**Day-11**

**31-01-2025**

**====================**

**Control Statements:**

**===============**

**Sequential Execution:**

**=====================**

-> The normal java program behavior is "sequential execution".

-> The program can execute statement-wise is called "Sequential Execution".

class MyClass{

public static void main(String[] args)

{

int a, b;

a = 100;

b = 200;

System.out.println("Hi");

int c = a + b;

System.out.println(c);

}

}

javac

java

-> Some times we need to execute the program based on the selection or the specific block of code want to execute repeatedly, in this case we need "control statements".

-> There three types of control statements:

1) Conditional statements/Selection statements

2) Loop Statements/Iterative Statements

3) Jump Statements/Transfer Statements

**1) Conditional statements/Selection statements**

**================================**

-> 4-different types of conditional statements:

1) Simple If statement

2) if-else statement

3) if-else if- else ladder

4) Nested if-else

-> We have one more selection statement:

called as "switch" statement

**1) Simple If statement**

**===============**

Syntax:

if(condition)

{

statement-1;

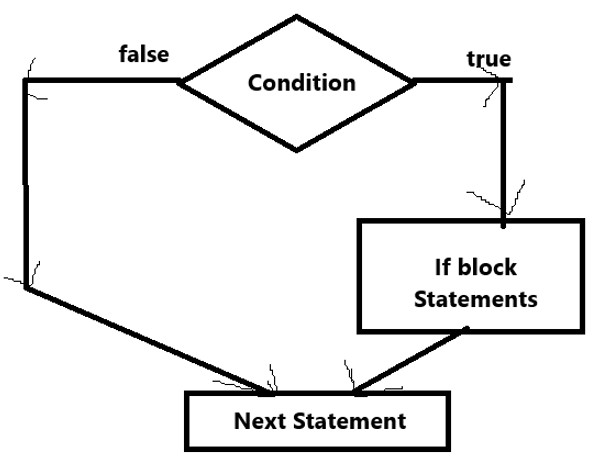
statement-2;

}

next statement;

**Flow-Diagram:**

**===========**



Q: WAP TO ACCEPT THE DISTANCE FROM THE USER TO RESTAURENT. IF THE DISTANCE IS BELOW 5KM THEN THERE IS NO DELIVERY CHARGES.

import java.util.Scanner;

public class MainClass {

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

System.out.println("Enter a distance in km:");

float distance = scan.nextFloat();

if(distance < 5.0f) {

System.out.println("Hi user,");

System.out.println("Your Distance from Restaurent is = "+distance);

System.out.println("Your Delivery Charges are:");

double deliveryCharges = 0.0; // block variable

System.out.println(deliveryCharges);

}

// System.out.println(deliveryCharges);

System.out.println("Restaurent not offering free delivery for you.");

}

}

Note:

=====

Block Variable:

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

The Variable which can be defined in any block, it has the scope of the block. In which block we have defined within the same block we can access. If we can try to use that block variable in out side the block scope, then we can get an error.

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

Assignment:

==========

1) WAP TO ACCEPT A NUMBER CONVERT THAT NUMBER TO POSITIVE IF IT IS ENETERED AS NEGATIVE ONLY.

int n = -23;

if(n < 0){

n = -n;

}

**Day-12**

**03-02-2025**

**===============**

Note:

=====

if we have only one statement if block then: {} is optional

when we have more than one statement in if-block, then: {} is mandatory.

ex:

number is positive or negative

number = 9

if(number > 0)

number is positive

if(number < 0)

number is negative

next level

**2) If else statement**

**=============**

-> if else statement can use to overcome the drawback of simple if.

-> drawback of simple if :

time consuming

when the first condition is "true", then also it can able to continue to check with remaining conditions.

