Recursion in java

Recursion in java is a process in which a method calls itself and the method that calls itself is known as a recursive method. Recursion provides a way to break complicated problems into simple smaller problems which are easier to solve.Using recursion, certain problems can be solved easily like Tower of Hanoi, fibonacci number, tree traversals etc.  
  
For beginner, recursion can be difficult to understand in the first place but once you understand it, a whole new world of programming will open for you.

Syntax of recursive function

return\_type method\_name(){

//statements

methodname();

}

Let's see how we can use recursion to find the value of factorial N.

N! = 1 x 2 x 3 x 4....x (N-2) x (N-1) x N

N! = (N-1)! x N

Let factorial(N) is a function to calculate and return value of N!. As per the above recurrence equation, To find factorial(N) we can first calculate factorial(N-1) then multiply it with N.

factorial(N) = factorial(N-1) x N

Function factorial(N) reduces the problem of finding factorial of a number N into sub-problem of finding factorial on N-1. Here is a recursive function implementing above mentioned recurrence relationship.

public int factorial(int n) {

if (n <= 1) {

return 1;

}

return factorial(n - 1) \* n;

}

In the above method, we have called factorial() method from the body the factorial() method. In order to stop the recursive call, we need to provide some termination condition inside the method. Otherwise, the method will be called infinitely and it will never terminate. Hence, we have added a if statement(n <= 1) to terminate the recursive call.

**Recursion termination condition**

Recursive functions can run into the problem of infinite recursion. When the recursive function never stops calling itself, it is called infinite recursion.  
  
Every recursive function should have a termination condition, which is the condition where the function stops calling itself. In above method, the termination condition is when n <= 0, then we return 1 instead of calling fatorial method again.

Java programs to find factorial of a number using recursion

|  |
| --- |
| public class FactorialRecursion {    static int factorial(int n) {      System.out.println("N = " + n);      // recursion termination condition      if (n &lt;= 1) {        return 1;      }      // Recursive call      return factorial(n - 1) \* n;    }      public static void main(String[] args) {      int factorial5 = factorial(5);      System.out.println("Factorial 5 = " + factorial5);    }  } |

Output

N = 5

N = 4

N = 3

N = 2

N = 1

Factorial 5 = 120

Java programs to print Nth fibonacci number using recursion

Fibonacci sequence is the series numbers in the following integer sequence  
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89 ....  
the first two numbers in the Fibonacci sequence are 0 and 1, and each subsequent term is the sum of the previous two terms. Nth term of Fibonacci series is defined by the recurrence relation:  
  
fibonacci(N) = Nth term in fibonacci series  
fibonacci(N) = fibonacci(N - 1) + fibonacci(N - 2);  
whereas, fibonacci(0) = 0 and fibonacci(1) = 1  
  
For Example :  
fibonacci(4) = fibonacci(3) + fibonacci(2);

|  |
| --- |
| public class FibonacciRecursion {      static int fibonacci(int n) {      // Recursion termination condition      if(n < 2) {        return n;      }      return fibonacci(n - 1) + fibonacci(n - 2);    }      public static void main(String[] args) {      int sixthTerm = fibonacci(6);      System.out.println("Sixth term = " + sixthTerm);    }  } |

Output

Sixth term = 8