COLBY COLLEGE
Computer Science Department

CS231 **Allen B. Downey**
Fall 1999

Assignment 4: Recursion

Due: Friday 8 October

Reading: Standish, Chapter 4 (skim 4.4)

The purpose of this assignment is to review some of the recursive programs we wrote last semester and to write some new ones.

A little number theory

1. Do Exercises 1, 2 and 4 on page 113 of the text.

Two silly recursive methods

1. Do exercises 8 and 9 on page 115 of the book.
2. Explain in a sentence why I consider these silly recursive methods.

And one semi-legitimate one

1. Do exercise 10 on page 115 of the book.

Finally, something useful

Consider the following algorithm for calculating Fibonacci numbers:

```
public static int fibonacci (int n) {
  if (n == 0 || n == 1) {
    return 1;
  } else {
    return fibonacci (n-1) + fibonacci (n-2);
  }
}
```

1. In a sentence or two, sketch a proof that this program terminates for all values of `n`. If necessary, explain any preconditions that are required to guarantee termination.
2. Check out the call tree on page 103 and draw a similar call tree for this method, given 4 as a parameter. How many times do the base cases get evaluated? As the parameter gets bigger, how does this number behave? Does this seem like an efficient way to evaluate Fibonacci numbers?
3. Design a better algorithm to do the same thing. How long does it take, as a function of the parameter? How much space does it require? Implement your algorithm. Was it easier or harder to write than the recursive version. If you had to prove its correctness, could you?

And something not so useful

In an old Saturday Night Live, there was a fake ad for a bank that specialized in making change. That's all they did, but they were very good at it.

You have been hired by this fictitious bank to write a program for them that counts the number of possible ways to make change for a given amount of money. Let's assume that we are dealing with coins only and not concern ourselves with paper money.

Ultimately, you want a method that takes two arguments:

- The number of cents, as an integer.
- How many different kinds of coins can be used in the solution.

In addition, the following information is available in a class variable:

- An array of integers that contains the denominations of the coins. In American currency, this array would contain the values `1 5 10 25 50`.

Thus the invocation `countChange (100, 5)` would ask how many ways there are make change for a dollar, using pennies, nickels, dimes and quarters and half-dollars. The answer is 292.

1. Have I given you enough information to solve this problem or is there more that needs to be specified? If the latter, then feel free to disambiguate in whatever way makes the implementation easiest.
2. Design an algorithm for doing this computation.
3. Implement it.

WARNING: This question is hard. I want you to think about it because it is worth thinking about. Even if you don't come up with an optimal answer, the process of thinking about it will be valuable.

So try not to be frustrated by the difficulty of the problem; try to enjoy the process of thinking about it.

CREDIT: This problem is taken from Abelson and Sussman, *Structure and Interpretation of Computer Programs*, MIT Press, 1987.