Syntax:

if(condition){

statement-1;

statement-2;

}

else{

statement-3;

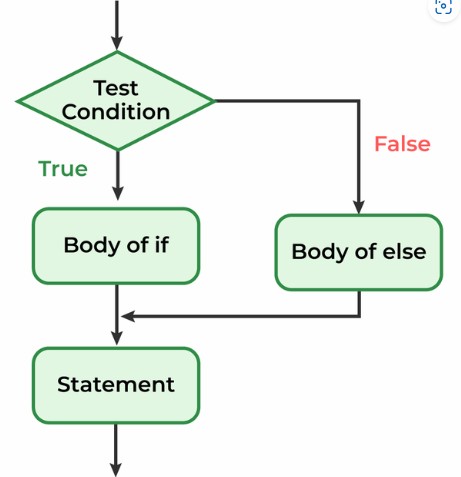
statement-4;

}

next statement;

**Flow-Diagram:**

**==========**



**function Vs method:**

**=============**

Similarity:

======

definitions are almost similar

difference:

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

function ==> can always define in outside the class

method ==> can always define inside the class.

Q: Is java support functions?

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

Ans: No, because functions generally allow to define in outside the class. But the java does not allow to write even the single line of code without the class.

-> Every method definition can allow to the above of main() or below of main() method.

-> The method definition is consisting of two parts:

1) Method header

2) Method Body

-> Method header describe three things:

1) Name of the method

2) return type of the method

3) parameters

-> Method body describe:

the logic to perform.

package pack1;

import java.util.Scanner;

public class NumberTest {

public static void main(String[] args) { // calling method

Scanner scan = new Scanner(System.in);

System.out.println("Enter some value:");

int number = scan.nextInt();

positiveOrNegative(number); // method calling

}

public static void positiveOrNegative(int num) { //called method

if(num > 0) {

System.out.println(num+"is a positive value.");

}

else {

System.out.println(num+"is a negative value.");

}

}

}

**Day-13**

**04-02-2025**

**================**

**3) if else if else ladder:**

**================**

**Syntax:**

**if(condition1)**

**{**

**statement-1;**

**statement-2;**

**}**

**else if(condition2)**

**{**

**statement-3;**

**statement-4;**

**}**

**else if(condition3)**

**{**

**statement-5;**

**statement-6;**

**}**

**else{**

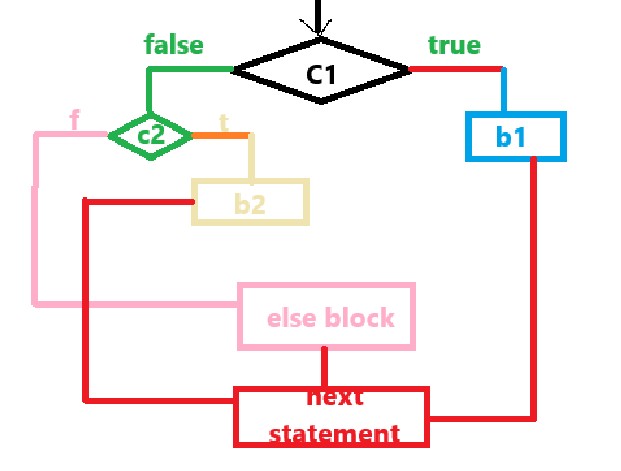
**statement-7;**

**statement-8;**

**}**

**Next Statement;**

**Flow Diagram:**



**Q-1:**

**====**

**WAP in java to find the biggest number among three given integers.**

**===================================================================**

**import java.util.Scanner;**

**class FindBig{**

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

**{**

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

**System.out.println("Enter First value:");**

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

**System.out.println("Enter Second value:");**

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

**System.out.println("Enter Third value:");**

**int z = scan.nextInt();**

**findingBiggest(x,y,z);**

**}**

**public static void findingBiggest(int x, int y, int z)**

**{**

**int big;**

**if(x > y && x > z)**

**{**

**big = x;**

**System.out.println("The Biggest number among three integers is = "+big);**

**}**

**else if(y > z)**

**{**

**big = y;**

**System.out.println("The Biggest number among three integers is = "+big);**

**}**

**else{**

**big = z;**

**System.out.println("The Biggest number among three integers is = "+big);**

**}**

**}**

**}**

**Q-2:**

**====**

**WAP in java to check whether the given number is even or odd.**

**Hint:**

**=====**

If a number is evenly divided with '2' then that number is called as "Even number" otherwise, it can be called as "Odd number".

