 and true as 1.**

**Installing Eclipse :**

**IDE : Integrated Development Environment**

**IDE’s are Eclipse, STS(Spring Tool Suite), IntelliJ,**

**VS Code, NetBeans, etc..**

1. **eclipse.org/downloads**
2. **click on download packages link**
3. **Download Eclise IDE for Enterprise Java and Web Developers.**
4. **eclipse-jee-2025-03-R-win32-x86\_64.zip file will be downloaded.**
5. **extract the downloaded file.**
6. **A folder, eclipse-jee-2025-03-R-win32-x86\_64 is created.**
7. **Open this folder, then open eclipse folder. You can find eclipse Application.**
8. **Note: create desktop shortcut for eclipse application, to start the IDE easily.**

**Writing first Java program on Eclipse:**

* **Before you start working with Eclipse, create a folder.**
* **This folder can be created at any location in your system.**
* **This folder will act as a workspace.**
* **In Eclipse terminology, a workspace is nothing but it is a folder where the projects developed in eclipse will be stored.**
* **For example, create a new folder 90-jfsd-workspace at D:\workspaces.**

**step-1: start eclipse using icon on desktop.**

**step-2: click on browse button, choose the workspace folder then launch.**

**step-3: File Menu --> New -->Project -->Java Project -->Next -->project name: HelloWorldApp -->uncheck module-info.java file -->next -->finish**

**step-4: Expand HelloWorldApp 🡪 right click on src folder 🡪 New 🡪 class 🡪 Name: Demo 🡪 select public static void main check box 🡪 finish**

**step-5: insert the below statement into main method.**

**System.*out*.println("Hello World!!!");**

**step-6: Right click on Demo.java file 🡪 RunAs 🡪 Java Application**

**output: Hello World!!!**

**Take user input:**

* **To make a Java program as a dynamic program, it should take the input from the users.**
* **To take the input, Java has provided a pre-defined class called Scanner class.**
* **This Scanner class is provided in java.util package.**
* **Java has provided many pre-defined classes for the developers to develop the applications.**
* **These pre-defined classes are distributed into different packages, based on their functionalities.**
* **So, we can say that Java has provided multiple packages like java.util, java.io, java.lang, java.net, java.awt, etc..**
* **To take the user input using Scanner class, first you have to import the Scanner class.**
* **The import statement should be,**

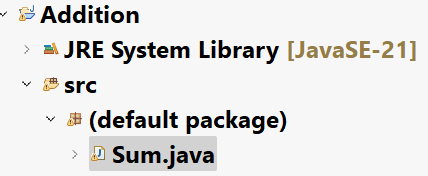
**import java.util.Scanner;**

* **You should connect the Scanner to keyboard input, using below statement.**

**Scanner scanner = new Scanner(System.in);**

* **Scanner class reads the keyboard input entered by user into string type first.**
* **Then it converts the input into the required data type, what programmer is asking for.**
* **To convert into the required data types, Scanner class has provided various methods.**
* **nextInt()**
* **nextLong()**
* **nextShort()**
* **nextByte()**
* **nextFloat()**
* **nextDouble()**
* **nextBoolean()**
* **next()**
* **nextLine(),etc..**

**example:**

****

**Sum.java**

**import java.util.Scanner;**

**public class Sum {**

**public static void main(String[] args) {**

**//connect Scanner to keyboard**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter first number");**

**int x = scanner.nextInt();**

**System.*out*.println("Enter second number");**

**int y = scanner.nextInt();**

**int z = x + y;**

**System.*out*.println("sum = " + z);**

**}**

**}**

**Java comments:**

**-----------**

**single line:**

**// single line comment**

**multiple lines :**

**/\***

**comment line1**

**comment line2**

**comment line3**

**\*/**

**documentation comments:**

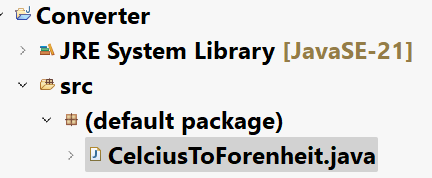
**/\*\***

**comment line1**

**comment line2**

**comment line3**

**\*/**

****

**CelciusToForenheit.java**

**/\***

**\* This program reads celsius value as input**

**\* from the user, then converts it into**

**\* forenheit value.**

**\*/**

**import java.util.Scanner;**

**public class CelciusToForenheit {**

**public static void main(String[] args) {**

**//connect Scanner to the keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter celsius value");**

**float celsius = scanner.nextFloat();**

**//convert celsius value to forenheit**

**float forenheit = ( celsius \* 9 / 5) + 32;**

**//display the output**

**System.*out*.println("celsius value = " + celsius);**

**System.*out*.println("forenheit value = " + forenheit);**

**scanner.close();**

**}**

**}**

**ForenheitToCelsius.java**

**/\***

**\* This program reads forenheit value as input**

**\* from the user, then converts it into**

**\* celsius value.**

**\*/**

**import java.util.Scanner;**

**public class ForenheitToCelsius {**

**public static void main(String[] args) {**

**// connect Scanner to the keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter forenheit value");**

**float forenheit = scanner.nextFloat();**

**// convert forenheit value to celsius**

**float celsius = (forenheit - 32) \* 5 /9;**

**// display the output**

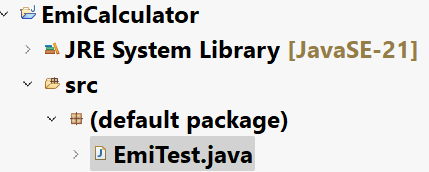
**System.*out*.println("celsius value = " + celsius);**

**System.*out*.println("forenheit value = " + forenheit);**

**scanner.close();**

**}**

**}**

****

**EmiTest.java**

**/\*\***

**\* This program calculates emi for the given input.**

**\* The input will be principle amount, rate of interest per annum**

**\* and tenure in years.**

**\* It has to convert rate of interest per annum to per month and**

**\* also tenure in years to in months.**

**\* The formula to calculate emi is,**

**\***

**\* n**

**\* emi = p \* r \* (1 + r)**

**\* -----------**

**\* n**

**\* (1 + r) - 1**

**\*/**

**import java.util.Scanner;**

**public class EmiTest {**

**public static void main(String[] args) {**

**//connect Scanner to the keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Please enter principle amount");**

**double p = scanner.nextDouble();**

**System.*out*.println("Please enter rate of interest per annum");**

**double pa = scanner.nextDouble();**

**System.*out*.println("Please enter tenure in years");**

**int years = scanner.nextInt();**

**//convert rate of interest per annum to per month**

**double r = pa / 12 / 100;**

**//convert tenure in years to in months**

**int n = years \* 12;**

**double x = Math.*pow*(1 + r, n);**

**double emi = p \* r \* x / (x - 1) ;**

**System.*out*.println("\u001B[32m" + "EMI to Pay = " + emi);**

**scanner.close();**

**}**

**}**

**Operators**

**Definition: operator is a symbol which can perform a well-defined task.**

**. operators are 3 types.**

* **unary operators**
* **binary operators**
* **ternary operators**
* **unary operators performs a task on a single operand.**
* **binary operators performs a task on two operands.**
* **ternary operator performs a task on three operands.**

