Assignment 19 - Recursive List Processing

The Process of Recursion

- Find a smaller/simpler version of the problem within itself.
- Call the function within itself with new arguments to solve the simpler problem.
- Work out the base cases, where the answer requires no working out.

Comments

- Comments should:
    - Be self-contained — The comment should not refer to other programs, e.g. should not say "This program is a helper function for [other program]."
    - Mention each parameter explicitly — The comment should explain what each input is, and call them by name.

The First Few Problems

The first few problems in this assignment are problems we've seen before, but we're now trying to solve them recursively rather than as we did before.

Note that it's possible to write, very nicely, IsSorted() and IsPalindrome() without using any IF statements.

Max()

There are two ways to write this, and they're very similar. One way will give the result for Max(range(100)) almost instantly, but the other way (though it looks fine at first) will take 335,000,000 times the age of the universe.

You may end up accidentally writing this program. If your program is not back within a second for range(100), then rewrite your program rather than waiting for it to finish.

The Slow Way of Writing Max()

It won't look too bad when you initially see it. There are two problems. One is more obvious, one is very subtle.

```
def Max(s):
    if len(s) == 0:
        return None
    elif len(s) == 1:
        return s[0]
    elif s[0] > Max(s[1:]):
        return s[0]
    else:
        return Max(s[1:])
```

The first problem

In this code, you find Max(s[1:]) twice, which is a massive redundancy.

Each of those calls will then find Max(s[2:]) twice, and each of those will find Max(s[3:]) twice, for a total of 16 calls of Max(s[3:]). For large lists, this is a huge increase in work.

Every time you add 1 element to the sequence, you double the amount of time the program takes to run.

Lambda

Testing the last few problems is easier if we use lambda. Lambda is a reserved word.

Normally when defining functions, we are doing two things:

- Making a process that will take in values, perform a calculation, and give a result.
- Giving the process a name.

Using it:

```
lambda feet,inches : 12 * feet + inches
```

This creates the process but does not give it a name. Here's how you call it:

```
(lambda feet,inches : 12 * feet + inches)(5, 9)
```

Side Note on Lambda's Importance

There's a book called Structure and Interpretation of Computer Programs which is an excellent book, and lambda (the letter) is on the the cover.

You can find this book at the following address (for free):

```
mitpress.mit.edu/sicp
```

Parameters

We can use extra parameters in our functions if we like, as long as the end-user is not required to use them in order to call the function. I.e. we may add default parameters to our functions if we would like.

Line Length

Manning's a19.py has 40 lines of content, not including comments or blank lines.