Course Outline, CSE 232, Fall 2014 Semester

Dr. Brett Olsen

Programming Skills Workshop

Class Description

This course provides an overview of practical programming skills with a focus on preparation for major programming competitions such as the International Collegiate Programming Contest and the Google Code Jam. Topics will include fast problem analysis, algorithm design and implementation, testing and debugging, and team programming. Prerequisites: fluency in a modern programming language and knowledge of major algorithms and data structures.

Introduction

This course is designed to prepare you to compete in major programming competitions such as the International Collegiate Programming Contest (ICPC) and the Google Code Jam. We will cover the details of these specific contests as well as general strategies for programming competitions and fast, efficient programming and a review of important algorithms and data structures important to a competitive environment. During the fall semester, we will be paying particular attention to the 2014 Midwest Regional ICPC competition on November 1st, and we encourage students to participate.

While the specific goal is to prepare you for programming competitions, all students can benefit from the course. You will gain skills in quickly analyzing and solving difficult algorithmic challenges. These skills will benefit you not only in programming competitions but in your ordinary work and (particularly) in technical interviews for programming jobs.

Prerequisites

While there are no explicit course requirements for taking this class, we do expect both basic fluency in a modern programming language and an understanding of major algorithms and data structures. The course itself will be largely language-independent, but if you plan to participate in the ICPC as a team member, fluency in either C++ or Java is necessary, and the ACM, which will be administering the Washington University ICPC teams this year, suggests Java as a language of choice for their teams.

Contact

The instructor for the course will be Dr. Brett Olsen, assisted by TAs Joey Woodson and Pengning Chao.

For questions about specific language syntax, methods, and libraries, we suggest you contact:

- Java: Joey Woodson
- C++: *TBD*
- Python: Dr. Olsen

The course website can be found at `https://acm.wustl.edu/cse232/`. We will be posting lecture notes, homeworks, handouts, and lecture videos on the website. We have also set up a Piazza site at `https://piazza.com/wustl/fall2014/cse232/home`. We will invite all students in the class to Piazza once we get your email after the first class.

We can also be contacted through email. Please put "CSE 232" in the subject line of your email so email filtering will make sure it doesn't go to spam and we can find everything appropriately. *E.g.*, "CSE

232 - Homework #1" or "CSE 232 - question about lecture 2". Homework submissions should be sent to Dr. Olsen. Questions and comments should be posted to the Piazza site.

- Brett Olsen
  `brett.olsen@gmail.com`
  `bnolsen@go.wustl.edu`
- Joey Woodson
  `josephcwoodson@wustl.edu`
- Pengning Chao
  `pengningchao@wustl.edu`

Office hours are *TBD*.

Times and Places

Class meetings will be 11:30-1:00 PM on Fridays in Urbauer 214.

We'll also be doing practice competitions at as-yet unscheduled times and places during the semester. These times will be set during the class depending on student and lab availability.

The regional ICPC competition will be November 1st, the day after class, so we will expect students participating in the competition to be absent from class. We'll plan on scheduling an additional practice competition during class time on October 31st for those students not participating in the ICPC.

Class Policies

Grading & Collaboration Grading is pass/fail. Passing requires completion of 80% of homework assignments and participation in at least one of the practice programming competitions. Completion doesn't necessarily mean getting the answers correct; it means making a significant effort to solve the problem and gain an understanding of the principles behind it.

Collaboration on homework problems is not only allowed but encouraged! However, all work you turn in must be either completely your own work from start to finish or clearly annotated (with comments) where it includes someone else's work. That means that discussing problems with other students is always fine. Borrowing code for a particular function you need, from another student or Stack Overflow or wherever, is fine too - as long as you comment it to let us know you borrowed it. If you can't explain to us exactly how your answer works, that means it's probably not yours and it should be annotated appropriately.

That said, we think that you'll get more benefit from the class if you make a strong effort on the problems before going to external sources. If you get stuck, though, we'd much prefer that you borrow someone else's work or ideas for a difficult step in a problem and then finish the rest of the problem yourself than that you just give up.

Collaboration on practice competitions is obviously not allowed at all. Generally, no discussion of problems will be allowed, but books and external resources are.

Violation of the collaboration policies will result in the offending assignment or competition being marked incomplete.

Homework We will provide homework assignments weekly at the end of each class that relate to what we have covered in class. Homework will be due (via email) by the following Wednesday, before the next class meeting.

Textbook There is no set textbook for this course. We'll provide practice problems and homework separately.

However, the following texts are highly recommended. Unfortunately, the bookstore has proven unwilling to stock them for us, so you'll have to get them independently.

Competitive Programming 3

This book focuses on specific strategies for programming competitions, guidelines for types of problems you'll see, specific strategies for the ICPC, and an enormous stash of sample problems linked to the UVa online

judge. Earlier versions of this book are available for cheap (version 2) or free (version 1) in e-book form as well, so there's no excuse for at least getting the early ones.

Introduction to Algorithms (CLRS)

This is *the* algorithms book for detailed information on most standard algorithms and data structures. If you don't already have it for another class, it will be very useful for this one.

The Algorithm Design Manual

While the first half of this book focusing on explanation and analysis of algorithms and data structures is good, it's not as good as CLRS. The second half is unparalleled - it contains a 2-3 page description of nearly every standard data structure and problem type that you could possibly want, including links to existing libraries that solve the problem. If you know that a particular problem is, for example, an instance of a maximum flow problem or a set cover problem, then you can crack open this book, turn to the right page, and get a guideline on how to solve it very quickly.

Online Judges Most of the practice problems, some of the homework problems, and much of the in-class competitions will be done using various online judges. These are various web-based systems that allow submission of code to solve a particular problem and will automatically judge the submitted code on correctness and speed. We encourage you to practice with these judges yourself; practice on quickly solving problems is the best way to improve! Note that for most of the online judges there is often commentary or code that solves the problem available if you Google or look in the right place. We strongly recommend against using these resources before putting work into the problem - often identifying the correct approach is the most difficult thing to learn, and going straight to solutions will make it harder to learn how to do this.

We'll largely be using the following, as they allow submission in arbitrary languages (or at least a pretty good range!):

- Timus Online Judge
- Sphere Online Judge
- Google Code Jam Practice Problems

Another very commonly used online judge is the UVa Online Judge. We won't be using it for class work because it's more language restrictive, but if you're preparing for the ICPC it's highly recommended.

Class Schedule

Week 1 (August 29) Introduction to Competitive Programming

Week 2 (September 5) Data Structures

Week 3 (September 12) Dynamic Programming & Memoization

Week 4 (September 19) Strings

Week 5 (September 26) Binary Mathematics (and ICPC Regional Qualifier)

Week 6 (October 3) Introduction to Graph Theory

Week 7 (October 10) More Graph Theory

Fall Break (October 17)

Week 8 (October 24) Computational Geometry

Week 9 (October 31) Regional ICPC & Practice Competition

Week 10 (November 7) Probability and Combinatorics

Week 11 (November 14) Number Theory

Week 12 (November 21) Random Algorithms

Thanksgiving Break (November 28)

Week 13 (December 5) Hard Problems