Lecture #19: More Recursion

Announcements:
- HKN Review Session for CS 61A Exam 2
  Sunday, 4 March 2012
  3PM-6PM
  306 Soda (HP Auditorium)
- Occupy Woz (HKN tutors and CS 61A/CS 61C tutoring):
  Now-11PM Saturday, 3 March 2012
  Tutoring for CS 61A/CS 61C both days until 11 PM
  tinyurl.com/occupywoz

Example II: Counting Ways to Make Change

Given the same arguments, how many different ways are there to make change?

```python
def count_change(amount, coins=(50, 25, 10, 5, 1)):
    """A sequence of integers giving a number of each type of coin in COINS
    such that the value of the indicated numbers of coins will be exactly AMOUNT.
    >>> # 9 cents = 1 nickel and 4 pennies, or 9 pennies
    >>> count_change(9)
    2
    >>> # 12 cents = 1 dime and 2 pennies, 2 nickels and 2 pennies,
    >>> # 1 nickel and 7 pennies, or 12 pennies
    >>> count_change(12)
    4
    ""
```

Example III: Escape from a Maze

Consider a rectangular maze consisting of an array of squares some of which are occupied by large blocks of concrete:

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14
0 X X X X X X X X X X X X X X X
1 X X X X X X X X X X X X X X X
2 X X X X X X X X X X X X X X X
3 X X X X X X X X X X X X X X X
4 X X X X X X X X X X X X X X X
5 X X X X X X X X X X X X X X X
6 X X X X X X X X X X X X X X X
7 X X X X X X X X X X X X X X X
8 X X X X X X X X X X X X X X X
9 X X X X X X X X X X X X X X X
```

Given the size of the maze and locations of the blocks, prisoner, and exit, how does the prisoner escape?

```python
def solve_maze(start, exit, maze):
    """Assume that 'maze' is a 2D array (list of lists) where
    maze[r][c] is true iff there is a concrete block occupying
    column 'c' of row 'r'. 'start' and 'exit' are (row,column)
    pairs indicating the initial position of the prisoner and the
    position of the exit. Returns a sequence of (row,column)
    pairs starting with 'start' and ending with 'exit' indicating
    a sequence of empty squares that are adjacent to each other
    vertically or horizontally.
    ""
    def search(p, visited):
        """Returns a list of pairs starting with 'p' and ending
        with 'exit' of empty, adjacent squares, none of which
        are contained in the list of squares 'visited'.""
        # FILL IN HERE
        return search(start, ())
```

Maze Program