Control Statements & Operators

A statement is a line of code that ends with " ; "

Statements is of 2 types....

1) Sequential statements

2) control statements

Sequential :

The statements which execute one by one from top to bottom....

class Sequential

{

public static void main(String[] args)

{

int a = 10 ;

int b = 20 ;

int c = a+b ;

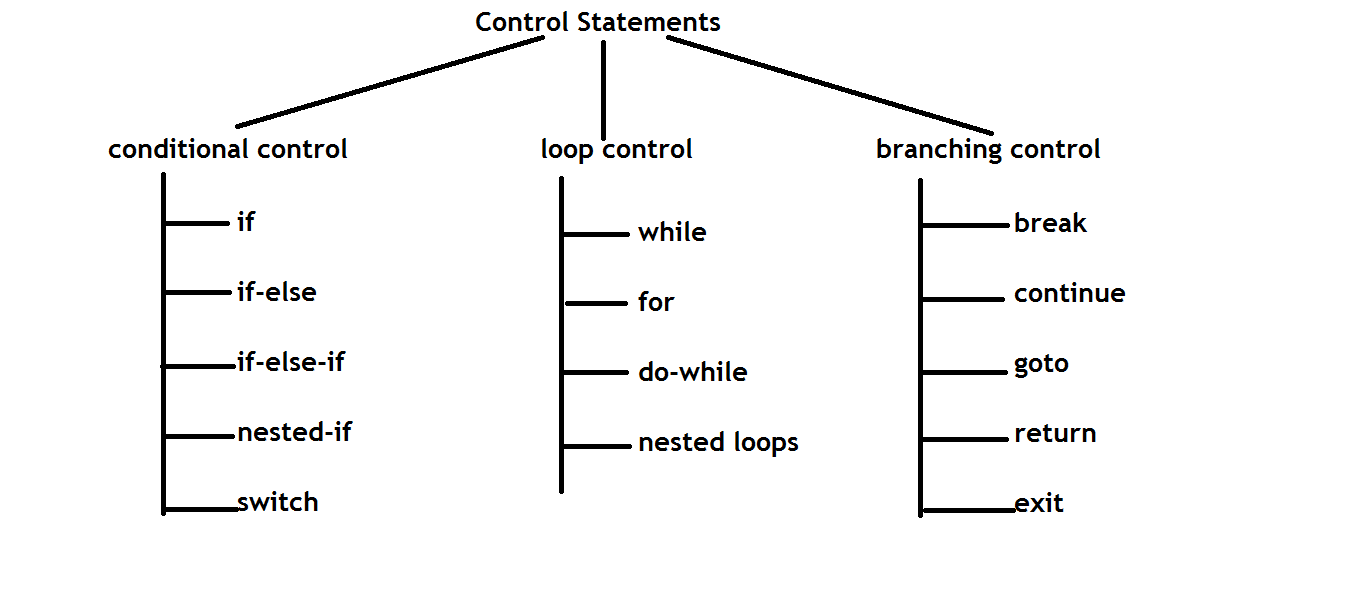
System.out.println("c value : "+c);

}

}

Control statements :

The statements which are executing randomly and repeatedly depends on condition.....



operator :

Operator is a symbol.

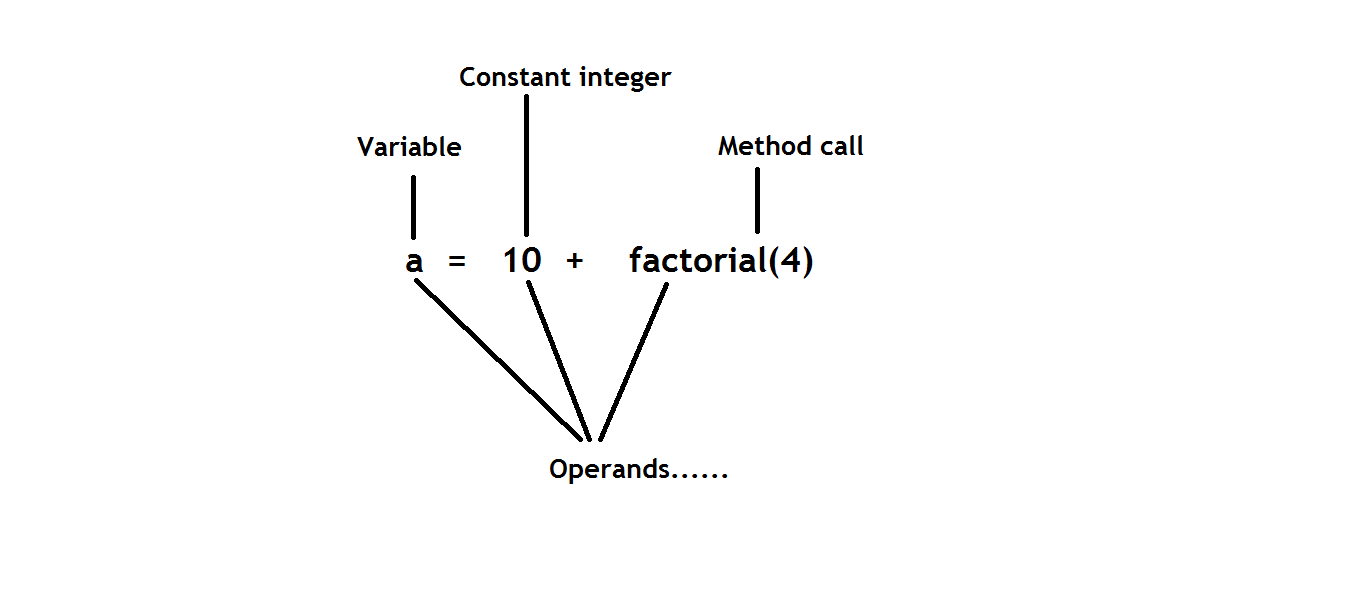
It performs operation on one or more operands...

Operators classified into 3 types

Unary : Performs operation on Single Operand.

Binary : Performs operation on two operands

Ternary : On 3 operands....



if : Executes a block on some condition....

syntax :

if(condition)

{

...statements ;

}

Examples :

class Demo

{

public static void main(String[] args)

{

if(true)

{

System.out.println("Inside if block....");

}

System.out.println("Outside to if block....");

}

}

class Demo

{

public static void main(String[] args)

{

if(false)

{

System.out.println("Inside if block....");

}

System.out.println("Outside to if block....");

}

}

/\*

Note : If we want to write only one statement inside a block, no need to use braces....

The statement which is followed by if, treats as block statement....

\*/

class Demo

{

public static void main(String[] args)

{

if(false)

System.out.println("Block statement");

System.out.println("Outside statement....");

}

}

if - else :

Used to provide optional information if the condition is fail.

if(condition)

{

statements....