**unary operators:**

**---------------**

1. **++ ----> increment operator**
2. **-- ----> decrement operator**

**a++ -----> post increment**

**++a -----> pre increment**

**a-- -----> post decrement**

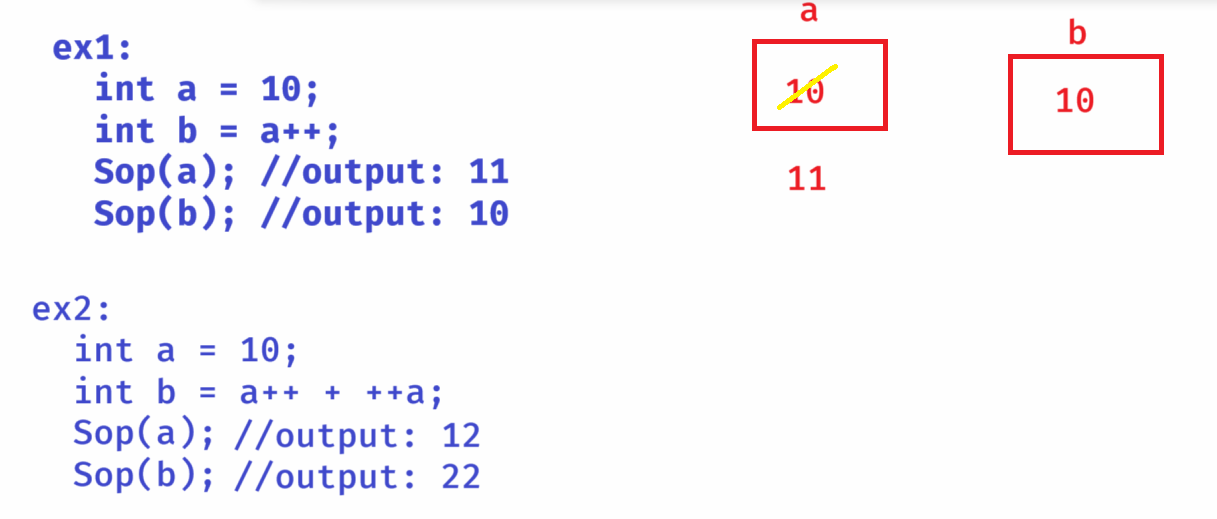
**--a -----> pre decrement**

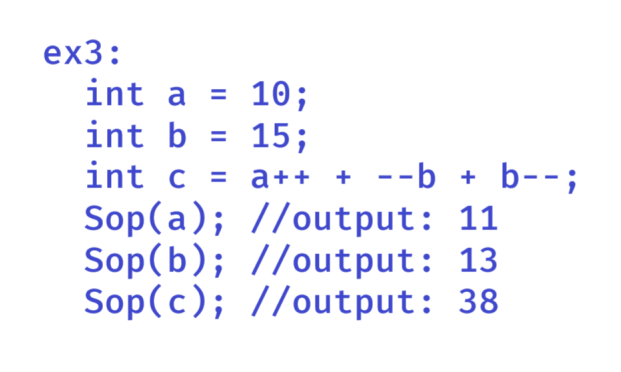
**a++ means, first use a, then increment it by 1.**

**++a means, first increment a by 1, then use it.**

**a-- means, first use a, then decrement it by 1.**

**--a means, first decrement a by 1, then use it.**

****

****

**Binary operators:**

1. **Arithmetic operators**
2. **Relational operators**
3. **Logical operators**
4. **Assignment operators**
5. **Bitwise operators**

**Arithmetic operators:**

**+ addition**

* **subtraction**

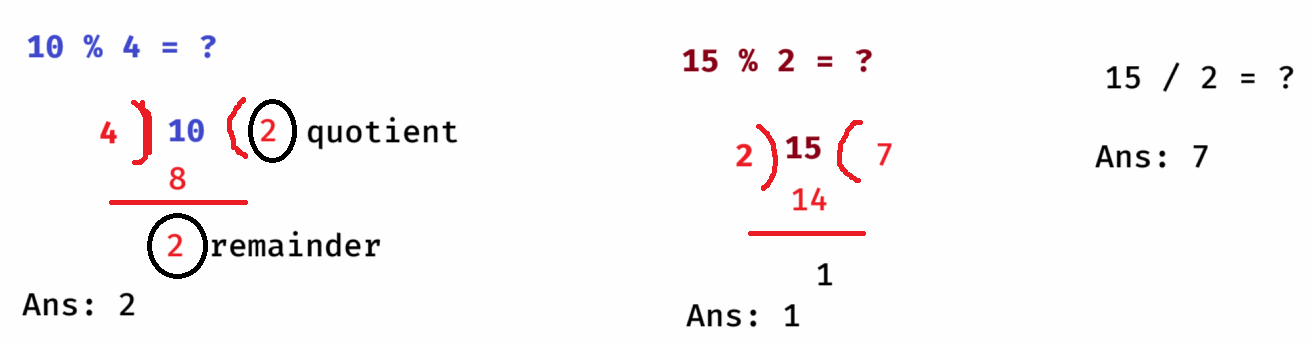
**\* multiplication**

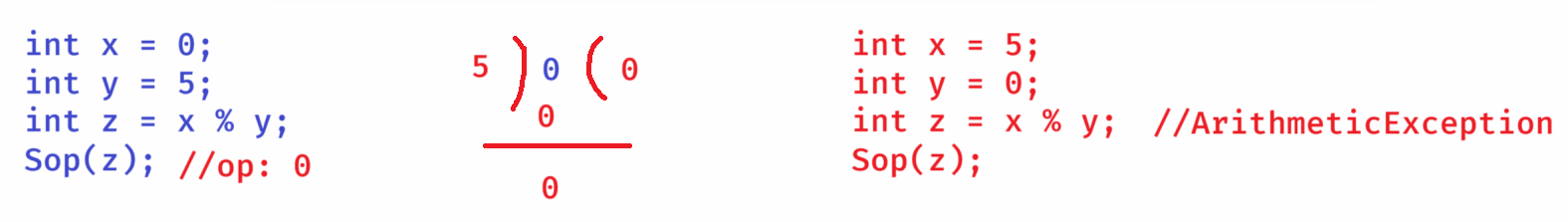
**% modulus**

**/ division**

* **modulus operator divides a numerator with denominator and returns remainder value.**
* **division operator divides a numerator with denominator and returns quotient value.**

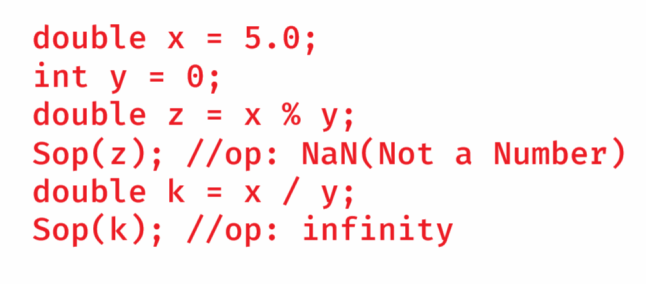
**ex:**

****

****

* **If you are dividing an integer with another integer, and if the denominator is zero(0) then Java throws ArithmeticException.**

**ex:**

****

* **If either numerator or the denominator is a float/double and if you are dividing with zero(0), modulus(%) operator returs NaN(Not a Number) and division(/) operator returns infinity.**