**import java.util.Scanner;**

**class EvenOdd{**

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

**{**

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

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

**int number = scan.nextInt();**

**evenOddTest(number);**

**}**

**public static void evenOddTest(int n)**

**{**

**if(n % 2 == 0)**

**{**

**System.out.println(n+" is an even number");**

**}**

**else**

**{**

**System.out.println(n+" is an odd number");**

**}**

**}**

**}**

**=========================================================**

**Note:**

**=====**

Actual Parameter can always define in method call.

Formal Parameter can always define in method definition.

**Assignment:**

**===========**

**1) WAP in java to define username and password as input.**

**if username is "admin" and password is "admin123", then display "login successful".**

**otherwise, display "login fail".**

**Hint:**

**=====**

**username == "admin" and password == "admin123"**

**Day-14**

**05-02-2025**

**==============**

**Behavior of == (Equal Operator)**

**=======================**

1) == (equal operator) can compare the values of two variables if the type of data as primitive type.

2) If the data is as reference type, equal operator (==) can compare the addresses/identity of objects

**next() Vs nextLine()**

**==============**

next() cannot understand the white spaces in the string.

If the white space is occurred in a string after the white space, next() can stop to accept the characters.

nextLine() can accept white spaces also while reading of the string.

**== Vs equls() Vs equalsIgnoreCase()**

**=========================**

== is an operator can compare the two identities (address) of string objects when you can define the string in run-time(dynamic definition)/compile time.

**Note:**

**====**

If two strings are defined in compile time with same value, then: both string objects can store in same address location of heap memory.

**equals() can compare internal content (value) of two string objects. It can return "true" if two string objects with same data otherwise it can return "false".**

Syntax:

String-object-name.equals("data2")

**equalsIgnoreCase() is same as equals() but it can ignore the case of the text.**

Syntax:

String-object-name.equalsIgnoreCase(data)

**WAP in java to define username and password as input.**

**if username is "admin" and password is "admin123", then display "login successful".**

**otherwise, display "login fail".**

import java.util.Scanner;

class LoginSystem{

public static void main(String[] args)

{

Scanner s = new Scanner(System.in);

System.out.println("Enter User name:");

String userName = s.nextLine();

System.out.println("Enter Password:");

String password = s.nextLine();

if(userName.equalsIgnoreCase("admin") && password.equalsIgnoreCase("admin123"))

{

System.out.println("Login Success");

}

else

{

System.out.println("Login Fail.");

}

}

}

**4) Nested If Else Statement:**

**===================**

Syntax:

if(condition1)

{

if(condition2)

{

b1 statements;

}

else

{

b2 statements;

}

}

else

{

if(condition3)

{

b3 statements;

}

else

{

b4 statements;

}

}

**Q: WAP IN JAVA TO ACCEPT THREE INTEGERS AND FIND THE SMALLEST NUMBER AMONG GIVEN THREE NUMBERS.**

**============================================**

import java.util.Scanner;

class FindSmall{

public static void findingSmallerValue(int x, int y, int z)

{

int small;

if(x < y)

{

if(x < z)

{

small = x;

System.out.println("The "+small+" is smaller");

}

else

{

small = z;

System.out.println("The "+small+" is smaller");

}

}

else

{

if(y < z)

{

small = y;

System.out.println("The "+small+" is smaller");

}

else

{

small = z;

System.out.println("The "+small+" is smaller");

}

}

}

public static void main(String[] args)

{

int x,y,z;

Scanner s = new Scanner(System.in);

System.out.println("Enter values for x,y and z:");

x = s.nextInt();

y = s.nextInt();

z = s.nextInt();

findingSmallerValue(x,y,z);

}

}

Assignment:

===========

WAP IN JAVA TO ACCEPT AN INTEGER FOR AN YEAR VARIABLE. CHECK WHETHER THE GIVEN YEAR IS LEAP YEAR OR NOT.