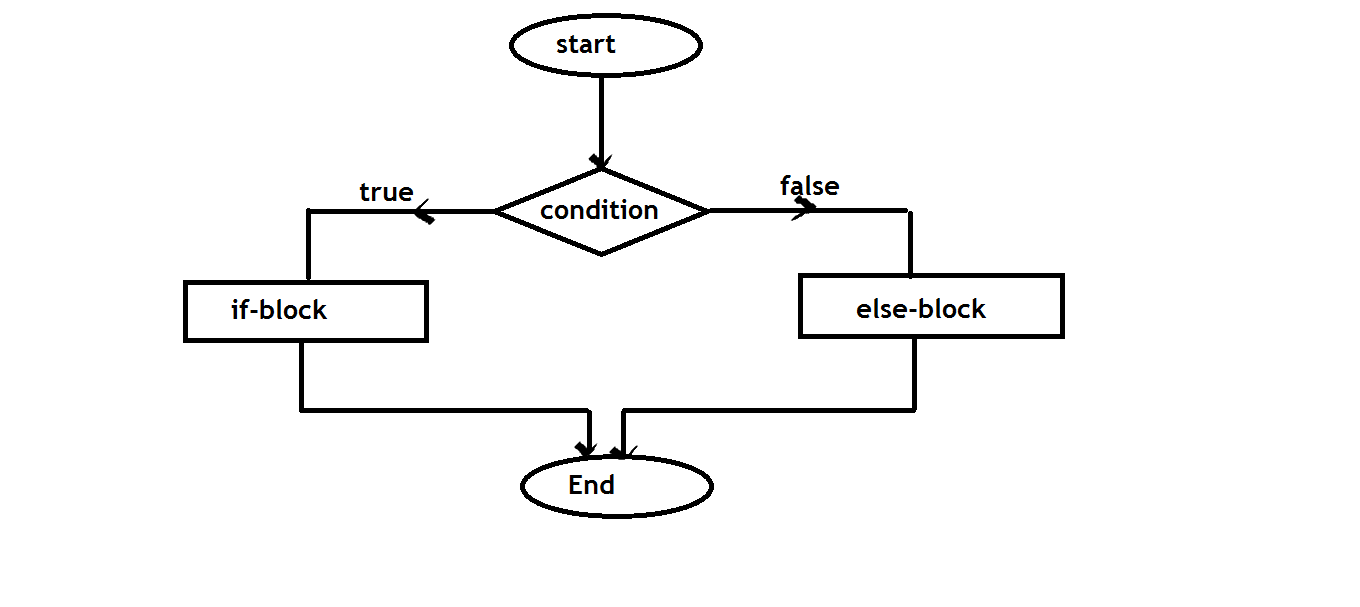
}

else

{

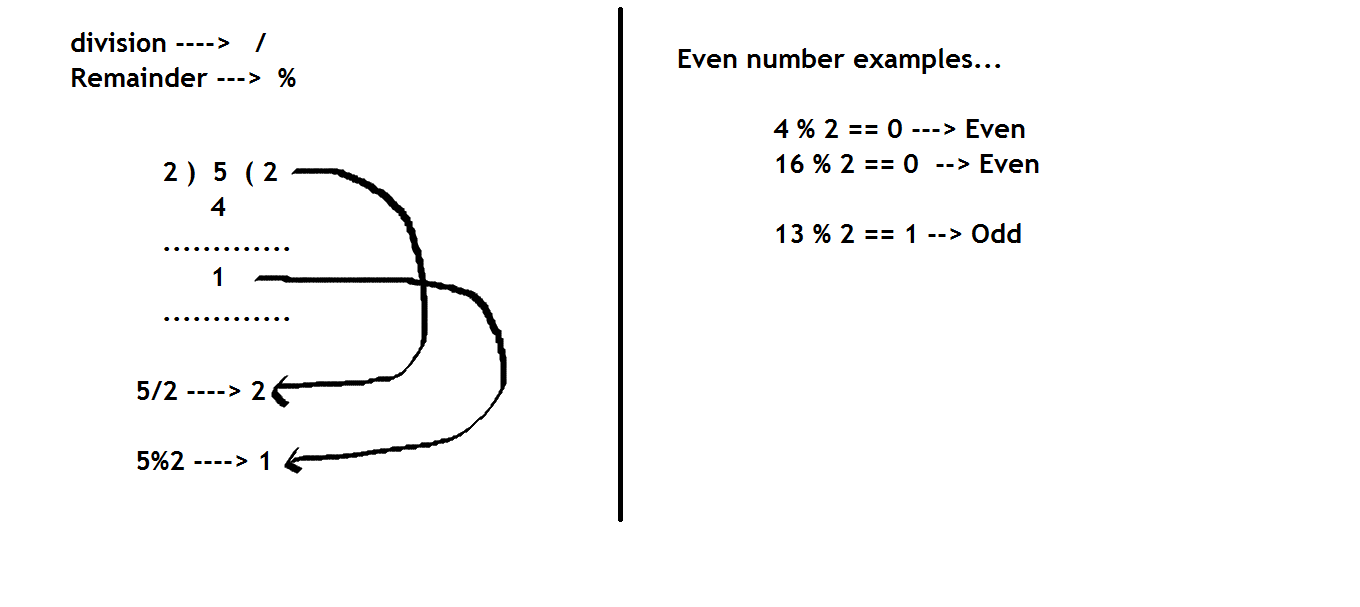
optional msg.....

}



WAP to check the given number is Even or Odd.....

Even : The number which is divisible by 2



class Demo

{

public static void main(String[] args)

{

int number = 12 ;

if(number%2 == 0)

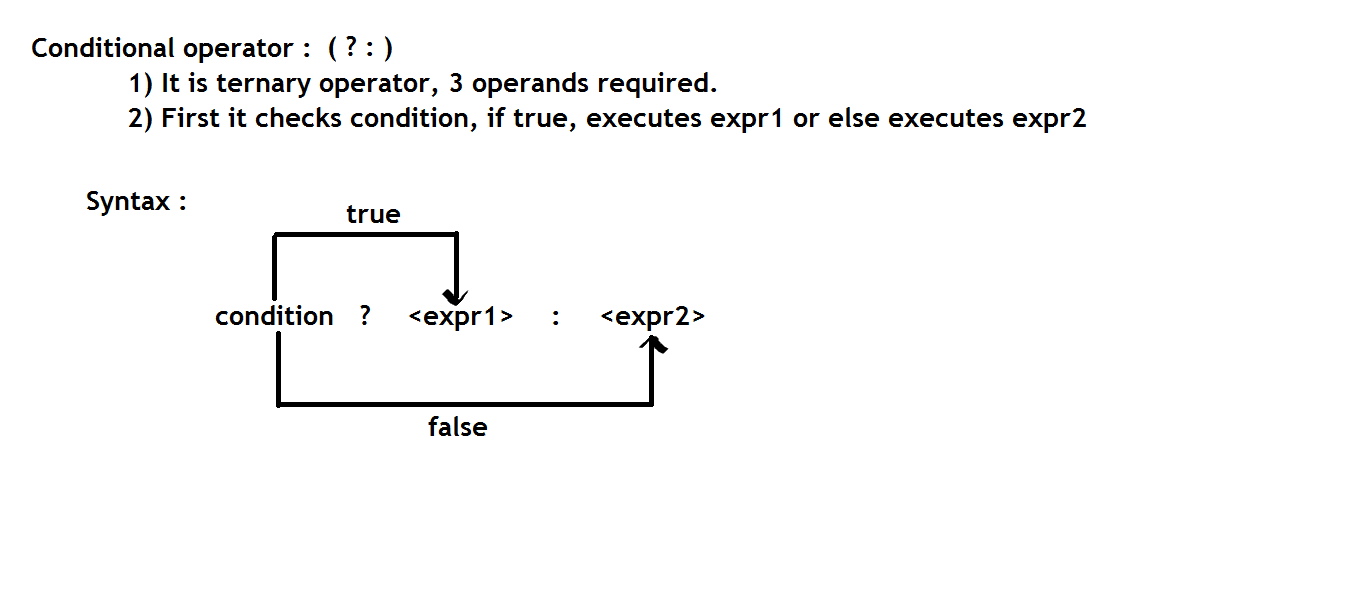
System.out.println(number+" is Even");

else

System.out.println(number+" is Odd");

}

}



class Demo

{

public static void main(String[] args)

{

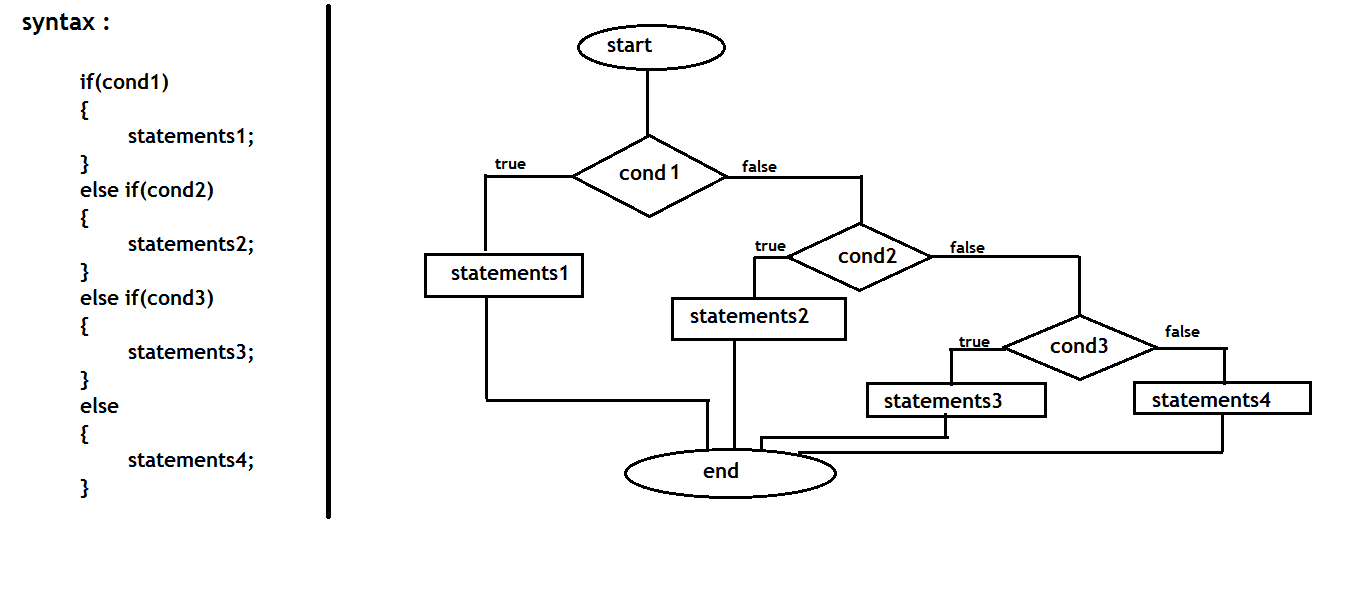
int n = 12 ;

n%2 == 0 ? System.out.println("Even") : System.out.println("Odd") ;

}

}

If-else-if ladder :



Logical operators :

Used to check relation among more than one expression.

