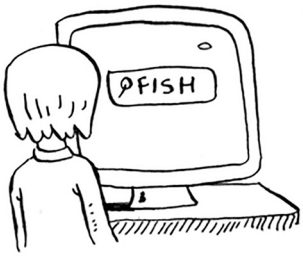
It can be hard to come up with a dynamic-programming solution. That's what we'll focus on in this section. Some general tips follow:

- Every dynamic-programming solution involves a grid.
- The values in the cells are usually what you're trying to optimize. For the knapsack problem, the values were the value of the goods.
- Each cell is a subproblem, so think about how you can divide your problem into subproblems. That will help you figure out what the axes are.

Let's look at another example. Suppose you run dictionary.com. Someone types in a word, and you give them the definition.



But if someone misspells a word, you want to be able to guess what word they meant. Alex is searching for *fish*, but he accidentally put in *hish*. That's not a word in your dictionary, but you have a list of words that are similar.

SIMILAR TO "HISH":

- FISH
- VISTA

(This is a toy example, so you'll limit your list to two words. In reality, this list would probably be thousands of words.)

Alex typed *hish*. Which word did Alex mean to type: *fish* or *vista*?

Making the grid

What does the grid for this problem look like? You need to answer these questions:

- What are the values of the cells?
- How do you divide this problem into subproblems?
- What are the axes of the grid?

In dynamic programming, you're trying to *maximize* something. In this case, you're trying to find the longest substring that two words have in common. What substring do *hish* and *fish* have in common? How about *hish* and *vista*? That's what you want to calculate.