import java.util.Scanner;  
  
public class Main {  
 static final double *phi* = (1 + Math.*sqrt*(5)) / 2;  
  
 public static long fib\_funni(int input) {  
 return Math.*round*((Math.*pow*(*phi*, input) - Math.*pow*((1 - *phi*), input))/Math.*sqrt*(5));  
 }  
  
 public static int fib\_iter(int input) {  
 int[] fib = {1, 1};  
 for (int i = 1; i < input; i++) {  
 int temp = fib[1];  
 fib[1] = fib[0];  
 fib[0] += temp;  
 }  
 return fib[1];  
 }  
  
 public static int fib\_rec(int input) {  
 if (input < 2) return input;  
 return (*fib\_rec*(input - 1) + *fib\_rec*(input - 2));  
 }  
  
 public static void main(String[] args) {  
 System.*out*.print("Enter input here: ");  
 Scanner scanner = new Scanner(System.*in*);  
 int input = scanner.nextInt();  
 System.*out*.println("Fib Funni = " + *fib\_funni*(input));  
 System.*out*.println("Fib Iterative = " + *fib\_iter*(input));  
 System.*out*.println("Fib Recursive = " + *fib\_rec*(input));  
 }  
}

I would use the iterative approach as when the number reaches bigger values the function will be called more putting strain on the CPU.

Triangle():

public int triangle(int rows) {

return rows == 0 ? 0 : rows + triangle(rows – 1);

}

NoX:

public String noX(String str) {

if (str.length() == 0) return "";

if (str.charAt(0) == 'x') return noX(str.substring(1));

return str.charAt(0) + noX(str.substring(1));

}

Array11:

public int array11(int[] nums, int index) {

return index == nums.length ? 0 : (nums[index] == 11 ? 1 : 0) + array11(nums, index+1);

}