Logical-AND (&&)

Logical-OR (||)

Logical-NOT (!)

Truth table :

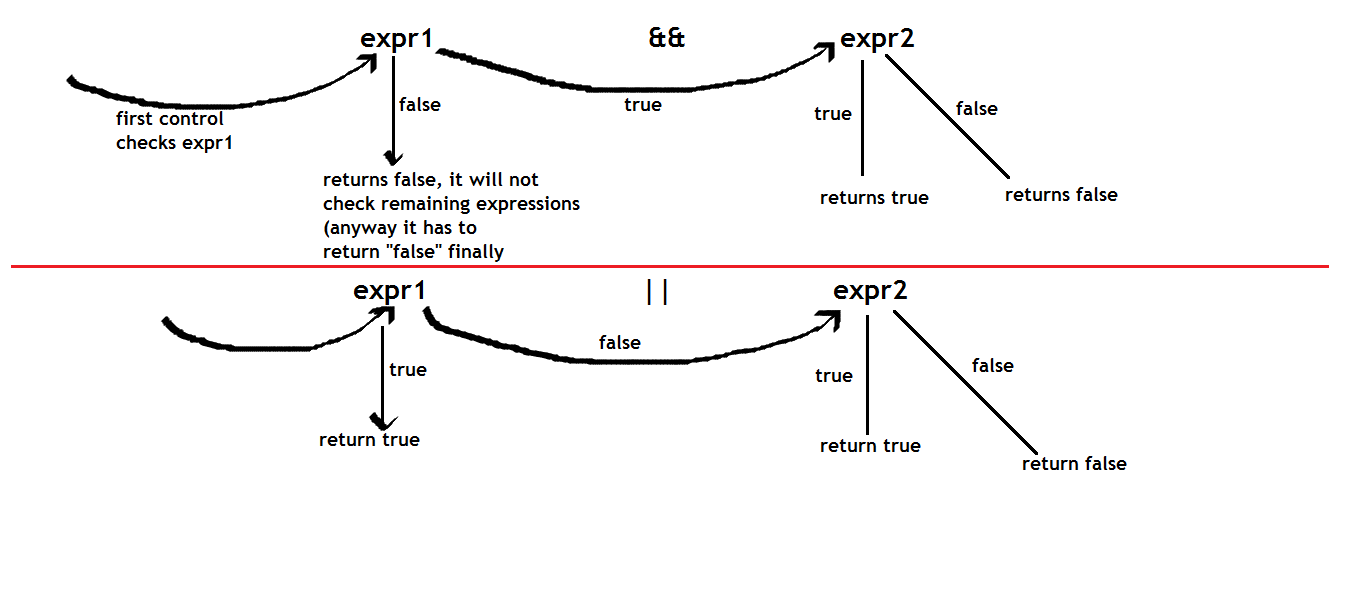
expr1 expr2 && || !expr1

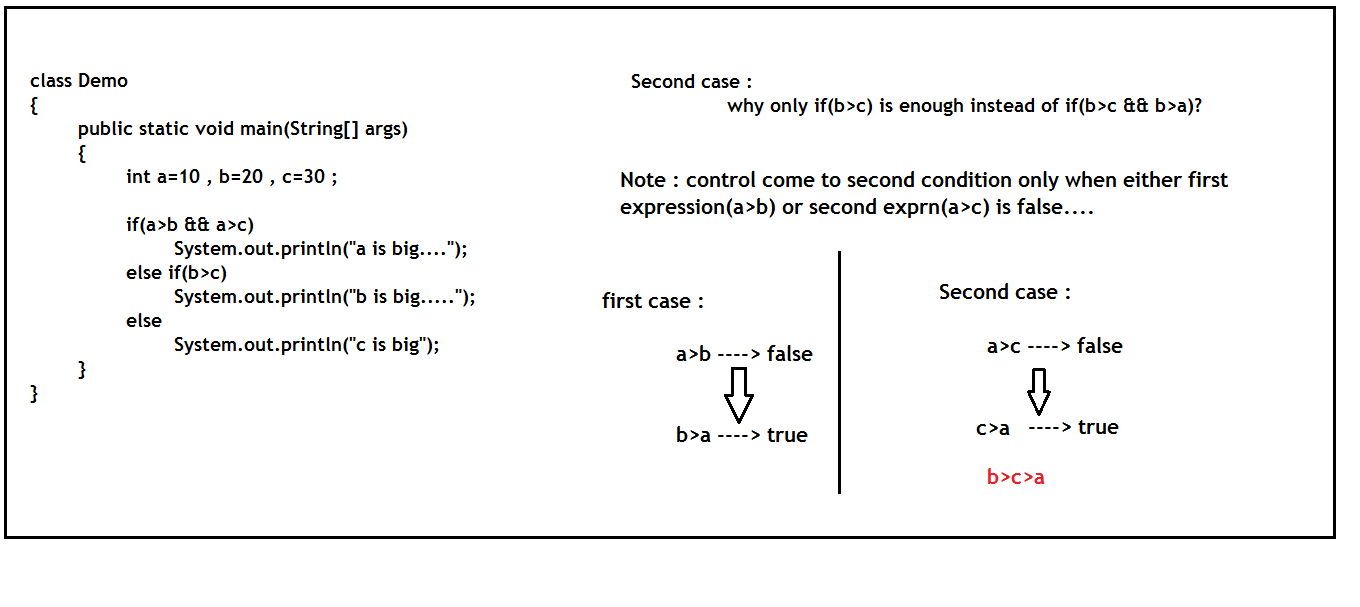
T T T T F

T F F T F

F T F T T

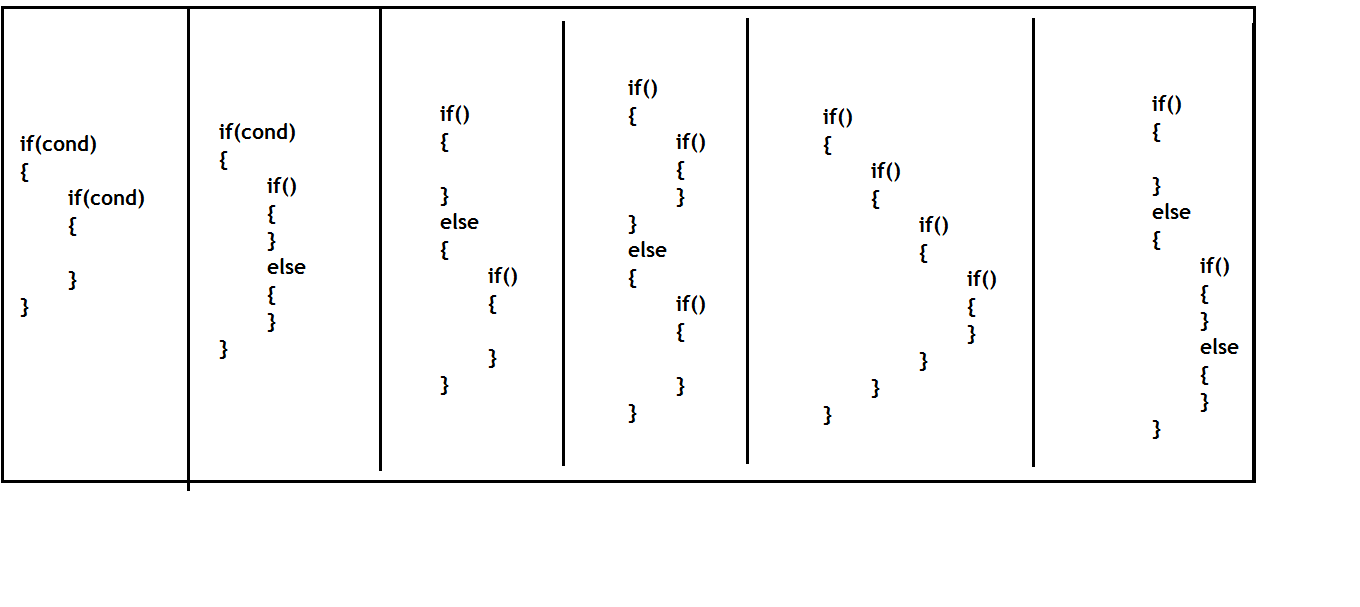
F F F F T

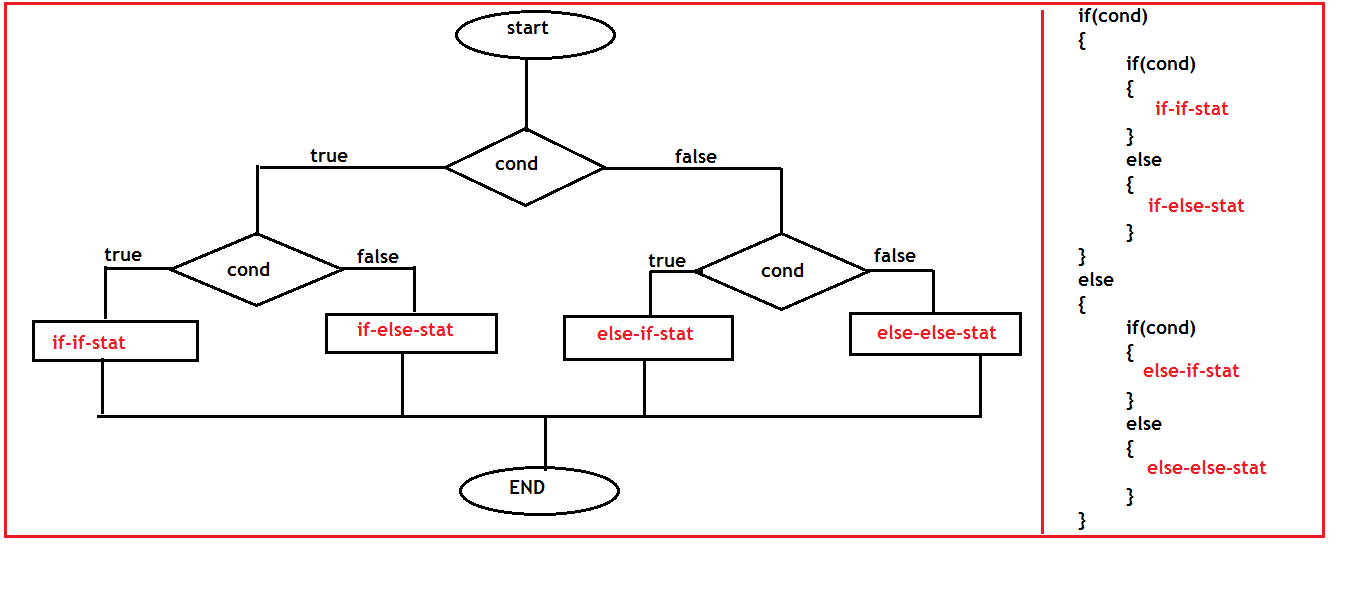




Nested if :

Writing if-statement inside another if-statement....





WAP to find biggest of 3 numbers using nested if-else:

Instead of …..

If(a>b && a>c)