**ex:**

**double x = 10.3;**

**double y = 3.5;**

**double z = x / y;**

**System.*out*.println(z);**

**double k = x % y;**

**System.*out*.println(k);**

**ouput:**

**2.9428**

**3.3**

**note: if x and y are float/double variables then x % y is calculated as**

**x – ( y \* quotient(integer part))**

**ex:**

**char ch1 = 'A';**

**char ch2 = 'B';**

**int x = ch1 / ch2;**

**System.*out*.println(x);**

**int y = ch1 % ch2;**

**System.*out*.println(y);**

**output:**

**0**

**65**

**ex:**

**int x = 145;**

**int y = 145 % 10;**

**int z = 145 / 10;**

**Sop(y); //op: 5**

**Sop(z); //op: 14**

**Relational operators:**

**< less than**

**> greater than**

**<= less than or equals**

**>= greater than or equals**

**== equals**

**!= not equals**

**ex:**

**int x = 5;**

**if ( x < 10 )**

**{**

**int y = 90;**

**}**

**Sop(y);**

**output: error**

**ex:**

**String username = “admin”;**

**if ( username == “admin” )**

**Sop(“Welcome Admin”);**

**else**

**Sop(“Invalid user”);**

**output: Invalid user**

**ex:**

**String username = “admin”;**

**if ( username.equals(“admin”))**

**Sop(“Welcome Admin”);**

**else**

**Sop(“Invalid user”);**

**output: Welcome Admin**

**Logical operators:**

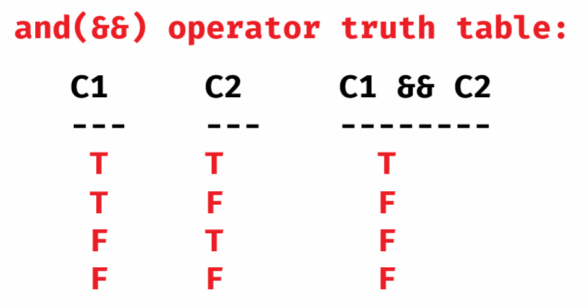
**----------------**

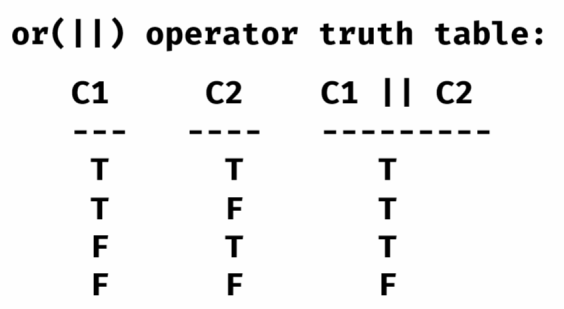
**&& and**

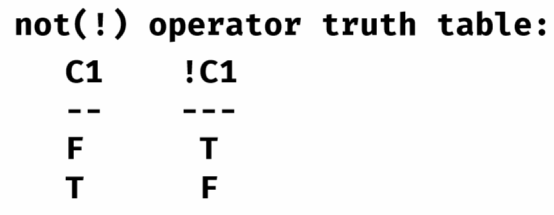
**|| or**

**! not**

* **Logical operators are used to combine the two conditions together as a single condition.**

****

****

****

**ex1:**

**int x = 10;**

**if (x++ > 10 && ++x < 15)**

**{**

**x = x + 40;**

**}**

**Sop(x);**

**output: 11**

**ex2:**

**int a = 10;**

**if( ++a > 10 || a++ < 13 )**

**{**

**a = a + 10;**

**}**

**Sop(a);**

**output: 21**

**Note: If the first condition is true then only the and(&&) operator will check the second condition.**

**If the first condition is false then only the or(||) operator will check the second condition.**

**ex3:**

**int a = 2, b = 10, c = 5;**

**if ( (a > 2 && b > c) || ++b < 12)**

**{**

**a = a + b + c;**

**}**

**Sop(a);**

**output: 18**

**Assignment operators:**

**= assign**

**+= add and assign**

**-= subtract and assign**

**\*= multiply and assign**

**/= divide and assign**

**%= modulus and assign**

**ex:**

**int a = 13;**

**a += 12; (a = a + 12)**

**Sop(a); //output: 25**

**ex:**

**int x = 37;**

**int y = 4;**

**x %= y;**

**x /= y;**

**Sop(x); //op: 0**

**Sop(y); //op: 4**

**Bitwise operators:**

**------------------**

**& Bitwise AND**

**| Bitwise OR**

**^ Bitwise XOR**

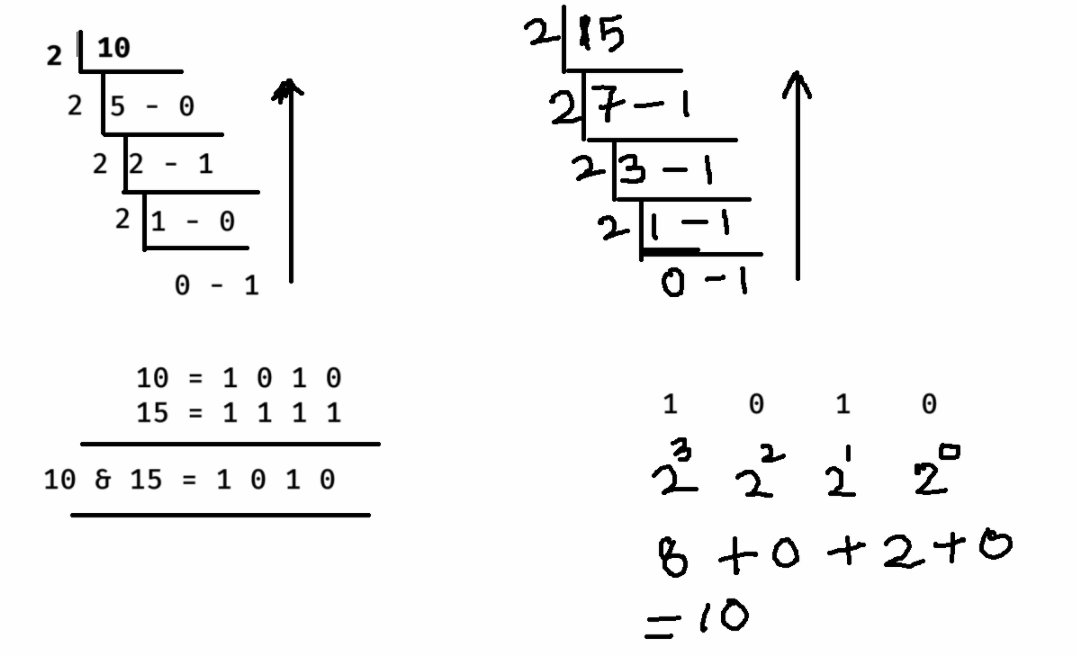
**<< left shift**

**>> right shift**

**Bitwise AND(&):**

* **If the corresponding bits of the operands is 1 then returns 1. Otherwise, returns 0.**

**10 & 15 = ?**

****

**12 & 8 = ?**

**12 = 1 1 0 0**

**8 = 1 0 0 0**

**-----------------**

**12 & 8= 1 0 0 0**

**---------------**

**output: 8**

**Bitwise OR(|):**

**-----------**

* **If the corresponding bits of the operands are 0, then returns 0. Otherwise returns 1.**

**10 | 20 = ?**

**10 = 0 1 0 1 0**

**20 = 1 0 1 0 0**

**--------------------**

**10 | 20 = 1 1 1 1 0**

**--------------------**

**output: 30**

**Bitwise XOR(^):**

**-----------**

* **If the corresponding bits of the operands are opposite then returns 1. Otherwise returns 0.**

**12 ^ 17 = ?**

**12 = 0 1 1 0 0**

**17 = 1 0 0 0 1**

**-------------------------**

**12 ^ 17 = 1 1 1 0 1**

**-------------------------**

**outut: 29**

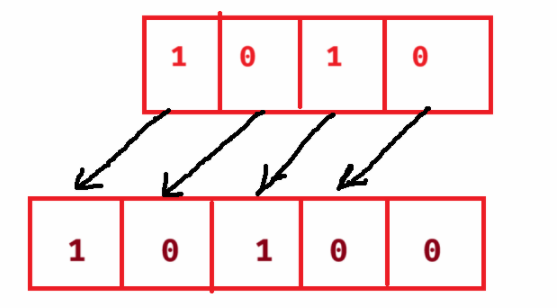
**Left shift(<<) :**

* **It shifts the bits of a binary number to the left side by the specified positions. Fills the empty bits with 0 on the right side.**

**ex:**

**10 << 1 = ?**

**10 = 1 0 1 0**



output : 20

right shift (>>):

