ECE242F (Fall 2002) Assignment 2:
Cheese logistics

Danny Heap
heap@cs.utoronto.ca
September 25, 2002

From an early draft of his Canterbury tales Chaucer removed an account of the pilgrims staying at Anne Hoy’s\textsuperscript{1} inn, an establishment that served poor ale but good cheese. The missing account explained how Anne kept her high-quality cheese stacked on stools, the largest rounds underneath the smaller rounds, to stop rats and mice from getting at them.

Occasionally the stool holding the cheese would need some maintenance (for example, the legs would start to buckle under the weight), and Anne would shift the entire stack from one stool to another. Since she could only move a single (hundred-pound plus) round of cheese at one time, and she refused to stack a larger-diameter cheese round on a smaller one (too unstable) she used three stools: one was her destination for her entire stack of cheese, one was the source (which likely needed its legs reinforced), and the third was for intermediate stacking. Chaucer immortalized the complicated routine Anne endured, lugging rounds of cheese from stool to stool as “The tour of Anne Hoy” (TOAH).

One of Chaucer’s pilgrims had a mathematical bent. He had read a very early edition of Data Structures & Program Design in C, by Kruse, Tondo, and Leung and noticed that Anne’s routine for moving cheeses was identical to the problem of moving rings between three posts, described on page 95 of his book. Using this similarity he calculated that to move \( n \) cheeses in this way required \( 2^n - 1 \) moves. This disheartened Anne, who had plans to increase her stack of cheese beyond the 8 she currently had. She decided to invest some of her profits in a fourth stool.

Anne figured that she could do substantially better than \( 2^n - 1 \) moves using the following strategy:

- For a stack of 1 cheese round, her four-stool configuration allowed her to move the stack in 1 move, using her previous three-stool TOAH method.
- For a pile of 2 cheese rounds, her four-stool configuration allowed her to move the stack in 3 moves, using her previous 3-stool TOAH method.
- For a stack of \( n > 2 \) cheese rounds, she reasoned that she could think of some number \( i \) between 1 and \( n \), and then:
  1. Move \( n - i \) cheese rounds to an intermediate stool using all four stools.
  2. Move \( i \) cheese rounds from the original stool to the destination stool, using the (now) only three available stools and her TOAH method.
  3. Move the \( n - i \) smallest cheese rounds from the intermediate stool to the destination stool, using all four stools.

\textsuperscript{1}In Middle English her name would have been spelled Authne H'Oeuil
Notice that steps 1 and 3 require Anne to know how to move \( n - i \) cheese rounds using four stools. Anne figured this wasn’t a problem, since she could apply her recursive strategy to this smaller move. She presented her plan to the above-mentioned mathematically-inclined pilgrim who said that if he called the number of moves needed to move \( n \) rounds of cheese \( M(n) \), and if some \( i \) between 1 and \( n \) where chosen, then (reasoning recursively):

\[
M(n) = \begin{cases} 
1 & n == 1 \\
3 & n == 2 \\
2 * M(n - i) + 2^i - 1 & \text{otherwise.} 
\end{cases}
\] (1)

After experimenting a bit Anne found she could move 3 cheese rounds in 5 moves (a little better than the 7 required by the TOAH method), and 6 cheese rounds in 17 moves — much better than the 63 required by the TOAH method. But the choice of \( i \) made all the different. She (and the aforementioned math-geek pilgrim, who had decided to stay at her inn permanently) spent many hours with early prototypes of pencil and paper, figuring out the best strategies for moving ever-larger stacks of cheese.

This is where matters stood until the invention of the computer.

Your job

You will implement the functions fourStoolMove and moveList specified in FourCheeseStools.h.

fourStoolMove(n) calculates how many moves are required (returned in the moves field of its return value) if you split a stack of \( n \) cheese rounds into the \((n - \text{splitIndex})\) smallest rounds, move these to an intermediate post using four-stool tactics, then move the \text{splitIndex} largest rounds to the destination using TOAH tactics, and finally move the \((n - \text{splitIndex})\) smallest rounds from the intermediate stool to the destination using four-stool tactics.

Since the number of moves must be minimized, you will need to iterate over all choices of \text{splitIndex} and calculate (and keep track of) fourStoolMove(k) results for \( k < n \). This suggests a recursive solution (and one is possible) but be careful that you don’t end up calculating the same result multiple times, since that may well lead to exponential complexity (or even worse, factorial...).

moveList(n, src, dst, i1, i2) returns an array of type move_t that indicates how to transfer \( n \) cheese rounds from src to dst using intermediate stools i1 and i2. For example, moveList(2, A, D, B, C) might return a list that looks like:

\[
\{A, C\} \\
\{A, D\} \\
\{C, D\}
\]

...where \{A, C\} denotes a move_t struct with fromStool A and toStool C. Notice that moving 2 cheese rounds uses the TOAH strategy and ignores stool B altogether.

If fourStoolMove(n) returns a moves field with value \( k \), then moveList(n, A, D, B, C) must return an array with \( k \) elements that correctly indicates how to move \( n \) cheese rounds from stool A to stool D using stools B and C as intermediates.

What to submit

You are to submit your implementations of fourStoolMove and moveList in a file called FourCheeseStools.c, which should include "FourCheeseStools.h". You may not change FourCheeseStools.h. You are welcome to build (and experiment with) TestFourCheeseStools.c, using makefile, but you should not submit either of these two files.

Grading

Assignment 2 is worth 5% of your final mark, distributed as follows
1. 2 points if your FourCheesesStools.c compiles and doesn’t crash (!) when linked to our test code.

2. 5 points (sliding scale) if your moveList moves the cheeses in a minimal number of moves. You get 0/5 if your solution uses $2^n - 1$ moves, since that is trivially possible using the TOAH method. The minimal moves required has never been proved (although plenty of people have tried) but there is a wide consensus on a pattern that includes 3 cheeses in 5 moves, 6 cheeses in 17 moves, 10 cheeses in 49 moves... You’ll get 5/5 if your solution matches this pattern, and bonus points (and probably a new scientific result) if you beat it.

3. 17 points for correctness.

4. 11 points for good organization, modularity, sensible variable names, good comments.