We must write as follows

If(a>b)

{

If(a>c)

{

}

}

Example :

if(a>b)

{

if(a>c)

{

S.o.p("a is big");

}

else

{

S.o.p("c is big");

}

}

else

{

if(b>c)

{

S.o.p("b is big");

}

else

{

S.o.p("c is big");

}

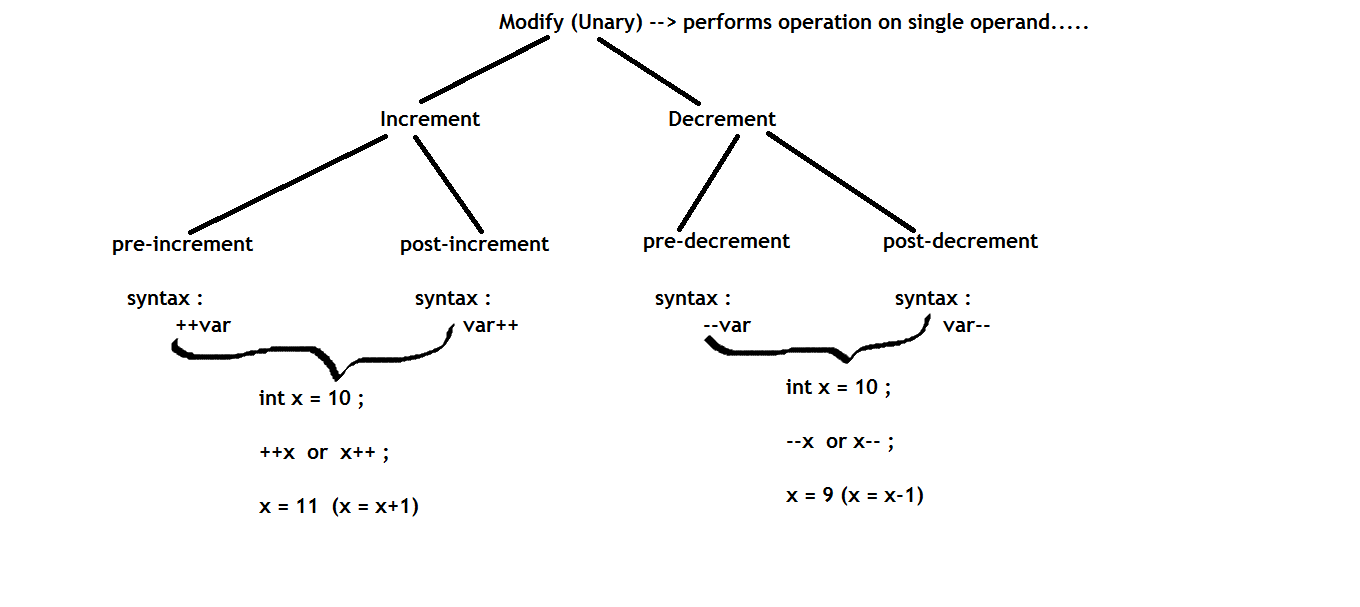
}

Loop control statements:

Modifying operators:

Java supports 2 types of modify operators

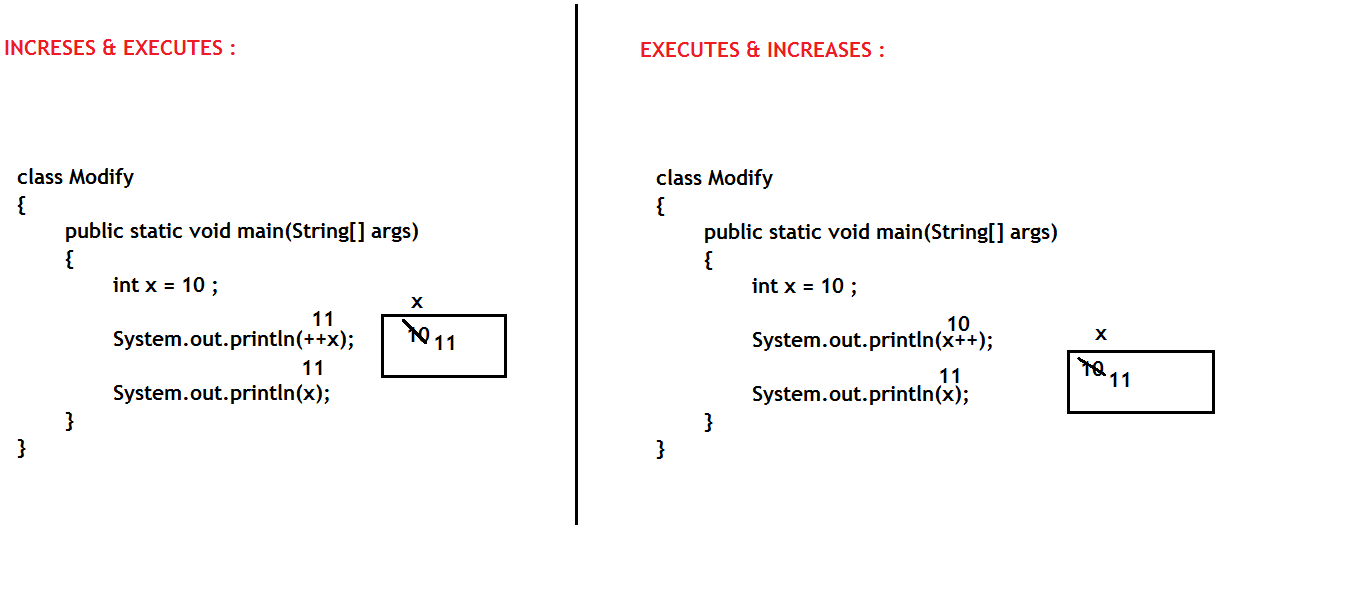
These operators used to increase or decrease the value of a variable by 1