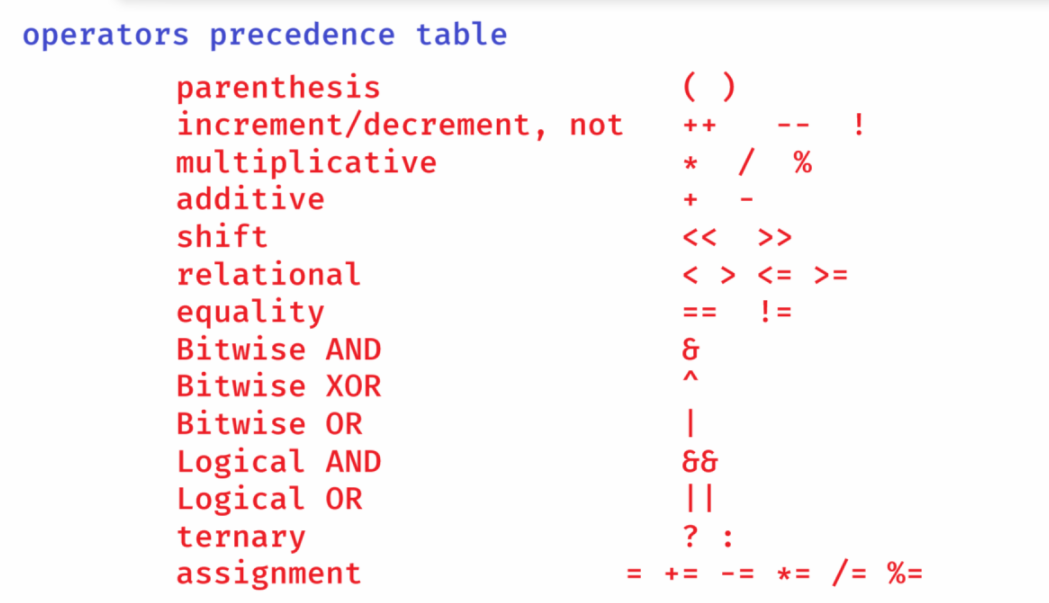
* It shifts the bits of the binary number to the right side by the specified positions. Fills the empty bits created at left side with 0.

10 >> 1 = ?

10 = 1 0 1 0

10 >> 1 = 0 1 0 1

output : 5



ex1:

int x = 9 \* 4 / 6 – (3 + 2 \* 2);

Sop(x);

output:

x = 9 \* 4 / 6 – 7;

x = 36 / 6 – 7;

x = 6 – 7;

x = -1;

ex2:

int x = **6 – 7 \* 8 + 9 / 5 – 6 % 3 – 7 / 2 - 3;**

**Sop(x);**

output: -55

ex3:

int x = **3 \* 5 – 7 + 7 \* 7 – 7 / 7 + 7 \* 7 % 7;**

Sop(x);

output: 56

ex4:

**int x = 10 ^ 15 & 10 | 5;**

Sop(x);

output: 5

ternary operator( ? : ):

----------------------

result = condition ? value1 : value2;

* first condition is executed, and if it is true then value1 will be stored into result. If it is false then value2 will be stored into result.

ex1:

int a = 23;

int b = 19;

int result = a > b ? a : b;

Sop(result);

ouptut: 23

ex2:

int a = 15;

int b = 10;

int c = 18;

int result = (a > b && a >c) ? a : (b > c) ? b : c;

Sop(result);

output: 18

ex3:

int year = 2032;

String str = ((year%4==0 && year%100 != 0) || year%400==0)? “Leap” : “not leap”;

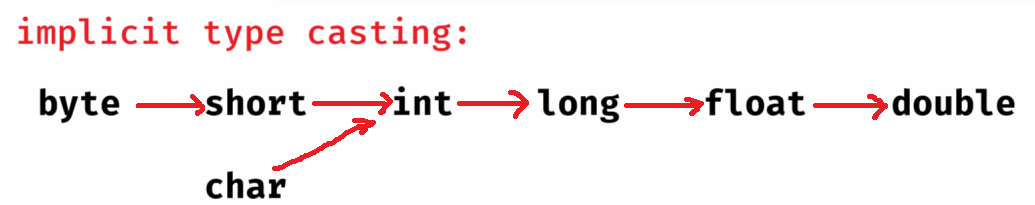
Sop(str);

output: Leap

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

Type casting:

* converting a value from one type to another type is called type casting.
* Type castings are 2 types.
* 1. implicit type casting / widening
* 2. explicit type casting / narrowing
* implicit type casting means, storing a variable from a low memory type to higher memory type.
* For example, if you are storing a variable of type int into another variable of type long, then it is implicit type casting.
* implicit type casting will be taken care by Java.



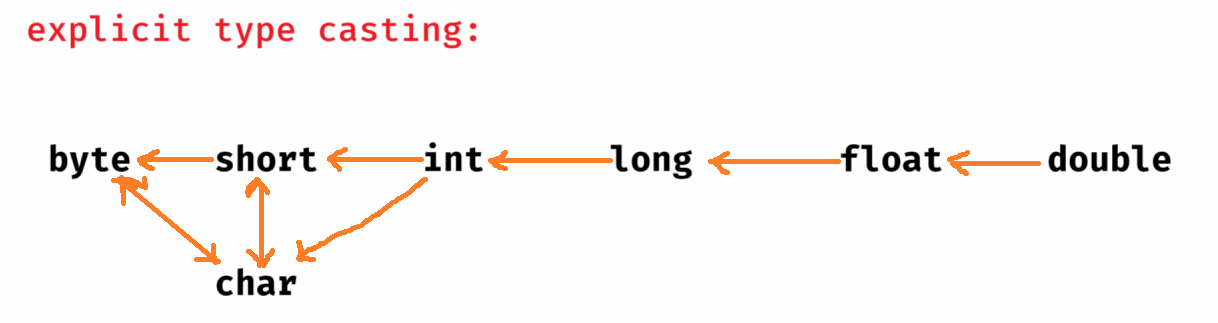
* Explicit type casting means, converting a variable from higher memory type to lower memory type.
* Explicit type casting should be done manually by writing the target data type with in parenthesis.

ex:

double x = 12.57;

int y = (int) x;

Sop(y); //output: 12



Note: boolean can not be converted to any other data type and also any other data type can not be converted to the boolean data type.

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

Control structures

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

1. conditional/selection control structures
2. Iterational/Loop control structures
3. Branching/Jump control structures

conditional control structures:

1. if statement
2. switch statement

Iterational control structures:

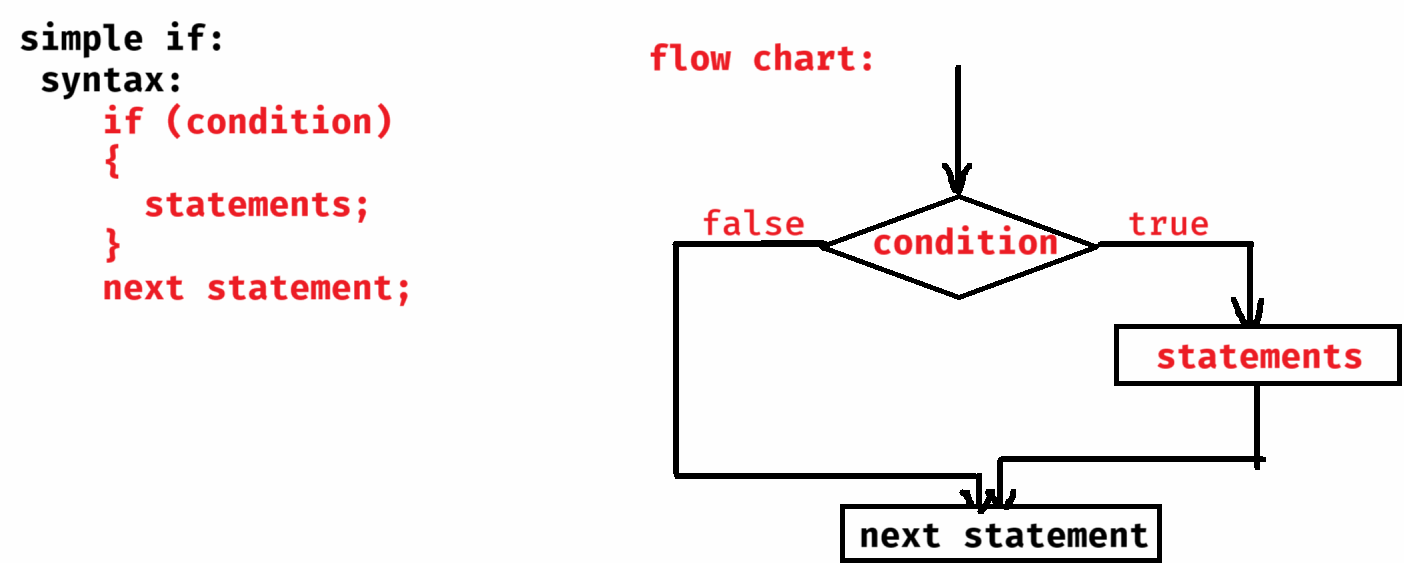
1. for loop
2. while loop
3. do while loop

