CSC148, Assignment #1
due October 8th, 2013, 11:59 p.m.

October 6, 2013

From an early draft of his *Canterbury tales* Chaucer removed an account of the pilgrims staying at Anne Hoy's[1] inn, an establishment that served bad 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 (the outer part of the larger cheese would droop) 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).[2]

One of Chaucer's pilgrims had a mathematical bent. She had seen a miraculous early draft of the Wikipedia article on the *Tour of Hanoi* in a vision, and noticed that Anne's routine for moving cheeses was identical to the problem of moving rings between three posts. Using this similarity she 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 stack of $n > 1$ cheese rounds, she reasoned that she could think of some number i from 1 and $n - 1$, 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

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 she called the number

[1] In Middle English her name would have been spelled *Auyne H'Oeuil*

[2] Chaucer conjectured Anne's muttered "cheeses crust!" and "gentle jumping cheeses!" were veiled religious oaths, not mundane references to cheese maintenance.

of moves needed to move n rounds of cheese $M(n)$, and if some i between 1 and n where chosen, then (reasoning recursively):

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

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 difference. 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, for centuries, until the invention of the computer.

Your job

You will implement a step-by-step design of an object-oriented design for representing and moving Anne's stools of cheese. In order to guide you, we present the steps below, in the order we recommend. Notice that you get credit for each step you complete.

We really that this is a challenging assignment. We aim to help guide you to a successful submission, provided you follow the guidance we offer. You may work on this assignments in groups of 1, 2, or at most 3.

Step 0: Read and understand the Cheese class in DomainStools.py.

Step 1: Read the documentation for the `CheeseView` class (and its methods), in CheeseView.py.

(i) Complete the implementation of `CheeseView`'s constructor by filling in the missing code described by the "TODO" comments inside it.

Those "TODO" comments (and this handout) use Object-Oriented terminology to describe what to do, and rely on an understanding of the OO concepts they refer to. If any of the concepts they refer to are unclear to you, please come for help with them as soon as possible.

(ii) Add the missing `place` method that is used by the constructor to position the rectangle (and later by `ManualController` objects to move it). To determine the header for `place`, read the constructor code to see how it calls `place`. Implement `place` using the following tkinter `Canvas` method which positions a rectangle:

```
coords(rectangle_index, coordinates).
```

Call it on the stored canvas, passing it the stored index returned from the constructor's call to `create_rectangle`, and a 4-tuple of coordinates (x0, y0, x1, y1) for the top-left and bottom-right (from the screen's perspective) corners of the new position of the rectangle. Also, record the parameter values in instance variables: they will be used by `ManualController`. Name the instance variables using the same names as in the call to `place`.

Steps 0–1 are worth 50% of the credit for this assignment.

Step 2: Read the `ManualController` class in ManualController.py, looking for how it uses a `DomainStools` object. Write the method headers for `DomainStools` in DomainStools.py according to their uses by `ManualController`.

Steps 0–2 are worth 60% of the credit for this assignment.

Step 3: Write the implementations of the `DomainStools` methods from Step 2. Notice in particular how `ManualController` relies on a `DomainStools` object to throw exceptions in certain circumstances (and without altering the state of the `DomainsStools` object). When done you should be able to run ManualController.py to manually play with Anne Hoy's cheeses.

Steps 0–3 are worth 70% of the credit for this assignment.

Step 4: Implement the module-level function `tour_of_four_stools` in Tour.py to solve a four stool instance of Anne Hoy's cheese-moving problem, trying to minimize the number of moves. You should be able to move n cheeses in considerably less than $2^n - 1$ moves achievable with just three stools.

You may add a method to `DomainStools` to refer to the cheeses you decide to move. The method should know nothing about solving, and not expose your internal representation of the stools.

Steps 0–4 are worth 85% of the credit for this assignment.

Step 5: Implement `SolvingController` to animate your four stool solution. You may continue to add an extra Cheese to each stool, as in `ManualController.py`, or you may modify `DomainStools.py` to make this unnecessary.

Start by copying the `ManualController` implementation, and make some changes (the fewer the better). In particular, a user clicking cheeses should not have any effect (other than, perhaps, beginning the animation). DO NOT use the module-level function `tour_of_four_stools` in directly. Instead re-implement the logic inside `SolvingController`, along with animating it. Remove the parameter for the number of stools, since it is always four, and add an extra (final) parameter for the number of seconds to pause between highlighting a cheese to move and moving it.

Steps 0–5 are worth 100% of the credit for this assignment.

Step 6: Implement a class `Controller` in a file `Controller.py` as a partially abstract superclass for both of `ManualController` and `SolvingController`, Put as much of the common code into `Controller`, and make versions (in that same file `Controller.py`) of `ManualController` and `SolvingController` that each subclass `Controller`. They must behave the same as they did before.

Steps 0–6 are worth 100% of the credit for this assignment.

WTF![3] Weren't Steps 0–5 also worth 100%?!

Yes. Step 6 gains you feedback and satisfaction.

Those licenses: Notice that of the starter code that we provide is distributed with a GNU GPL license. One consequence of this is that any copies of that code, or any work you derive from it (for example the files you submit to MarkUs), must be similarly licensed if you pass it on to someone else — you may just add your name to Gary's in the license declaration, and include the COPYING notice when passing it along.

[3] What the fact...

What to submit

You'll need to submit your version of the following files to MarkUs:

- `CheeseView.py`
- `DomainStools.py`
- `Tour.py`
- `SolvingController.py`
- `Controller.py`