Difference between pre-increment & post increment :

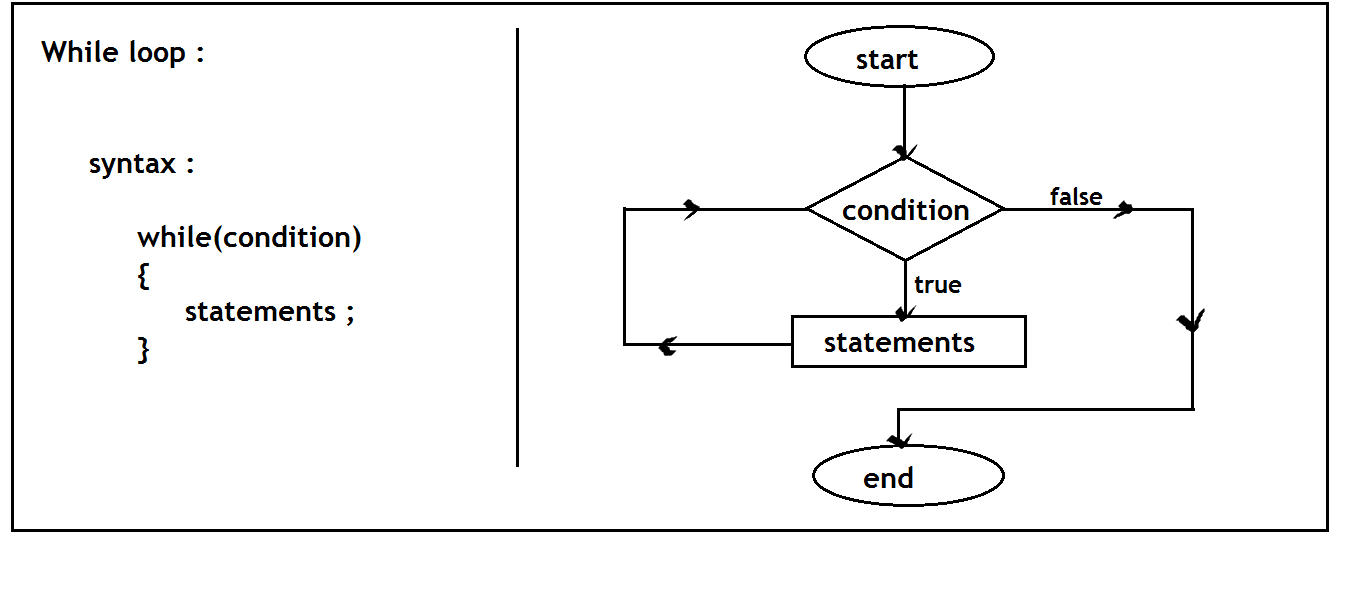
Pre-increment : Increases and then executes

Post-increment : executes and then increases



While loop :

Loops are used to execute a statement/block of statements repeatedly as long as the condition is true.



class Modify

{

public static void main(String[] args)

{

while(true)

{

System.out.println("loop......");

}

}

}

class Modify

{

public static void main(String[] args)

{

while(false)

{ //unreachable statement.....

System.out.println("inside loop......");

}

System.out.println("outside to loop.....");

}

}

class Modify

{

public static void main(String[] args)

{

boolean b = false ;

while(b) //compiler only checks whether "b" variable is present or not, if never checks value....

{

System.out.println("inside loop......");

}

System.out.println("outside to loop.....");

}

}

class Modify

{

public static void main(String[] args)

{

int x = 0 ;

while(x<=10)

{

System.out.println("x value : "+x);

}

}

}

class Modify

{

public static void main(String[] args)

{

int x = 0 ;

while(x<=10)

{

System.out.println("x value : "+x);

++x ;

}

}

}

class Modify

{

public static void main(String[] args)

{

int x = 0 ;

while(++x<=10)

{

System.out.println("x value : "+x);

}

}

}

class Modify

{

public static void main(String[] args)

{

int x = 0 ;

while(x<=10)

{

System.out.println("x value : " + ++x);

}

}

}

/\*

Even : The number which is divisible by 2

\*/

class Loops

{

public static void main(String[] args)

{

int n=12 ;

if(n%2 == 0)

{

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

}

}

}

/\*

Print all the even numbers with in the given range....

\*/

class Loops

{

public static void main(String[] args)

{

int i=1 , n=20 ;

while(i<=n)

{

if(i%2==0)

{

System.out.println("i value : "+i);

}

++i ;

}

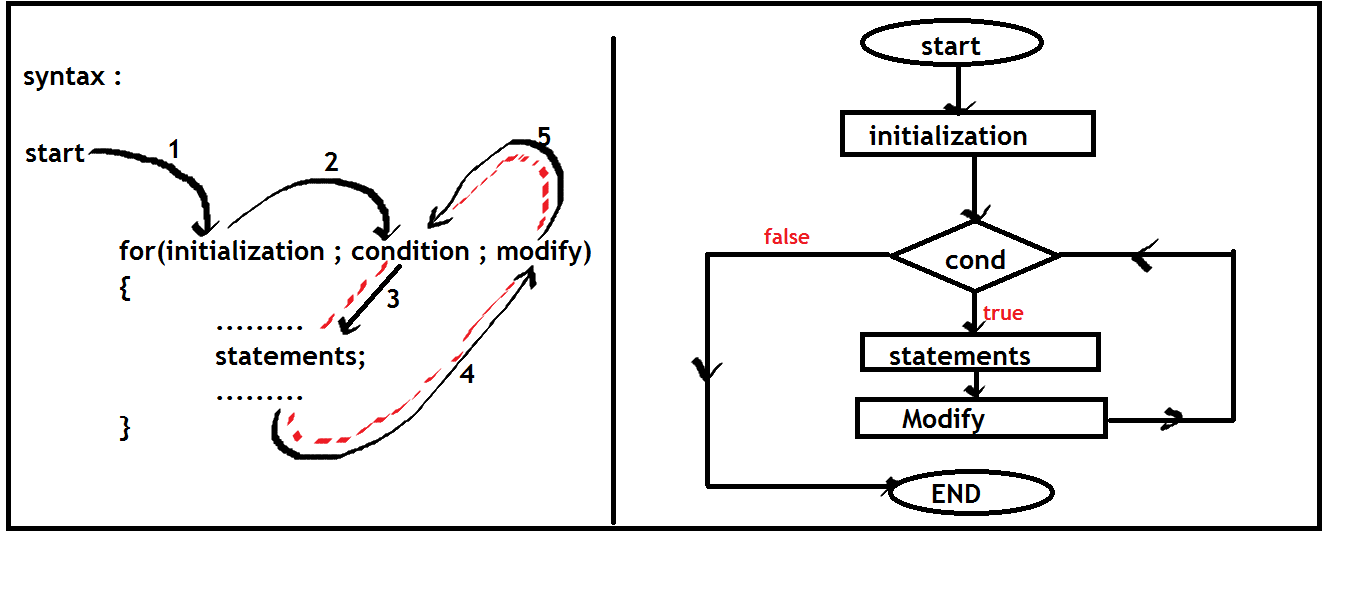
}

}

for loop :

for loop is used to execute a block by placing all the 3 statements(initialization, condition, modify).

Much working like while-loop, but reduces complexity in while loop.



/\*

Sum of first n numbers ......

\*/

class Loops

{

public static void main(String[] args)

{

int n=10 , sum=0 ;

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

{

sum = sum + i ;

}

System.out.println("Sum of first "+n+" numbers is : "+sum);

}

}

/\*

Sum of first n numbers ......