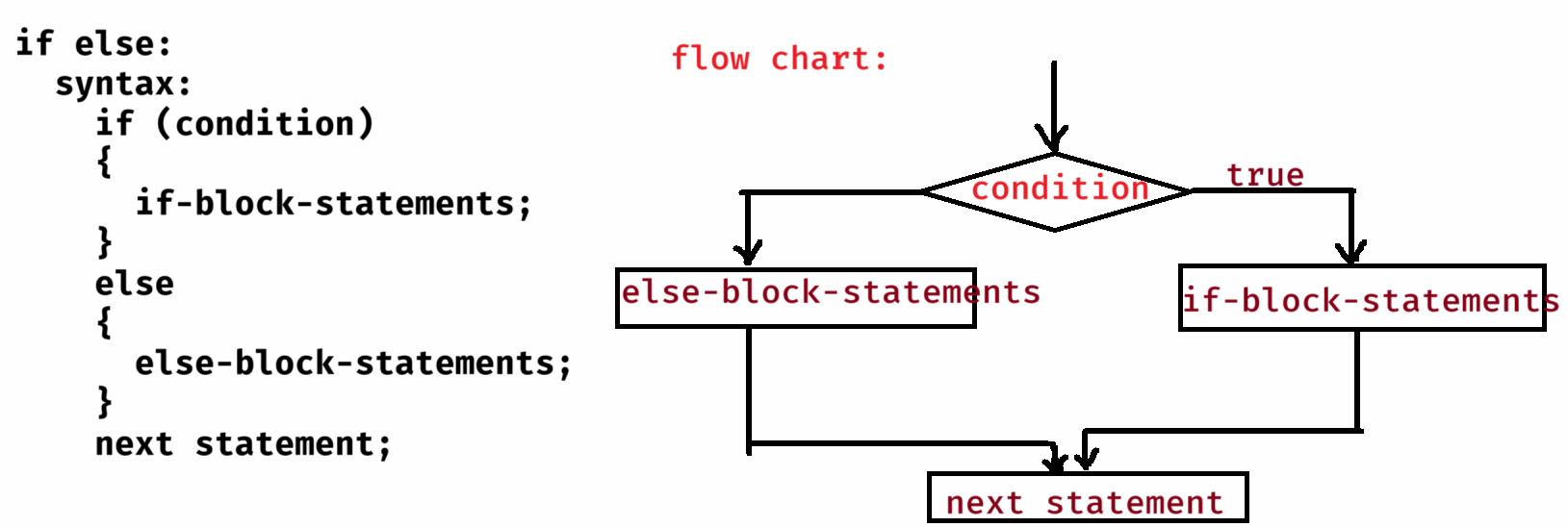
Branching control structures:

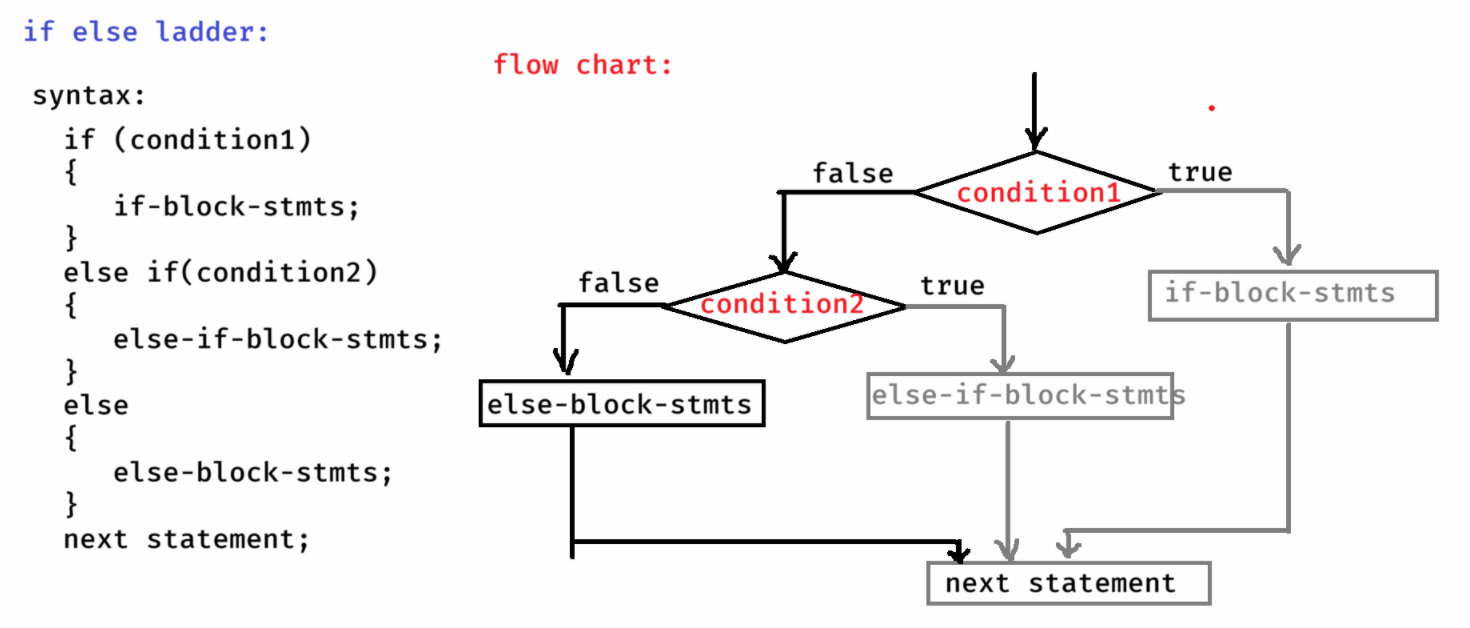
1. break
2. continue
3. return

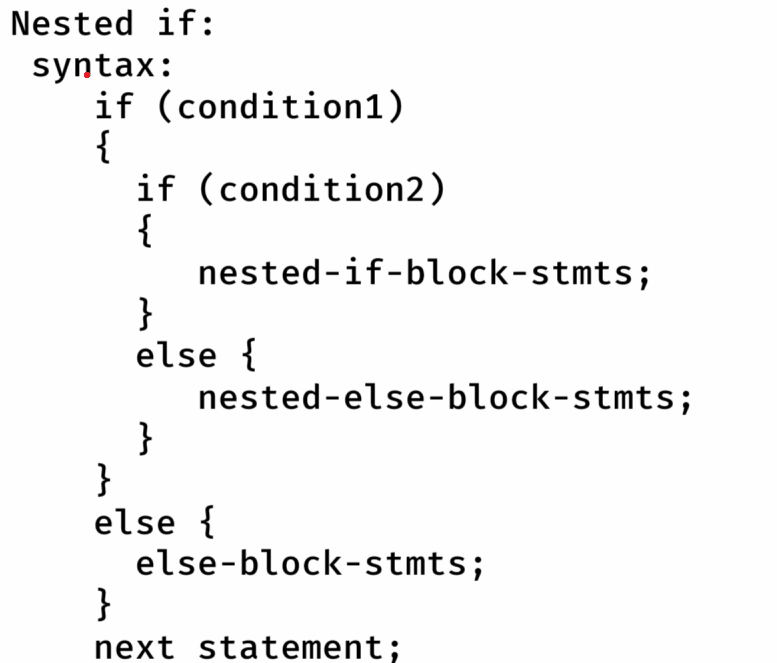
if statement:

