Recursive functions

Objectives: Usage of recursion, observation of the pro's and con's, analysis of the program flow, use of simple data structures to store intermediate results to speed-up computations.

1. Write a recursive function to calculate a binomial coefficient $\binom{n}{k}$ using the formula

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

   - Ask the user for the values of n and k.
   - Use a main loop to compute and visualize the results until the user wants to stop (for instance, giving a negative number for n).
   - Make experiments to figure out what are the largest input values such that python is able to compute the coefficients using this recursive method in a reasonable amount of time (What is the largest k respective to n that generate the largest coefficient?).

2. Write a recursive function to compute the nth value of the Fibonacci series (remember, the series starts with: 1,1,2,3,5,8,13,21,34,...). Not considering the first two values $f_0 = 1$ and $f_1 = 1$ Quitando los dos casos iniciales $f_0 = 1$ y $f_1 = 1$, the recursion formula is $f_n = f_{n-1} + f_{n-2}$.

   - Ask the user for a value of n.
   - Use a main loop to compute and visualize the result until the user wants to stop (for instance, giving a negative number for n).
   - Make experiments to figure out what is the largest input value such that python is able to compute the series using this recursive method in a reasonable amount of time.

3. Analize the function calls your programs are performing, i.e., try to find an answer to the question: How many times a recursive function is called?

   Improve your programs considerably (i.e., the run times) with the help of simple data structures that store intermediate results such that simple look-ups are sufficient, rather than recalculation of a value (for the Fibonacci series we saw it in class).