\*/

class Loops

{

public static void main(String[] args)

{

int n=10 , sum=0 ;

System.out.println("i \t sum \t + \t i \t = \t final-sum ");

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

{

System.out.print(i+"\t");

System.out.print(sum+"\t + \t ");

System.out.print(i+"\t = \t ");

sum = sum+i ;

System.out.println(sum);

}

}

}

class Loops

{

public static void main(String[] args)

{

int n=10 , sum=0 ;

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

{

System.out.print(sum + " + "+ i + " = ");

sum = sum + i ;

System.out.println(sum);

}

}

}

class Loops

{

public static void main(String[] args)

{

int n=5 , fact = 1 ;

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

{

fact = fact \* i ;

}

System.out.println("Factorial of "+n+" is : "+fact);

}

}

class Loops

{

public static void main(String[] args)

{

int n=7 ;

for(int i=1 ; i<=10 ; i++)

{

System.out.println(n+" \* "+i+" = "+n\*i);

}

}

}

/\*

7 \* 1 = 7

7 \* 2 = 14

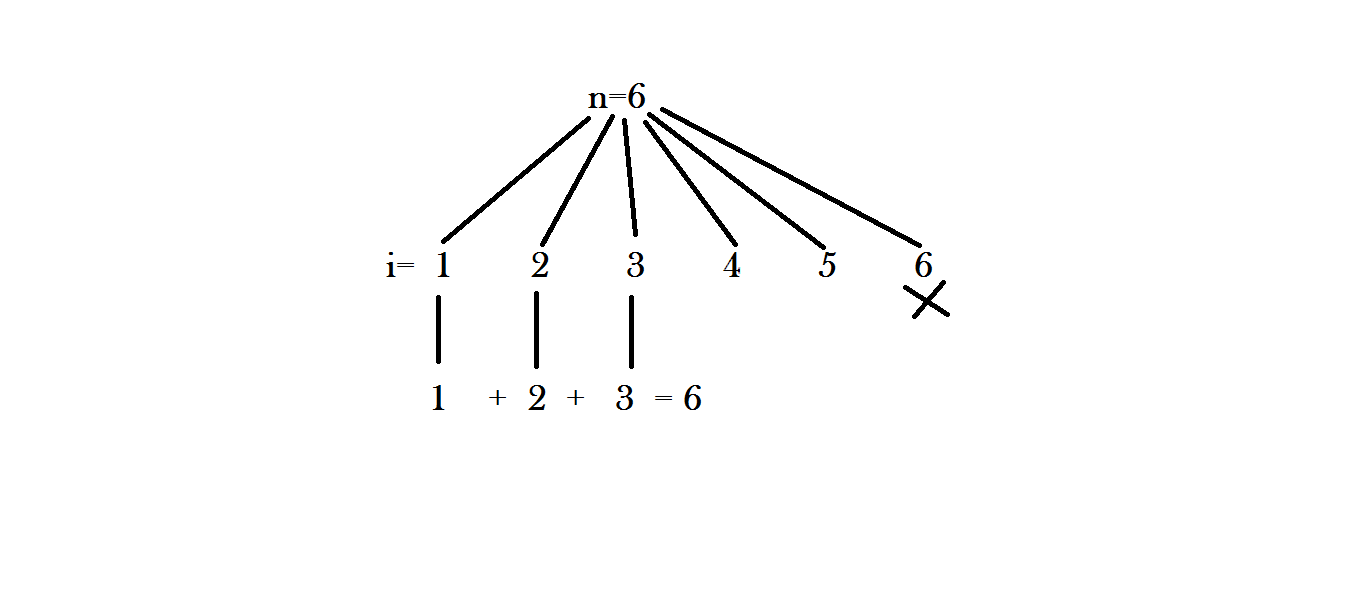
7 \* 3 = 21

\*/

1. WAP to check input number is perfect number or not
2. WAP to check input number is Prime number or not

Perfect number :

Sum of factors of given number except itself is equals to the same number.



class PerfectNumber

{

public static void main(String[] args)

{

int n=6 , sum=0 ;

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

{

if(n%i == 0)

{

sum = sum+i ;

}

}

if(n == sum)

{

System.out.println(n+" is the perfect number.....");

}

else

{

System.out.println(n+" is not the perfect number.....");

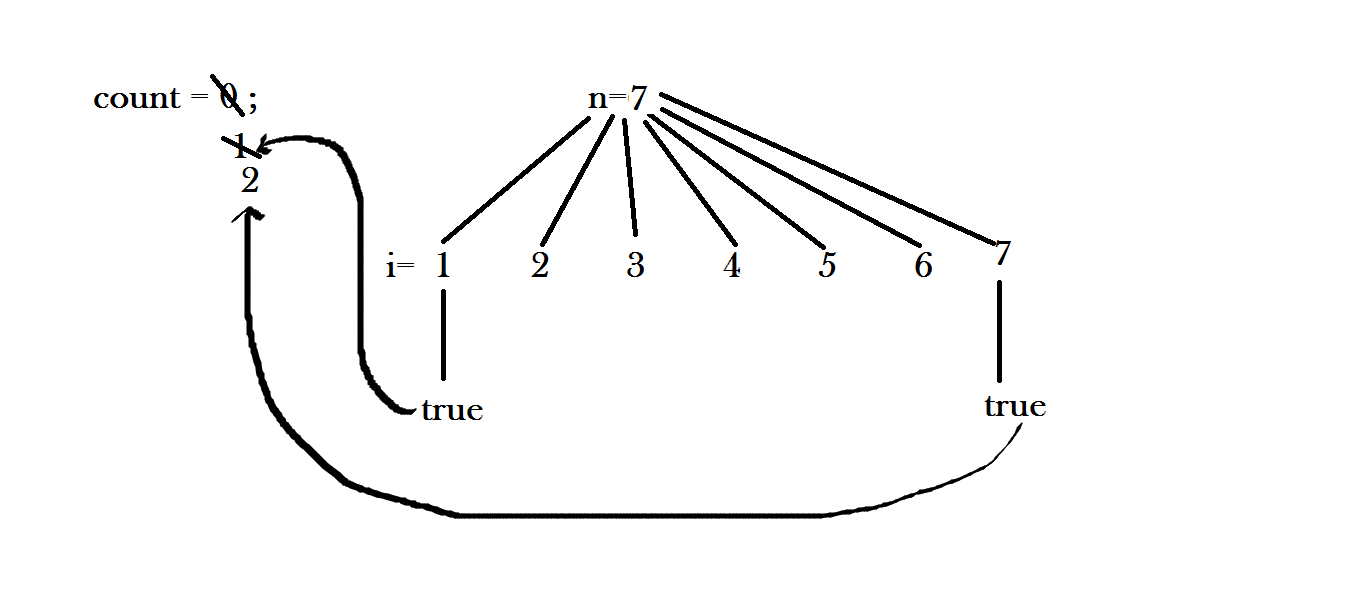
}

}

}

Prime number :

The number which is having 2 factors…….



class PrimeNumber

{

public static void main(String[] args)

{

int n=6 , count=0 ;

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

{

if(n%i == 0)

{

count++;

}

}

if(count==2)

{

System.out.println(n+" is the prime number.....");

}

else

{

System.out.println(n+" is not the prime number.....");

}

}

}

break :

Used to break a loop or switch statement.

class Demo

{

public static void main(String[] args)

{

if(true)

{

break; //CE : need to place either inside the loop or switch

}

}

}

class Demo

{

public static void main(String[] args)

{

while(true)

{

System.out.println("loop......");

break ;

}

}

}

class Demo

{

public static void main(String[] args)

{

while(true)

{

break ;

System.out.println("loop......");//CE : unreachable statement

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=10 ; i++)

{

System.out.println("i value : "+i);

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=10 ; i++)

{

if(i==5)

{

break ;

}

System.out.println("i value : "+i);

}

}

}

Nested loops :

Writing loops inside another loop

while(outer-loop)

{

//outer-statements....

while(inner-loop)

{

//inner statements....