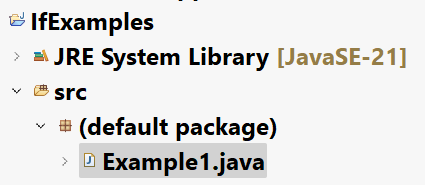
1. simple if
2. if else
3. if else ladder
4. nested if











Example1.java

**/\***

**\* This program will accept two integers from**

**\* the user.**

**\* If any integer is 0 then finds the addition**

**\* Otherwise, finds the multiplication.**

**\*/**

**import java.util.Scanner;**

**public class Example1 {**

**public static void main(String[] args) {**

**//connect Scanner to the keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter first number");**

**int a = scanner.nextInt();**

**System.*out*.println("Enter second number");**

**int b = scanner.nextInt();**

**if (a == 0 || b == 0) {**

**int c = a + b;**

**System.*out*.println("sum = " + c);**

**}**

**else {**

**int c = a \* b;**

**System.*out*.println("multiplication = " + c);**

**}**

**scanner.close();**

**}**

**}**

problem:

Take an integer value as input and implement the below conditions.

1. if the input is divisible by 3 then display “Zip”
2. if the input is divisible by 5 then display “Zap”.
3. if the input is divisible by 3 and 5 then display “Jar”.
4. Otherwise, display “Rar”.

Example2.java

**import java.util.Scanner;**

**public class Example2 {**

**public static void main(String[] args) {**

**//connect Scanner with keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter a number");**

**int n = scanner.nextInt();**

**if ( n % 3 == 0 && n % 5 == 0)**

**System.*out*.println("Jar");**

**else if ( n % 3 == 0)**

**System.*out*.println("Zip");**

**else if (n % 5 == 0)**

**System.*out*.println("Zap");**

**else**

**System.*out*.println("Rar");**

**scanner.close();**

**}**

**}**

problem:

Take distance(in kms) as input, then calculate delivery fee as below.

1. for First 3 kms, free delivery
2. For next 3 kms, Rs.12 per kilometer.
3. For remaining kilometers, Rs.18 per kilometer.

Example3.java

**import java.util.Scanner;**

**public class Example3 {**

**public static void main(String[] args) {**

**// connect Scanner with keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter distance");**

**int distance = scanner.nextInt();**

**int deliveryFee;**

**if (distance <= 3)**

**deliveryFee = 0;**

**else if (distance <= 6)**

**deliveryFee = (distance - 3) \* 12;**

**else**

**deliveryFee = 3 \* 12 + (distance - 6) \* 18;**

**System.*out*.println("Delivery Fee = " + deliveryFee);**

**scanner.close();**

**}**

**}**

problem:

Take 3 integers as input. (They are the sides of a triangle).

If these 3 sides forms a triangle, then

1. if they are same, display “Equilateral triangle”.
2. if any two sides are equal, display “Isosceles triangle”.
3. otherwise, display “Scalane triangle”.

Otherwise, display “The sides can not form a triangle”.

**import java.util.Scanner;**

**public class Example4 {**

**public static void main(String[] args) {**

**// connect Scanner with keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter side1 value of triangle");**

**int a = scanner.nextInt();**

**System.*out*.println("Enter side2 value of triangle");**

**int b = scanner.nextInt();**

**System.*out*.println("Enter side3 value of triangle");**

**int c = scanner.nextInt();**

**if (a + b > c || b + c > a || c + a > b)**

**{**

**if (a == b && b == c)**

**System.*out*.println("Equilateral triangle");**

**else if ( a == b || b == c || c == a )**

**System.*out*.println("Isosceles triangle");**

**else**

**System.*out*.println("Scalane triangle");**

**}**

**else**

**{**

**System.*out*.println("The given sides can not form a triangle");**

**}**

**scanner.close();**

**}**

**}**

problem:

Take a year as input and check is it a leap year or not.

Hint:

1. If a year is divisible by 4 and not divisible by 100 then it is a leap year.
2. If a year is divisible by 4 and 100 then it must be divisible by 400 also, then it is a leap year. Otherwise, not a leap year
3. If a year is not divisible by 4, then it is not a leap year.

Example5.java

**import java.util.Scanner;**

**public class Example5 {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter a year");**

**int year = scanner.nextInt();**

**if (year % 4 == 0)**

**{**

**if (year % 100 == 0)**

**{**

**if (year % 400 ==0)**

**{**

**System.*out*.println("Leap Year!");**

**}**

**else**

**{**

**System.*out*.println("Not a Leap year!");**

**}**

**}**

**else**

**{**

**System.*out*.println("Leap Year!");**

**}**

**}**

**else**

**{**

**System.*out*.println("Not a Leap Year!");**

**}**

**scanner.close();**

**}**

**}**

problem:

Take 3 integers as input and find the biggest number.

Example6.java

**import java.util.Scanner;**

**public class Example6 {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter first number");**

**int a = scanner.nextInt();**

**System.*out*.println("Enter second number");**

**int b = scanner.nextInt();**

**System.*out*.println("Enter third number");**

**int c = scanner.nextInt();**

**if (a > b && a > c)**

**{**

**System.*out*.println("Biggest number : " + a);**

**}**

**else if (b > c)**

**{**

**System.*out*.println("Biggest number : " + b);**

**}**

**else**

**{**

**System.*out*.println("Biggest number : " + c);**

**}**

**scanner.close();**

**}**

**}**

problem :

Take input as paise and display the output in

rupees, fity paise, twenty five paise, ten paise and one paise.

ex:

paise = 287

output:

2 – rupees

1 – fifty paise

1 – twentyfive paise

1 – ten paise

2 – one paise

Example7.java

**import java.util.Scanner;**

**public class Example7 {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter the paise : ");**

**int paise = scanner.nextInt();**

**int rupees = 0, fifty = 0, twentyfive = 0, ten = 0, one = 0;**

**if (paise >= 100) {**

**rupees = paise / 100;**

**paise = paise % 100;**

**}**

**if (paise >= 50) {**

**fifty = paise / 50;**

**paise = paise % 50;**

**}**

**if (paise >= 25) {**

**twentyfive = paise / 25;**

**paise = paise % 25;**

**}**

**if (paise >= 10) {**

**ten = paise / 10;**

**paise = paise % 10;**

**}**

**one = paise;**

**System.*out*.println(rupees + " -- rupees");**

**System.*out*.println(fifty + " -- fifty paise");**

**System.*out*.println(twentyfive + " -- twentyfive paise");**

**System.*out*.println(ten + " -- ten paise");**

**System.*out*.println(one + " -- one paise");**

**scanner.close();**

**}**

**}**

switch statement

* When you want to write multiple conditions on the same variable and the condition is equals, then you can use switch statement, to make your code more readable and understandable.

syntax:

switch(variable/expression)

{

case value1:

statements;

break;

case value2:

statements;

break;

. . .

