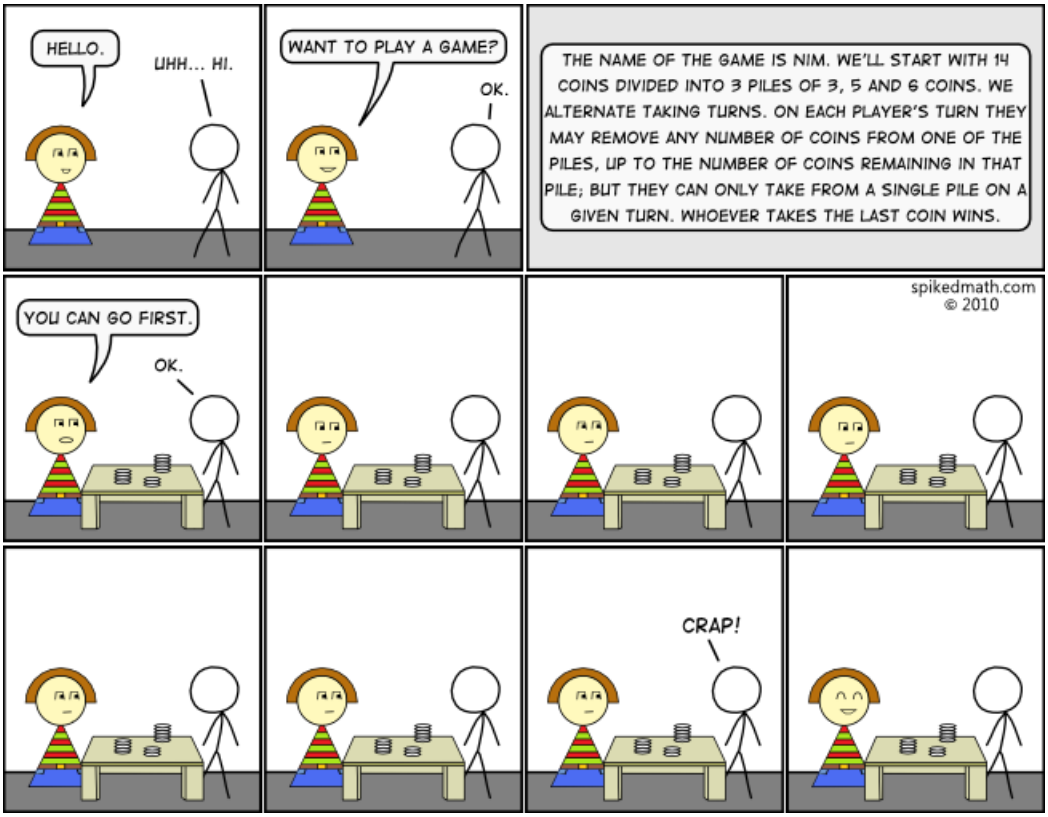
CS51A - Assignment 11

Due Tuesday, April 28



`http://spikedmath.com/221.html`

The game of Nim has two players and K piles of coins. Players take turns, and on each turn a player must remove one or more coins from a single pile. The player who takes the last coin wins.

The goal of this assignment is to create the best 4-pile nim player that you can. You may work in groups of up to four people if you'd like, but pairs may be more efficient. If you do work in a group, as always, all people in the group must be there when any person from the group is working on the assignment.

Starter code

Create a project called `assignment11`, download the starter code into this directory, and unzip the contents.

```
http://www.cs.pomona.edu/classes/cs51a/assignments/assign11-starter.zip
```

The starter codes contains two files, `assign11.py` where you'll write all of your code, and `nimsupport.py`, which contains a number of things that help us play nim.

If you look inside the `nimsupport.py` file, you'll see a few things:

- `NimGame` class that models the nim piles and the interactions.

  We will represent the piles as a list of integers. Take a look at the methods available. The only one your nim player should be calling is `get_piles` which returns the current pile configuration.

- `play_game` function that is used to play two players against each other. It takes two player *functions* as input and a game.

- Three different nim player functions:

  – Two functions that implement basic strategies:
  `nim2_strategy` only works for 2-pile nim games and tries to keep each of piles equal.
  `random_nim_strategy` works for any nim game, but just picks a random non-empty pile and picks a random number to remove from that pile.

  – `human_player` which prompts the user for input and allows you to play against either another human or an nim player function.

Some example runs

If you download the starter code, you can immediately run some of the basic strategies. For example:

- Play as first player against `nim2_strategy`:
  `play_nim(human_player, nim2_strategy, NimGame([4] * 2))`
- Have a person to person game on a 4-pile board (`[4, 4, 4, 4]`):
  `play_nim(human_player, human_player, NimGame([4] * 4))`
- Play `nim2_strategy` against itself on a [4, 4] game board:
  `play_nim(nim2_strategy, nim2_strategy, NimGame([4]*2))`
- Play the random player against itself on a [5, 5, 5, 5] game board:
  `play_nim(random_nim_strategy, random_nim_strategy, NimGame([5]*4))`

- Play `nim2_strategy` against the random player on a [5, 5] game board (this one is the saddest to run repeatedly... poor random player):

  ```
  play_nim(nim2_strategy, random_nim_strategy, NimGame([5]*2))
  ```

Rules and requirements

You are to write your own strategy function.

- Your function should be called `nim_strategy`.
- Your function should take as input a list of integers (representing the piles) and return a tuple with the first element being the pile number and the second element the number to remove from that pile.
- Your player will be tested on any number of piles up to 10 and any number of coins in each pile up to 100. Your player should work for any game configuration within these ranges.
- Your player must not take more than 1 second to calculate its move.
- No looking online! Nim is a game that has been fairly extensively discussed. The point of this is for you to think about the problem and strategies and come up with your own ideas.
- Do not put any code in `nimsupport.py` or modify any code in there.

Strategy

There are many ways to tackle this problem. Be creative! I don't expect you to take a search-based approach, but you can try something along those lines if you'd like. The most common type of strategy is something heuristic that looks at the current state of the board and tries to make an educated decision. That said, a combination of these two things can be very effective.

Work iteratively and try many different things. You can play different versions against each other and/or play against your teammates.

Writeup

In addition to writing your nim strategy, you must also submit a short write-up containing 2–3 paragraphs as a multiline string at the top of your file. You should discuss two things: 1) a description of your player's strategy and 2) how you came to that strategy. For the second part of this question, you should discuss what are the other things you tried, experiments you ran, etc.

Tournament

Time permitting, I will setup a tournament between all the submissions and will play them against each other. You are not required to enter the tournament (though it should be a lot of fun). If you'd like to enter the tournament, please put a line starting with "Team name: " followed by the name of your nim player, e.g.,

```
Team name: Nimble Nim Player
```

at the top of your file.

When you're done

You should have a single file `assign11.py` with

1. The team name (optionally) at the top
2. The writeup as a multiline string
3. Your `nim_strategy` function and any supporting functions.

Comments/style:

- You should have comments at the very beginning of the file stating your name, course, assignment number and the date.
- Each function should have an appropriate docstring.
- Include other miscellaneous comments to make things clear.

Submit your .py file online using the courses submission mechanism.

Grading

Because this assignment is a bit open-ended, I will be grading your nim player based on a number of components, including:

- How well does your player play? You don't have to win the tournament, but I expect you to put some time and effort into achieving a good algorithm.
- How thoughtful/creative is the solution?
- How much time and effort does it appear that went into developing the player?

- How is the quality of the implementation? Make sure to submit code that is well-written, understandable and commented.

Grading

	points
player function works	4
quality of player function	7
writeup	6
comments/style	3
total	20