}

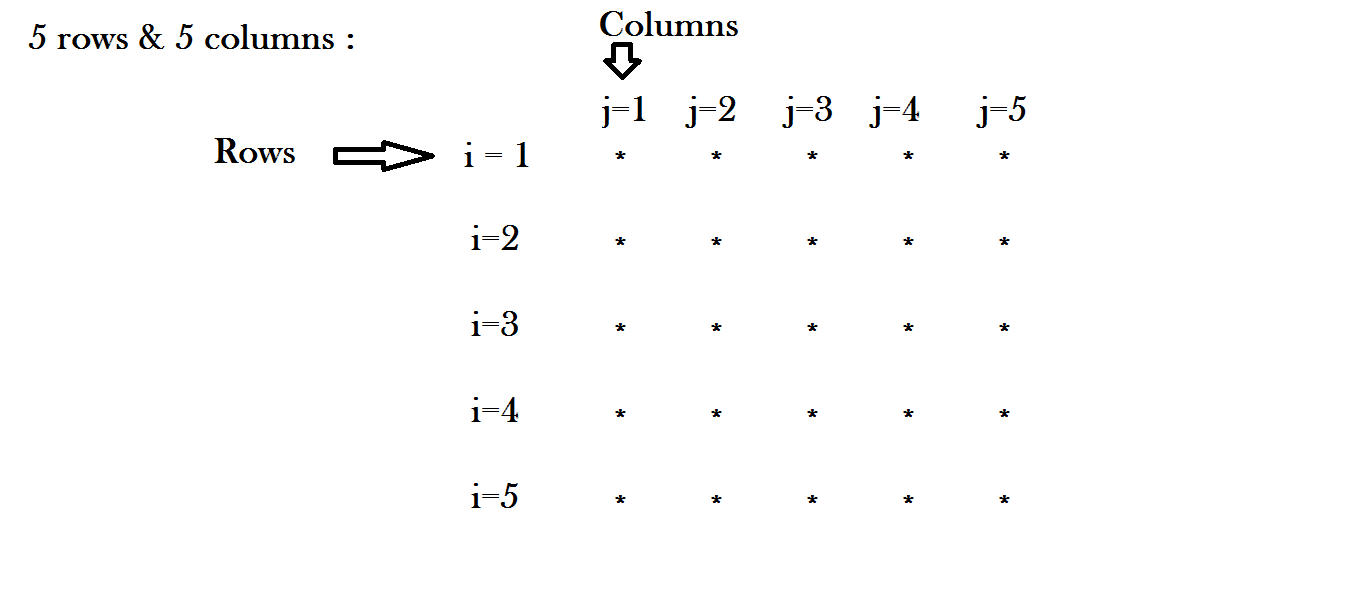
//outer-statements....

}

Nested loops is used to print the information in 2 dimensional format(rows & columns)

Outer loop always represents number of rows

Inner loop always represents number of columns



class Demo

{

public static void main(String[] args)

{

int i, j ;

i=1 ;

while(i<=5)

{

j=1;

while(j<=5) //repeats 5 times(and print 5 stars)

{

System.out.print("\*");

j++;

}

System.out.println(); //sends the control to next line.....

i++;

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=1 ; j<=5 ; j++)

{

System.out.print("\*");

}

System.out.println();

}

}

}

class PrimeNum

{

public static void main(String[] args)

{

int j = 1;

while (j < 5)

{

int i = 1;

do

{

System.out.print(i + ", ");

i++;

if (i > 10)

break;

} while (true);

j++;

System.out.println(" ");

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=4 ; i++)

{

for(int j=1 ; j<=10 ; j++)

{

System.out.print(j+", ");

}

System.out.println();

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=4 ; i++)

{

for(int j=1 ; j<=10 ; j++)

{

System.out.print(i+", ");

}

System.out.println();

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=1 ; j<=5 ; j++)

{

System.out.print("\*");

}

System.out.println();

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=1 ; j<=5 ; j++)

{

System.out.print(i);

}

System.out.println();

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=1 ; j<=5 ; j++)

{

System.out.print(j);

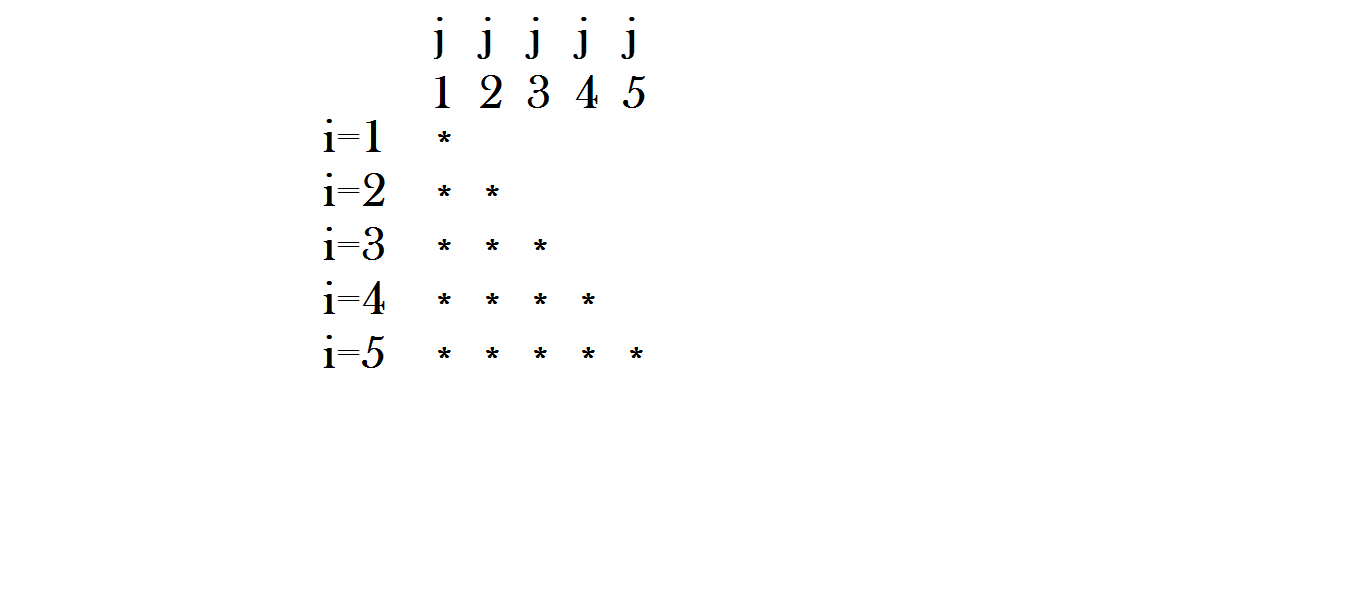
}

System.out.println();

}

}

}



class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

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

{

//System.out.print("\*");

//System.out.print(i);

System.out.print(j);

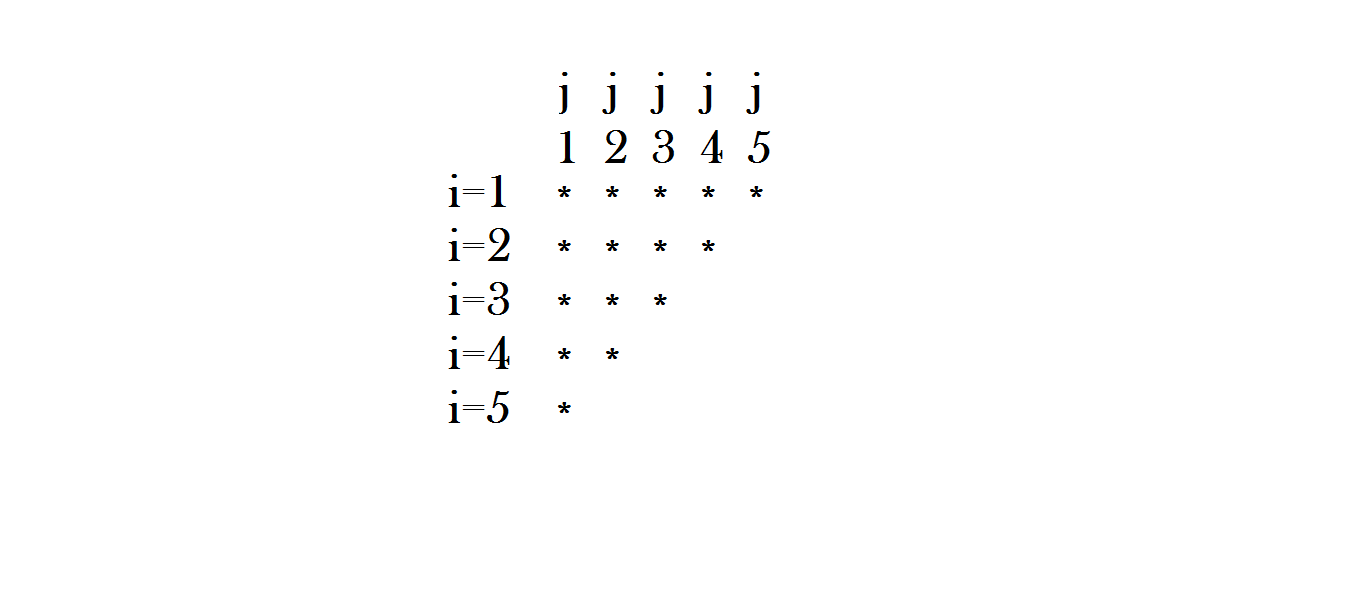
}

System.out.println();

}

}

}



class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=i ; j<=5 ; j++)

{

System.out.print("\*");

//System.out.print(i);

//System.out.print(j);