. . .

case valueN:

statements;

break;

default:

statements;

}

* The variable in switch must be integer type(byte/short/int/long) or character type or string type.
* The case values are also must be integer type or character type or string type.
* The switch statement compares the variable/expression value with the case value. If matched then the corressponding statements are executed.
* If no case value is matched then default will be executed.
* Writing the default is optional.

ex1:

int rating = 2;

switch(rating)

{

case 1:

Sop(“bad”);

break;

case 2:

Sop(“good”);

break;

default:

Sop(“wrong option”);

}

output:

good

ex2:

int rating = 2;

switch(rating)

{

case 1:

Sop(“bad”);

break;

case 2:

Sop(“good”);

default:

Sop(“wrong option”);

}

output:

good

wrong option

ex3:

int rating = 3;

switch(rating)

{

default:

Sop(“wrong option”);

case 1:

Sop(“bad”);

case 2:

Sop(“good”);

break;

}

output:

wrong option

bad

good

ex4:

boolean status = true;

switch(status) {

case true: Sop(“Success”); break;

case false: Sop(“Failed”); break;

}

output:

error

ex5:

char ch = ‘i’;

switch(ch)

{

case ‘a’:

case ‘e’:

case ‘i’:

case ‘o’:

case ‘u’:

Sop(“vowel”);

break;

default:

Sop(“consonant”);

}

output:

vowel

ex6:

String color = “teal”;

switch(color)

{

case “blue”:

Sop(“blue”);

break;

case “Teal”:

Sop(“teal”);

case “red”:

Sop(“Red”);

break;

}

output:

doesn’t give any output

ex7:

int x = 5;

switch(x \* x) {

case 5: Sop(“Five”);

case 25: Sop(“Twenty Five”);

break;

case 50: Sop(“Fifty”);

}

output:

Twenty Five

ex8:

int day=1;

switch(day) {

case 1: Sop(“Monday”); break;

case 2: Sop(“Tuesday”); break;

default: Sop(“Sunday”);

}

Sop(“All days are good”);

output:

Monday

All days are good

ex9:

int day=1;

switch(day) {

case 1: return “Monday”;

case 2: return “Tuesday”;

default: return “Sunday”;

}

Sop(“All days are good”);

output:

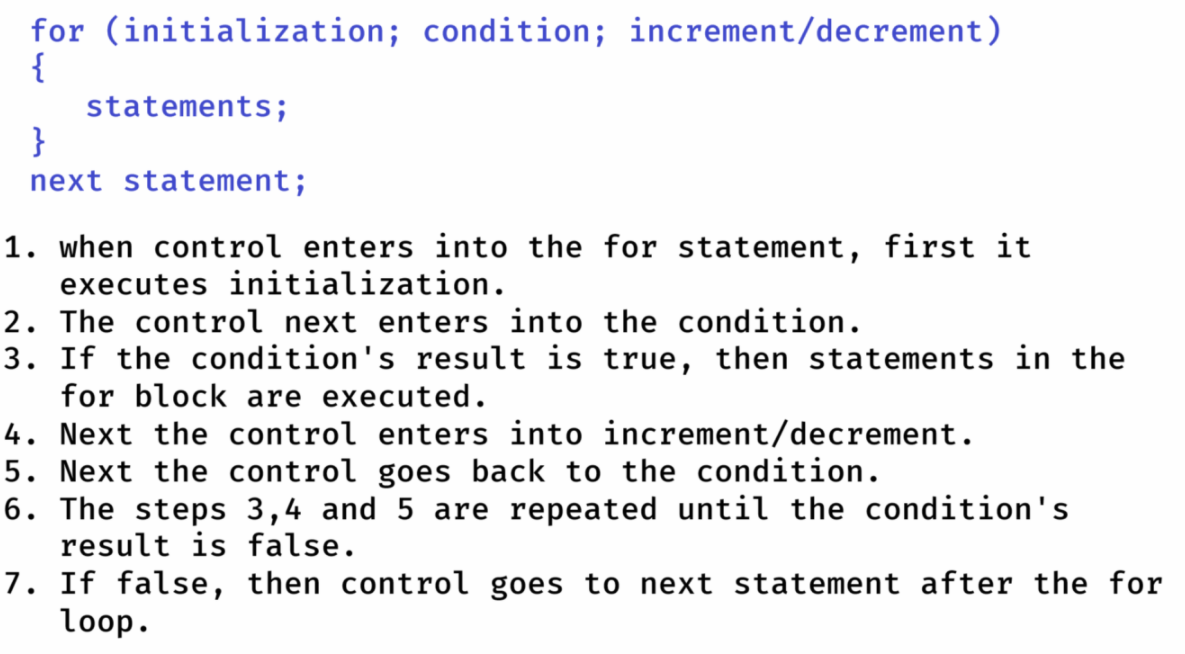
return statement will make the control to exit the

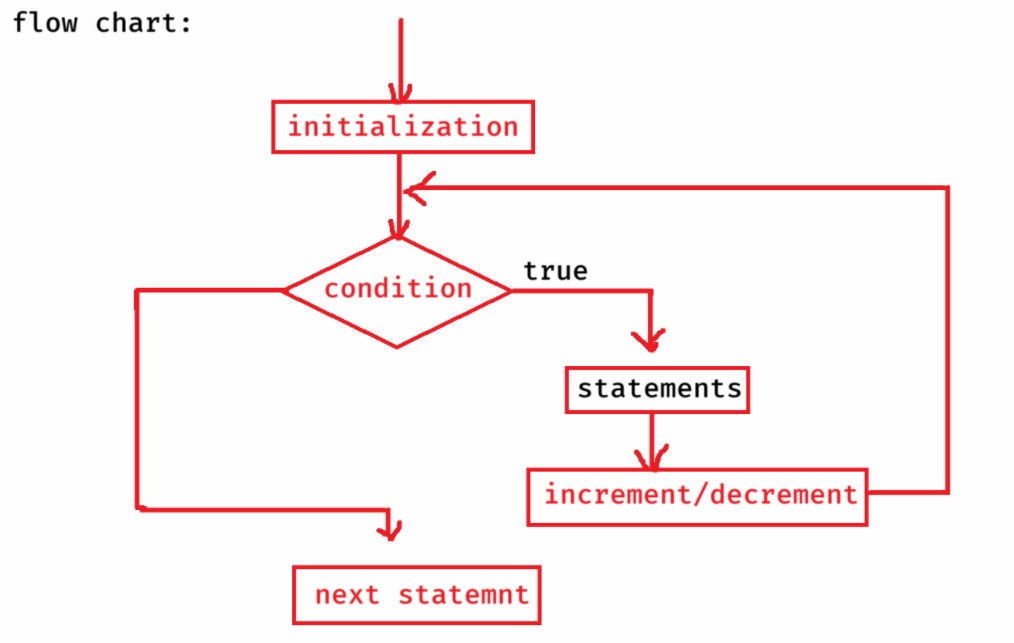
current method entirely and goes back to the caller.

So, the next statement after switch will not be executed.

for loop

* A loop is used to iterate the control over a group of statements repeatedly for multiple times.





ex1:

for (int i=1; i<=3; i++)

{

S.o.p(i);

}

output:

1

2

3

ex2:

for (int i=1; i<=3; i++)

{

S.o.p(i);

}

S.o.p(i);

output:

error

ex3:

int i=1;

for( ; i<=3; i++)

S.o.p(i);

S.o.p(i\*i);

output:

1

2

3

16

ex4:

int i=10;

for( ; i<=15; i++);

S.o.p(i);

output:

16

ex5:

int k = 5;

for(;;)

{

if(k==8)

break;

k++;

Sop(k);

}

output:

6

7

8

ex6:

int i=1;

for(; i<=3; i++)

{

if(i == 3)

return;

S.o.p(i);

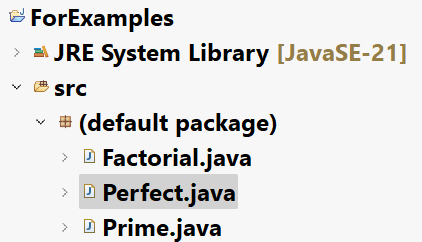
}

S.o.p(i);

output:

