Competitive Programming

Core directive: “Given well-known computer science problems, solve them as quickly as possible!”

What does this mean?

- “well-known” = no open research problems
- “computer science” = algorithms and data structures
- “solve” = matches output on test cases in reasonable run time
- “quickly” = natural human desire to compete
Tip 1: Type Code Faster!

No kidding!

Typing skills are important
- However, accuracy matters as much as speed
- Try the test: http://www.typingtest.com
- Goal: at least 50 awpm (adjusted words per minute)
- Stretch goal: 80+ awpm

Learn keyboard positions of programming symbols
- Grouping: ( ) { } [ ] ‘ ” : ;
- Operators: < > & | ! + - * /
- Don’t become dependent on a particular keyboard!
Tip 2: Quickly Identify Problem Types

By the end of this course, you should be able to quickly categorize problems by their problem type: (* = more frequent)

- “Ad hoc” (*)
- Complete search (*)
- Divide and conquer
- Greedy
- Dynamic programming (*)
- Graph (*)
- Mathematics (*)
- String processing
- Computational geometry
- “Harder/rarer”
Tip 2: Quickly Identify Problem Types

At the beginning of a contest, briefly read all the problems and attempt to categorize them by difficulty:

- **Easy**: Solve it quickly and get some points on the board!
- **Moderate**: Spend the bulk of your time here unless you’re aiming for the win, in which case you should solve these quickly and move on.
- **Hard**: Stretch goals unless you’re aiming for the win, in which case you should spend the bulk of your time here.
- **Challenge**: Intended to be difficult even for the best teams. Avoid these unless you are aiming for the win.

You may need to re-evaluate your categorization of particular problems later.
Tip 3: Do Algorithm Analysis

▶ Remember Big-O analysis from CS 240?
▶ It’s now your best friend!

▶ Do some basic analysis to determine feasibility
▶ Look for assumptions about the input size in the problem statement
▶ Choose the simplest solution that will work
▶ Check out Table 1.4 in the textbook
Tip 4: Master Programming Languages

- C++ is preferred for the major contests
- However, familiarity with other languages always helps
- Try to learn all the features of your chosen language
- Use shortcuts, macros, and libraries

**DURING CONTESTS ONLY:** Avoid commenting
  - OR: Use comments as stubs
    - Come back and add documentation afterwards!

- For CS 240 students, we recommend using Java for contests this semester to avoid issues with C/C++ incompatibilities
Tip 5: Master the Art of Testing Code

We’ve all experienced it: you think you’re done and you submit, but then you don’t get AC.

Other possibilities:

- PE: Presentation Error
- RTE: Runtime Error
- TLE: Time Limit Exceeded
- MLE: Memory Limit Exceeded
- WA: Wrong Answer

**Debugging** is a very important acquired skill in CS; cultivate it deliberately!
Tip 6: Practice!

See your UVa stats at [http://uhunt.felix-halim.net](http://uhunt.felix-halim.net)
- Just click your name on the weekly scoreboard

Goal: complete the textbook’s starred exercises
- Scroll down to table of contents, click each chapter

Other practice sites:
- [http://naipc.uchicago.edu/practices/](http://naipc.uchicago.edu/practices/)
- [https://www.topcoder.com/](https://www.topcoder.com/)
Problem solving process

Work first to solve the problem on your own

- **Frustration level 1**: Ask for hints and/or suggestions
- **Frustration level 2**: Have someone look at your code
- **Frustration level 3**: Look at someone else’s code

Go away for a few days (or wait until next week)

- Solve the problem from scratch on your own

After you’ve solved the problem

- Study other solutions and improve yours
- Internalize (not memorize) your solution
Personal Notebook

Things to include:

- Common algorithms and data structures (e.g., from CS 240)
- Commonly-used reference pages (STL, etc.)
- Your coded problem solutions
  - Document them first!
- Common code snippets
  - Frequently-used methods, debug macros, boilerplate, etc.

Avoid:

- Unfamiliar code (it won’t help you)

Basic idea: Include anything you had to search for while solving problems, because you won’t have internet access during competitions
Tip 7: Teamwork

If you are here just for fun (or the course), then have fun!
  ▶ Working in teams can be fun too!

If you are here to compete in the official contest, then pay close attention to the next few slides.
  ▶ The ACM ICPC contest is a team contest.
Tip 7: Teamwork

One possible team composition:

- **Problem solver** ("the professor")
  - Often the most theory/math-minded member
  - Quickly comes up with ideas on how to solve problems
  - Writes pseudocode solutions on paper

- **Coder** ("the code jockey")
  - Often the member with the highest attention to detail
  - Translates pseudocode solutions quickly and accurately
  - Handles the actual testing and submission process

- **Debugger** ("the hacker")
  - Often the most engineering-focused member
  - Can quickly identify ways that code could break
  - Works with solver and coder to identify and fix defects

You will only have **ONE** computer per team - learn to share it!
Tip 7: Teamwork

Before the competition:

- Evaluate your strengths and weaknesses
- Form a team with people who have complementary strengths
- Spend time practicing with your team
- **Practice coding and debugging on paper**
- Practice inventing edge/corner test cases
- Solve a **LOT** of problems
  - Try to solve a wide range of problem types
  - Try to solve at least a few hard problems
- Assemble and curate your notebooks
- Hang out with your team outside class
Tip 7: Teamwork

During the competition:

▶ **Don’t panic!**
▶ Have a well-defined game plan for the first 15 minutes
  ▶ Read all the problems
  ▶ Type in boilerplate code and solve the easiest problem
  ▶ Ask the administrators early about hardware/software issues
▶ Use the “submit and print” strategy
▶ Be flexible and adapt to the circumstances
  ▶ It’s a marathon, not a sprint
  ▶ Re-evaluate problems occasionally throughout competition
  ▶ Check the scoreboard often for hints
  ▶ Keep in mind that there is no partial credit
▶ **Take pride in your work, and have fun!**
Today’s Contest

Form groups of 2-3 people.

▶ If you are in CS 240, consider finding other CS 240 students and form groups to use Java instead of C++.

▶ If you are planning to compete more seriously in the regional contest, try to form a group with the two others that you wish to work with at the actual contest.

▶ Use only ONE computer per team. Before you begin, talk amongst your team about how you will share the computer.

You need to register for the online qualifier contest TODAY!

We need to finalize teams for the regional contest within the next couple of weeks.