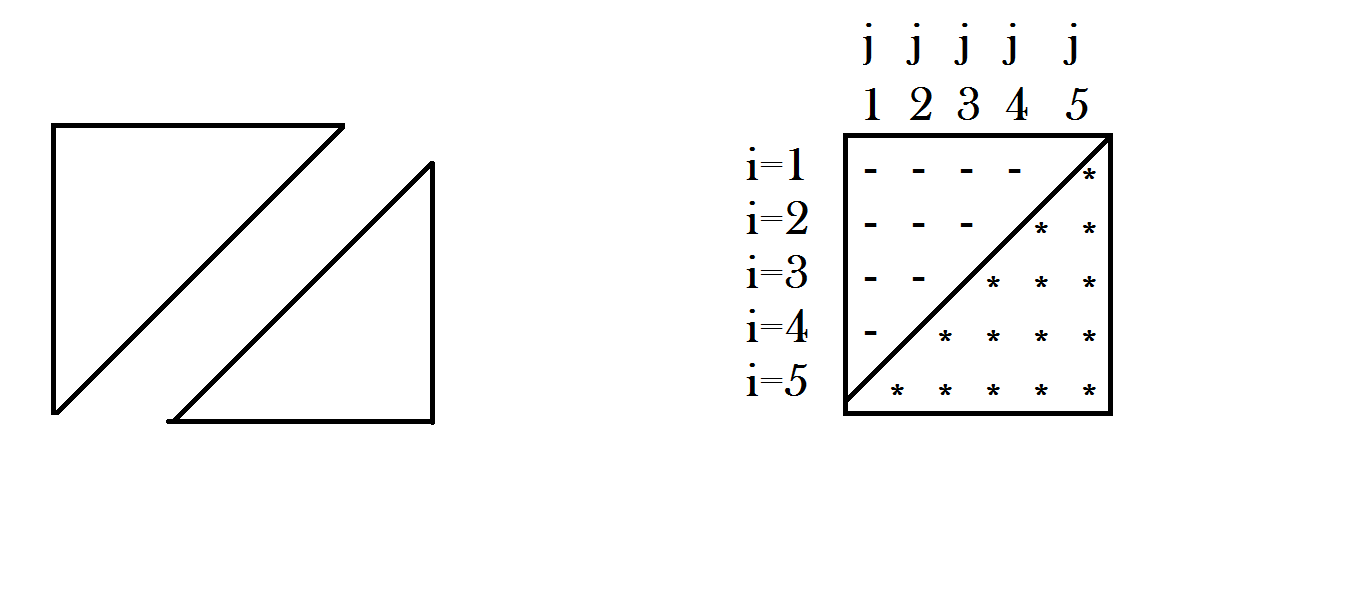
}

System.out.println();

}

}

}



class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=i ; j<5 ; j++)

{

System.out.print(" ");

}

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

{

System.out.print("\*");

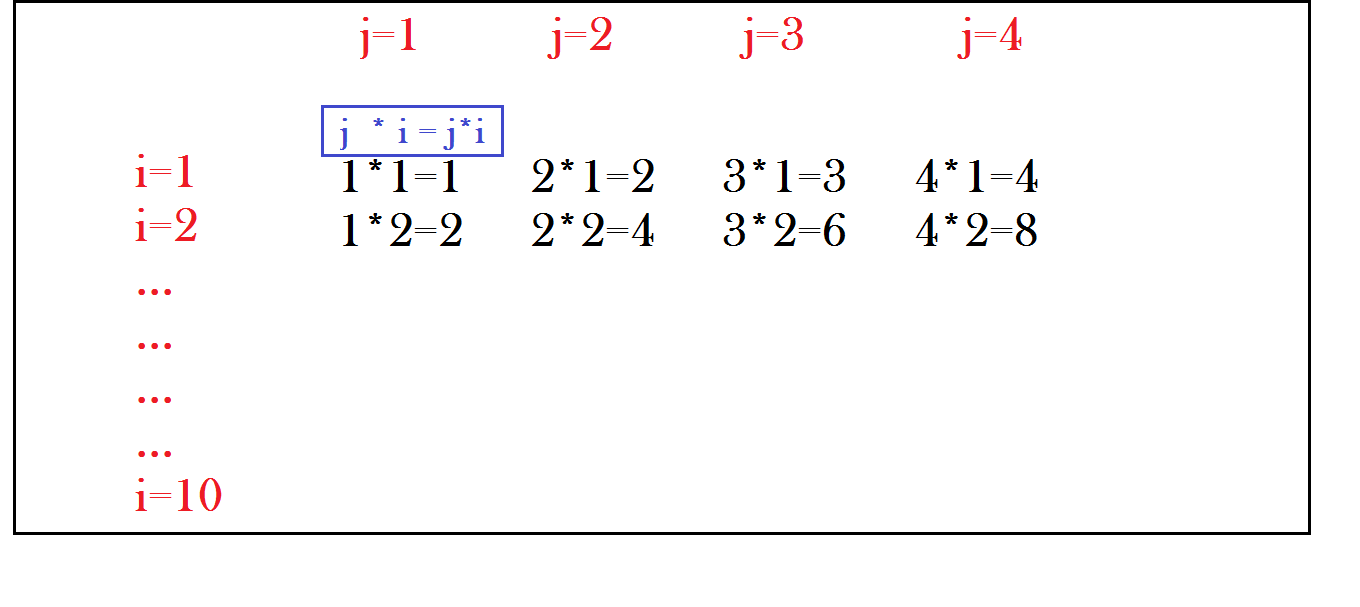
}

System.out.println();

}

}

}



class Demo

{

public static void main(String[] args)

{

int n=5 ;

for(int i=1 ; i<=10 ; i++)

{

for(int j=1 ; j<=n ; j++)

{

System.out.print(j+"\*"+i+"="+j\*i+"\t");

}

System.out.println();

}

}

}

Continue :

1. When “continue” statement executes inside the loop, it will end a particular(current) iteration.
2. It must be placed inside a loop.

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=10 ; i++)

{

System.out.println("i value : "+i);

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=10 ; i++)

{

if(i==5)

{

break ;

}

System.out.println("i value : "+i);

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=10 ; i++)

{

if(i==5)

{

continue ;

}

System.out.println("i value : "+i);

}

}

}

Note : when break statement executes inside inner loop, it terminates execution of inner loop only.

class Demo

{

public static void main(String[] args)

{

while(true)

{

System.out.println("outer loop");

while(true)

{

System.out.println("inner loop");

}

}

}

}

one time "outer loop"

infinite times "inner loop"

class Demo

{

public static void main(String[] args)

{

while(true)

{

System.out.print("outer loop\t");

while(true)

{

System.out.println("inner loop");

break ;

}

}

}

}

output :

infinite times, it prints "outer loop & inner loop"

class Demo

{

public static void main(String[] args)

{

while(true)

{

System.out.print("outer loop\t");

while(true)

{

System.out.println("inner loop");

break ;

}

break ;

}

}

}

class Demo

{

public static void main(String[] args)

{

while(true)

{

System.out.print("start....\t");

continue;

System.out.println("End....."); //CE : unreachable statement

}

}

}

class Demo

{

public static void main(String[] args)

{

char ch='r';

switch(ch)

{

case 'r' : System.out.println("Red color.....");

case 'g' : System.out.println("Green color.....");

case 'b' : System.out.println("Blue color.....");

default : System.out.println("No color.....");

}

}

}

Note : After executing a particular “case-statements”, we need to use “break” to come out of the switch.

class Demo

{

public static void main(String[] args)

{

char ch='g';

switch(ch)

{

case 'r' : System.out.println("Red color.....");

break;

case 'g' : System.out.println("Green color.....");

break;

case 'b' : System.out.println("Blue color.....");

break;

default : System.out.println("No color.....");

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=1 ; j<=5 ; j++)

{

System.out.println("i : "+i+"\t j : "+j);

}

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=1 ; j<=5 ; j++)

{

if(i==j)

{

System.out.println("i : "+i+"\t j : "+j);

}

}

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=1 ; j<=5 ; j++)

{

if(i==j)

{

break ;

}

System.out.println("i : "+i+"\t j : "+j);

}

}

}

}

class Demo

{

public static void main(String[] args)

{

for(int i=1 ; i<=5 ; i++)

{

for(int j=1 ; j<=5 ; j++)

{

if(i==j)

{

continue ;

}

System.out.println("i : "+i+"\t j : "+j);

}

}

}

}

Ques :

1. WAP to check input number is prime or not?
2. WAP to check input number is perfect or not?
3. Find sum of first n numbers?
4. Find factorial of given number?
5. WAP to check input number is Armstrong or not?
6. WAP to check input number is strong number or not?
7. WAP to check input number is palindrome or not?