1

2

****

**import java.util.Scanner;**

**/\*\***

**\* This program finds the factorial of**

**\* a given number**

**\* ex: n = 5**

**\* output: 120**

**\*/**

**public class Factorial {**

**public static void main(String[] args)**

**{**

**//connect scanner to keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter a number");**

**int n = scanner.nextInt();**

***findFactorial*(n); //calling point**

**scanner.close();**

**}**

**private static void findFactorial(int n) //definition point**

**{**

**int result = 1;**

**for (int i = 1; i <= n; i++)**

**{**

**result = result \* i;**

**}**

**System.*out*.println("Factorial = " + result);**

**}**

**}**

Prime.java

**import java.util.Scanner;**

**/\*\***

**\* This program checks whether a given number is**

**\* prime or not.**

**\* prime number : A natural number which contains only**

**\* 2 factors(one and itself) is called**

**\* a prime number.**

**\* ex:**

**\* n = 5**

**\* output: Yes, it is a prime number**

**\* n = 14**

**\* output: No, it is not a prime number**

**\*/**

**public class Prime {**

**public static void main(String[] args) {**

**//connect Scanner to the keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter a number to check is it prime nor not");**

**int n = scanner.nextInt();**

**boolean flag = *isPrime*(n);**

**if(flag == true) {**

**System.*out*.println("Yes, it is a prime number");**

**}**

**else {**

**System.*out*.println("No, it is not a prime number");**

**}**

**scanner.close();**

**}**

**private static boolean isPrime(int n) {**

**boolean flag = true;**

**for( int i = 2; i <= (int) Math.*sqrt*(n); i++)**

**{**

**if (n % i == 0)**

**{**

**flag = false;**

**break;**

**}**

**}**

**return flag;**

**}**

**}**

Perfect.java

**import java.util.Scanner;**

**/\*\***

**\* This program checks whether a given number is**

**\* perfect number or not.**

**\* Definition: If sum of the factors of a number excluding the given**

**\* number is equal to the same number then it is**

**\* called a perfect number.**

**\* example:**

**\* n = 6**

**\* sum of the factors: 1 + 2 + 3 = 6**

**\* Yes, it is perfect number**

**\* example:**

**\* n = 24**

**\* sum of the factors: 1 + 2 + 3 + 4 + 6 + 8 + 12 = 36**

**\* No, it is not a perfect number**

**\*/**

**public class Perfect {**

**private static boolean checkPerfect(int n) {**

**int sum = 0;**

**for (int i = 1; i <= n / 2; i++)**

**{**

**if(n % i == 0) {**

**sum = sum + i;**

**}**

**}**

**if (sum == n)**

**return true;**

**else**

**return false;**

**}**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter a number to check is it a perfect number");**

**int n = scanner.nextInt();**

**boolean flag = *checkPerfect*(n);**

**if (flag == true)**

**System.*out*.println("Yes, it is a perfect number");**

**else**

**System.*out*.println("No, it is not a perfect number");**

**scanner.close();**

**}**

**}**

Fibonacci.java

**import java.util.Scanner;**

**/\*\***

**\* This program prints the fibonacci series of**

**\* n terms.**

**\* The first two terms of the fibonacci series are,**

**\* 0 and 1**

**\* The next term is calculated as sum of the previous**

**\* two terms.**

**\* for example,**

**\* if n = 5, the terms are,**

**\* 0 1 1 2 3**

**\* if n = 10, the terms are,**

**\* 0 1 1 2 3 5 8 13 21 34**

**\*/**

**public class Fibonacci {**

**public static void main(String[] args) {**

**//connect Scanner with keyboard input**

**Scanner scanner = new Scanner(System.*in*);**

**System.*out*.println("Enter the number of terms to display the fibonacci series");**

**int n = scanner.nextInt();**

***printFibonacci*(n);**

**scanner.close();**

**}**

**private static void printFibonacci(int n) {**

**int firstTerm = 0;**

**int secondTerm = 1;**

**if(n == 1) {**

**System.*out*.println(firstTerm);**

**}**

**else if(n == 2) {**

**System.*out*.print(firstTerm + " " + secondTerm);**

**}**

**else {**

**System.*out*.print(firstTerm + " " + secondTerm + " ");**

**for(int i = 3; i <= n; i++) {**

**int nextTerm = firstTerm + secondTerm;**

**System.*out*.print(nextTerm+" ");**

**firstTerm = secondTerm;**

**secondTerm = nextTerm;**

**}**

**}**

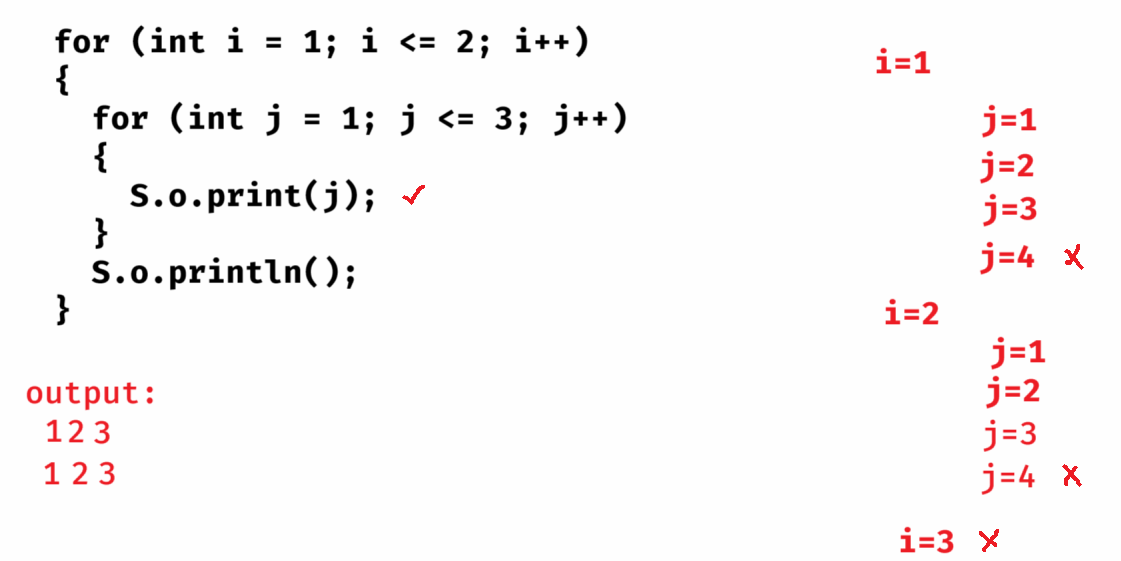
**}**

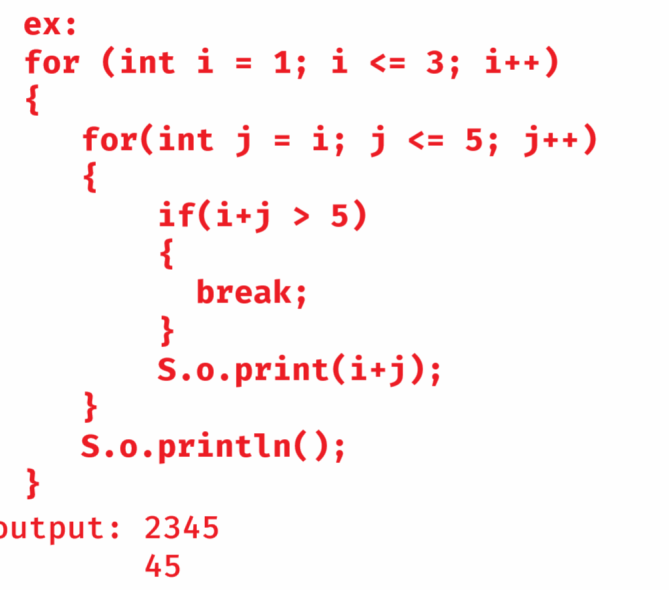
**}**

Nested for loop:

---------------

* writing a for loop inside a for loop is called as nested for loop.
* For each iteration of the outer loop, the inner loop will be completely executed.





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

Pattern programs

* we have to use nested for loops to display the patterns
* we have to use outer for loop for rows.
* we have to use inner for loop for columns. We need to iterate the inner loop by finding the relationship between rows and columns.
* we have to print the star in the inner for loop.