COS126 Fibonacci Programs

```
/*************************************************************************
 *  Compilation:  javac Fibonacci.java
 *  Execution:    java Fibonacci N
 *
 *  Compute the Nth Fibonacci number using recursion.
 *
 *  WARNING  This program is spectacularly inefficient and is meant
 *  to illustrate a performance bug. (e.g., set N = 45)
 *
 *  Remarks:  The 93rd Fibonacci number would overflow a long, but
 *            this will take so long to compute with this function,
 *            that we don't bother to check for overflow.
 *
 *  %  java Fibonacci 10
 *  55
 *
 *************************************************************************/

public class Fibonacci {

   public static long fib(int n) {
       // base case
       if (n <= 1) return n;
       else return fib(n-1) + fib(n-2);
   }

   public static void main(String[] args) {
       int N = Integer.parseInt(args[0]);

       // print results
       System.out.println( fib(N) );
   }
}
```

```
/******************************************************************
 *  Compilation:  javac Fibonacci.java
 *  Execution:    java Fibonacci N
 *
 *  Compute the Nth Fibonacci number using dynamic programming
 *  via recursion and memo-ization.
 ******************************************************************/

public class Fibonacci {
   public static long fib(int n, long[] f) {
       // base cases: f[0], f[1], f[n] already set
       if (n <= 1) return n;
       if (f[n] !=0 ) return f[n];

       // Still zero? compute fibonacci number
       f[n] = fib(n-1, f) + fib(n-2, f);
       return f[n];
   }

   public static void main(String[] args) {
       int N = Integer.parseInt(args[0]);
       if (N < 1 || N > 92) {
           throw new RuntimeException("N must be between 1 and 92");
       }

       // All elements in array initially at default value zero.
       long[] f = new long[N+1];

       // print results
       System.out.println( fib(N, f) );
   }
}
```

```
/******************************************************************
 *  Compilation:  javac Fibonacci.java
 *  Execution:    java Fibonacci N
 *
 *  Compute the Nth Fibonacci number using bottom-up dynamic programming.
 ******************************************************************/

public class Fibonacci {
   public static void main(String[] args) {
       int N = Integer.parseInt(args[0]);
       if (N < 1 || N > 92) {
           throw new RuntimeException("N must be between 1 and 92");
       }

       long[] fib = new long[N+1];

       // base cases
       fib[0] = 0;
       fib[1] = 1;

       // bottom-up dynamic programming
       for (int n = 2; n <= N; n++)
           fib[n] = fib[n-1] + fib[n-2];

       // print results
       System.out.println( fib[N] );
   }
}
```