Lab: Stirling Numbers, Bell Numbers, and Integer Partitions
CSC/MAT 208: Discrete Structures
Department of Computer Science
Department of Mathematics and Statistics
Grinnell College

Pairs

Here are the programmer pairs for today’s lab. Drivers are on the left.

- Michael Owusu and Julia Fay
- Adam Hudson and Muhammad Hamza
- Adam Wesely and Ella Nicolson
- Bazil Mupisiri and Elizabeth Zak
- Tyler Williams and Thu Nguyen
- Lindsey Byrne and Anita DeWitt
- Linda Oyolu and Jacob Ekstrand
- Jong Hoon Bae and David Chang
- Dennis Chan and Corey Simmonds
- Matt Jasaitis and Madeleine Goldman
- Kathryn Yetter and Marcel Champagne
- Amanda Hinchman-Dominguez and Danielle Williams
- Samee Zahid and Kevin Connors

Devin Dooley will replace anyone who is absent.

Stirling numbers of the second kind

As we saw in today’s reading, it is easier to count the partitions of a class if we consider each possible number of parts separately. To partition a class $C$ with $n + 1$ members into exactly $k + 1$ mutually exclusive and jointly exhaustive non-empty subclasses, we pick out one of its members, say $c$. We then find all the ways of partitioning $C - \{c\}$ into exactly $k$ subclasses, and add the subclass $\{c\}$ to each of those partitions. Then we find all the ways of partitioning $C - \{c\}$ into exactly $k + 1$ subclasses, and generate $k + 1$ partitions of $C$ from each of them by adding $c$ to each of its “compartments” in turn.
If we write ‘$S(n, k)$’ for the number of ways of partitioning an $n$-member class into $k$ pieces, the construction just described shows that

$$S(n + 1, k + 1) = S(n, k) + (k + 1) \cdot S(n, k + 1),$$

since there are $S(n, k)$ partitions of $C - \{c\}$ into $k$ parts, each producing one partition of $C$, and $S(n, k + 1)$ partitions of $C - \{c\}$ into $k + 1$ parts, each yielding $k + 1$ partitions of $C$.

This recursion applies for all positive integers $n$ and $k$ such that $k < n$. For the cases not covered, take $S(n, n) = 1$ for every natural number $n$, and $S(n, 0) = 0$ for every positive $n$.

**Exercise 0:** Define a procedure `stirling2` that computes and returns $S(n, k)$, given $n$ and $k$ as arguments.

*Hint:* The recursion equation above defines $S(n + 1, k + 1)$ in terms of $S(n, k)$ and $S(n, k + 1)$. To implement this in Scheme, on the other hand, you’ll have to express $S(n, k)$, for positive values of $n$ and $k$, in terms of $S(n - 1, k - 1)$ and $S(n - 1, k)$. You may want to rewrite the recursion equation on paper, making the necessary adjustments, before translating it into Scheme code.

### Bell numbers

The number of class partitions of a class of cardinality $n$ is called the $n$th Bell number, $\mathcal{B}_n$.

**Exercise 1:** Define a procedure `bell` that takes $n$ as argument and returns the $n$th Bell number, computing it as a summation of Stirling numbers.

**Exercise 2:** The second section of the reading suggests another way to compute Bell numbers, using a recursion that involves binomial coefficients. Implement that algorithm as a Scheme procedure.

### Integer partitions

An integer partition of a natural number $n$ is a bag of positive integers whose sum is $n$. For instance, the bag $[3, 3, 2, 2, 1]$ is an integer partition of 11, since $3 + 3 + 2 + 2 + 1 = 11$. 
Once again, it is easiest to count integer partitions if one considers each possible bag cardinality separately. In the reading, we derive the recursion equation

\[ I(n + 1, m + 1) = I(n, m) + I(n - m, m + 1), \]

where \( n \) and \( m \) are positive integers such that \( m \leq n \).

**Exercise 3:** Write a procedure `sized-integer-partition-count` that takes \( n \) and \( m \) as arguments and computes \( I(n, m) \). As in exercise 0 above, you'll need to rewrite the recursion equation carefully in order to put it in a form that can be translated directly into Scheme code.

**Exercise 4:** Define a procedure `integer-partition-count` that takes a natural number \( n \) as argument and returns the number of integer partitions of \( n \).

**Hint:** Use `sized-integer-partition-count` and the summation procedure (or the design pattern it expresses).

**Exercise 5:** The combinatorial argument that we used to arrive at the recursion equation for \( I(n, m) \) suggests a strategy for generating the class of all integer partitions of a given natural number. Implement this strategy as a Scheme procedure.