Tutorial Week04

Question 1:

Prove that $F_n = f(n) = f(n-1) + f(n-2) \geq (\frac{3}{2})^n$ for "large" n.

Answer:

Note well, the question asks you to guarantee (prove) that $f(n)$ or F_n grows faster than $(\frac{3}{2})^n$ for "large" n. This does not mean that it holds for *all* n. For example, when $n = 2$, $F_n = 2$ yet $(\frac{3}{2})^2 = \frac{9}{4} > 2$. So $n = 2$ cannot be a base case for the theorem. I will leave it to you to find the base case where the relationship holds.

I hope that the plot below demonstrates that the theorem becomes true eventually (as n gets larger) but proving it is a different matter.

The theorem is proved by induction.

Inductive case: Assume that for some value K, the Kth Fibonacci number and the one before that (K-1) satisfy the relationship. That is

$$F_K \geq (\frac{3}{2})^K \quad \text{and} \quad F_{K-1} \geq (\frac{3}{2})^{K-1}.$$

What about F_{K+1}? Every Fib. no. satisfies $F_i = F_{i-1} + F_{i-2}$, so

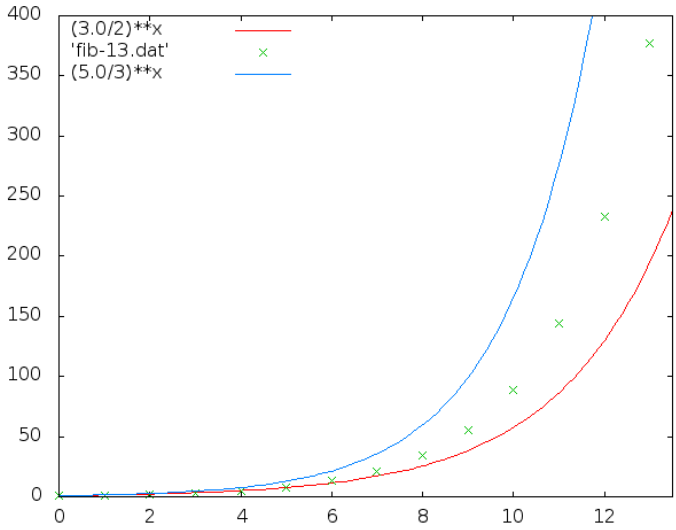
$$F_{K+1} = F_K + F_{K-1}$$

But the two terms on the RHS fall in the range of our inductive assumption. So

$$\begin{aligned}
F_{K+1} &= F_K + F_{K-1} \\
&\geq \left(\frac{3}{2}\right)^K + \left(\frac{3}{2}\right)^{K-1} \\
&= \frac{\left(\frac{3}{2}\right)^{K+1}}{\frac{3}{2}} + \frac{\left(\frac{3}{2}\right)^{K+1}}{\left(\frac{3}{2}\right)^2} \\
&= \frac{2}{3}\left(\frac{3}{2}\right)^{K+1} + \frac{2^2}{3^2}\left(\frac{3}{2}\right)^{K+1} \\
&= \left[\frac{2}{3} + \frac{2^2}{3^2}\right]\left(\frac{3}{2}\right)^{K+1} \\
&= \frac{10}{9}\left(\frac{3}{2}\right)^{K+1} \\
&> \left(\frac{3}{2}\right)^{K+1}
\end{aligned}$$

So we have argued the truth of the inductive step. You will need to find an appropriate starting point (comprising two consecutive values of n for which the claim holds true) to base the induction on.

A puzzle: even though $F_0 = 1 = \left(\frac{3}{2}\right)^0$ why can we not use $n = 0$ as a base case?



This graphic was computed with `gnuplot` with the following commands:

```
unset logscale

set key left top Left
set output 'fib-13.png'
set term png

plot [x=0:13.5] [0:400] (3.0/2)**x w lines, 'fib.dat', (5.0/3)**x w lines;
```

The file `fib.dat` contained the first 13 Fibonacci numbers in the form `i: Fi`, one per line.

Even more of a puzzle: We have proved that $F_n \geq \left(\frac{3}{2}\right)^n = (1.5)^n$ when n is "largish". If you look carefully at the role of $\frac{3}{2}$ in the inductive step above you will see that it could be replaced by $\frac{8}{5}$ and by following this through the proof would still work, meaning that $F_n \geq \left(\frac{8}{5}\right)^n = (1.6)^n$ when n is "largish". (Instead of deriving $F_{K+1} \geq \frac{10}{9}\left(\frac{3}{2}\right)^{K+1}$, we would prove $F_{K+1} \geq \frac{65}{64}\left(\frac{8}{5}\right)^{K+1}$.) This is a better answer than the previous one because it's a larger lower bound – reflect carefully on that – but the catch is that the base case associated with this one is a lot bigger. You have to go out to $n = 30$ to find the base case now (the two consecutive values of n):
$F_{29} = 832040 \not\geq (1.6)^{29} = 830767.497365$, but
$F_{30} = 1346269 \geq (1.6)^{30} = 1329227.995784$.
$F_{31} = 2178309 \geq (1.6)^{31} = 2126764.79325586$.
So the price you pay for getting a tighter approximation to F_n is that it doesn't become valid until later values of n are reached.

Question 2 (2.15 of *Weiss, ed. 3*)

The key property to spot about the contents of the array is that if ever an array element has contents less than its index – that is, $A_i < i$ – then the same must hold for every array element smaller than i. (The properties that all the array's contents are integers and the integers strictly increase as we traverse the array are crucial for this to hold.) This means that if we probe the array at position i and find $A_i < i$ then we can safely ignore every array index j where $j \leq i$.

By the same argument if we probe the array at index i and find $i < A_i$ then no index j where $i \leq j$ will satisfy $A_j = j$.

Thus every time (and anywhere) we probe the array we can say one of the three following holds:

- if $A_i = i$ **gotcha**
- if $A_i < i$, concentrate search on portion to the right of i
- if $A_i > i$, concentrate search on portion to the left of i

This is the classic formulation of *binary search*.