**Day-15**

**08-02-2025**

**================**

/\* WAP TO ACCEPT A YEAR AS AN INPUT AND CHECK

WHETHER THE GIVEN YEAR IS LEAP YEAR OR NOT \*/

/\*

i) if a year is divided with 4

but not with 100 ==> leap year

ex: 4, 8, 12, 16, ....

ii) if year is divisible with 100

and also with 400 ==> leap year

ex: 2000, 2400,

\*/

**Solution-1:**

**========**

import java.util.Scanner;

class LeapYearOrNot{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter the value for year:");

int year = scan.nextInt();

boolean result = isLeapYear(year);

if(result == true){

System.out.println(year+" is leap year");

}

else{

System.out.println(year+" is not leap year");

}

}

public static boolean isLeapYear(int y){

if(y % 4 == 0){

if(y % 100 == 0){

if(y % 400 == 0){

return true;

}

else{

return false;

}

}

else{

return true;

}

}

else{

return false;

}

}

}

**Logical and (&&)**

**===========**

-> binary operator

Truth table:

========

a b a && b

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

true true true

true false false

false true false

false false false

-> if both inputs are "true" then: logical and returns/gives "true"

-> if any of the input "false" then: logical and returns/gives "false"

working phenomena:

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

-> if first input is "true" then: the logical and operator can decode the second input.

if the second input is also "true", then: logical and output ==> "true".

if the second input is "false" then" output ==> "false"

-> If the first input is "false" then: the logical and can give the output as "false" without even decoding the second input.

**Logical or(||)**

**==========**

-> binary operator

Truth table:

a b a || b

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

true true true

true false true

false true true

false false false

-> if any of the input is "true" then: the logical or can give "true" as an output.

-> if both inputs are "false", then only the output is "false"

Working Phenomena:

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

-> if first input is "true", the output is "true" without decoding of the second input.

-> if the first input is "false", output is based on second input.

if second input is "true" ==> output : "true"

otherwise: output ==> "false".

**Solution-2:**

**========**

import java.util.Scanner;

class LeapYearOrNot{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter the value for year:");

int year = scan.nextInt();

boolean result = isLeapYear(year);

if(result == true){

System.out.println(year+" is leap year");

}

else{

System.out.println(year+" is not leap year");

}

}

public static boolean isLeapYear(int y){

if(y % 4 == 0 && y % 100 != 0 || y % 400 == 0){

return true;

}

else{

return false;

}

}

}

**Assignment:**

**========**

1) WAP TO ACCEPT THREE NUMBERS AND FIND WHICH IS BIGGEST AMONNG THE GIVEN NUMBERS.

Hint:

12,13,14 ==> 14 is bigger

12,13,13 ==> b and c ==> bigger

12,12,12 ==> all are equal

**Solution-1:**

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

import java.util.Scanner;

class FindLarger{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter three integers:");

int x = scan.nextInt();

int y = scan.nextInt();

int z = scan.nextInt();

findingBigger(x,y,z); // here x, y and z are actual arguments

}

public static void findingBigger(int x, int y, int z)

{

// here x, y and z ==> formal arguments

int big = 0;

if(x > y && x > z)

{

System.out.println("x is bigger");

}

else if(y > x && y > z)

{

System.out.println("y is bigger");

}

else{

if(x == y && x > z)

{

System.out.println("x and y both are larger");

}

else if(y == z && y > x)

{

System.out.println("y and z both are larger");

}

else if(z == x && z > y)

{

System.out.println("x and z both are larger");

}

else{

System.out.println("x, y and z are equal");

}

}

}

}

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

**Solution-2:**

**========**

import java.util.Scanner;

class FindLarger{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Enter three integers:");

int x = scan.nextInt(); // 12

int y = scan.nextInt(); // 12

int z = scan.nextInt(); // 10

findingBigger(x,y,z); // here x, y and z are actual arguments

}

public static void findingBigger(int x, int y, int z)

{

// here x, y and z ==> formal arguments

if(x >= y && x >= z){

if(x == y){

System.out.println("x and y both are larger");

}

else if(x == z)

{

System.out.println("x and z both are larger");

}

else{

System.out.println("x is the largest");

}

}

else if(y >= z && y >= x){

if(y == z){

System.out.println("y and z both are larger");

}

else if(y == x)

{

System.out.println("y and z both are larger");

}

else{

System.out.println("y is the largest");

}

}

else{

if(z == x)

