CS35X: Competitive Programming

Lecture 1: Introduction, I/O

Joshua Brody

What is Competitive Programming?

- Teams of three students collaborate to solve coding problems
  - Each team has one computer
  - 8-10 problems of ***varying difficulty***
  - Students can use reference books, but start with no code
  - Score is # problems solved; tiebreaker: time to solve, penalties
- Solutions are submitted to **online judge**

This Class

We will use competitive programming problems to explore ADTs and data structures and gain expertise in algorithmic problem solving.

- Hands-on practice in class
- Weekly graded problem sets
- 2-3 practice contests
- 2-3 quizzes

YOU: consistent practice and effort, 4+ hours per week.

Exercise: set up Kattis accounts

CP Style Problems

Competitive Programming style problems all have a similar structure:

1. **Flavor Text**: description of problem, what you should solve
2. **Input** Specification: precise description of input
3. **Output** Description: precise description of what to output
4. ***Sample input/output(s)***

Note: all Kattis problems use standard input/output (**cin, cout**).

Kattis Problem: hello

Basic UNIX skills

- Make a directory: `% mkdir cs35x`
- Change into a directory: `% cd cs35x`
- Where am I? `% pwd`
- What files are in this directory? `% ls`

Development Process Recap

1. **Make a directory for each problem**
2. **Write sample inputs into files in that directory**
3. **Think about how to solve your problem**
4. **Code your solution in a single C++ file.**
5. ***Test your solution on sample inputs***
6. Are you confident the solution works? ***Submit to online judge!***
7. **Debug/resubmit** until solution is accepted.

C++ Input/Output

Every problem takes in input and produces output, so it's important to understand exactly how you'll process I/O!.

You should know how to:

- Read in one input variable at a time
- Read in an entire line of input at once
- Detect end-of-file.

C++ Standard I/O

- Load Standard I/O Stream library: `#include <iostream>`
- Read one variable at a time:
  - `int x;`
  - `cin >> x;`
- Read in an entire line:
  - `String input;`
  - `getline(cin, input);`
- Detect end of file:
  - `while(getline(cin, input)) {…}`

Kattis Problem: *echoechoecho*