{

System.out.println("z and x both are larger");

}

else if(z == y)

{

System.out.println("z and y both are larger");

}

else{

System.out.println("z is the largest");

}

}

}

}

2) WAP TO TAKE AN INTEGER AS AN INPUT AND DO THE FOLLOWING:

I) IF IT IS DIVISIBLE WITH '3' ==> DISLAY AS "ZIP"

II) IF IT IS DIVISIBLE WITH '5' ==> DISPLAY AS "ZAP"

III) IF IT IS DIVISIBLE WITH BOTH '3' AND '5' ==> "ZIPZAP"

IV) OTHERWISE ==> "NOT WITH 3 AND NOT 5".

import java.util.Scanner;

class CheckingConditions{

public static void main(String[] args)

{

Scanner s = new Scanner(System.in);

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

int number = s.nextInt();

if(number % 3 == 0 && number % 5 == 0)

{

System.out.println("Zipzap");

}

else if(number % 3 == 0)

{

System.out.println("Zip");

}

else if(number % 5 == 0)

{

System.out.println("Zap");

}

else{

System.out.println("The number not divisible with 3 and 5 both");

}

}

}

3) WAP TO MULTIPLY THREE INTEGERS. HOWEVER IF ONE OF NUMBER IS '7' THEN CONSIDER THE VALUES REIGHT TO IT FOR MULTIPLICATION. IF '7' IS IN LAST, THEN DISPLAY '-1' AS OUTPUT.

HINT:

3,4,5 ==> 3 \* 4 \* 5

7,2,3 ==> 2 \* 3

1,7,9 ==> 9

1,2,7 ==> -1

import java.util.Scanner;

class MultiplyThreeNumbers{

public static void main(String[] args)

{

Scanner s = new Scanner(System.in);

System.out.println("Enter three numbers:");

int a = s.nextInt();

int b = s.nextInt();

int c = s.nextInt();

int product = 0;

if(a == 7){

product = b \* c;

System.out.println("The Product = "+product);

}

else if(b == 7)

{

product = c;

System.out.println("The Product = "+product);

}

else if(c == 7)

{

product = -1;

System.out.println("The Product = "+product);

}

else{

product = a \* b \* c;

System.out.println("The Product = "+product);

}

}

}

4) WAP TO CALCULATE DELIVERY COST BY FOLLOWING:

I) IF DISTANCE FROM THE RESTAURENT IS <= 3KM, NO DELIVERY COST

II) FOR NEXT 3 KM, DELIVERY COST IS Rs. 3 PER KM

III) FOR REMAINING, DELIVERY COST IS Rs. 5 PER KM.

EX: IF A PERSON

DISTANCE FROM RESTAURENT ==> 1 KM

ORDER COST ==> 100 + 0 ==> 100

PERSON CAN 5 KM FROM RESTAURENT

0 TO 3 ==> NO CHARGES

4 TO 5 ==> 2 X 3 ==> 6

ORDER ==> 100 + 6 ==> 106

PERSON CAN 10KM DISTANCE FROM RESTAURENT

0 TO 3 ==> NO CHARGES

4 TO 6 ==> 3 \* 3 ==> 9

7 TO 10 ==> 4 x 5 ==> 20

DELIVERY CHARGES ==> 0 + 9 + 20 ==> 29

ORDER ==> 100

COST ==> 100 + 29 ==> 129

import java.util.Scanner;

class OrderCost{

public static void main(String[] args)

{

Scanner s = new Scanner(System.in);

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

int distance = s.nextInt();

System.out.println("Enter the order cost:");

double orderCost = s.nextDouble();

double deliveryFee = 0.0;

double orderPayment = 0.0;

if(distance <= 3)

{

deliveryFee = 0;

orderPayment = orderCost + deliveryFee;

}

else if(distance <= 6){

deliveryFee = (distance - 3) \* 3;

orderPayment = orderCost + deliveryFee;

}

else{

deliveryFee = (distance - 3) \* 3 + (distance - 6) \* 5;

orderPayment = orderCost + deliveryFee;

}

System.out.println("The Total Order Payment = "+orderPayment);

}

}