Instructions for Exercises

Throughout the semester, I will distribute handouts with problems to give you more practice with the material. We will occasionally go over solutions to problems from handouts in lecture.

The problems on handouts won't be listed in any order. Just because a problem is listed later doesn't mean that it's harder, or that the material necessary to solve it was covered later in the course. I will indicate which problems I consider easy and which ones are challenging.

Emailing Me Your Solutions

Send your solutions to dalibor@cs.wisc.edu. You don't need to send solutions to all problems at once. Just send me solutions (or even partial solutions including explanations why you are stuck and can't figure out what to do next) as you get them done.

If I get a solution before noon on Sunday, I'll make a printout with your solution, write some comments on it, and give it to you in lecture on Tuesday (or I'll email you my comments if you prefer that, so let me know in the email which way you prefer). Send .txt files for pseudocode and other "essay" questions, .java files for code, or pdfs for both pseudocode and Java code. Please don't send me files in any other formats such as Word documents. If you do, I'll send a reply back asking you to resend them in the requested format if you do. If you're sending multiple files, it's ok to send a zip archive with all your files.

My Philosophy Behind Problem Handouts

I'd like to give you programming practice without you stressing about grades. You don't get that much feedback in this course, so you can view these exercises as additional ways of getting feedback on your performance. I will be happy to discuss the problems with you, and I do give hints for these problems. Just come to my office hours or send me an email. My office is 5381 CS, and my email is dalibor@cs.wisc.edu.

I have a mix of exercises of varying difficulty. There are some easy problems for the people who struggle with the material. I hope they help them get back on track faster, or that they simply give them the additional experience necessary for being comfortable with the material. There are also some difficult problems for the students who understand everything easily. That way the outstanding students also get an opportunity to learn something in this course.

Problem Solving Suggestions

Here are some problem solving suggestions. Not all of them will probably make sense to you right away. They should start making more sense as the semester goes on.

- Think in English first. Then write pseudocode as comments. Finally, write your Java code.
- Sometimes a simple picture is worth a thousand words.
- You can carry out simple examples of the problems you're working on using paper and pencil. This may help you figure out what you need to do in general in order to solve a particular problem.
- If you're stuck, talk to someone about the problem (this includes sending me an email or talking to your friends). Just talking your way through why you're stuck may help you realize what you need to do. I'll be more than happy to help you, and will give you hints.
- When you program something, make sure you test it. For example, you should run your program on various inputs, or you should trace through it by hand. Not only that, you should do this as soon as possible. It is a good idea to test even partial solutions.
- Some problems require you to write programs that take a lot of input. You probably won't want to enter it all by hand. Even though the exercises say that the input comes from the keyboard, you could just generate random input using the `Random` class. You could also store all the data in some other class. As an alternative, you could also write your input into a file and use File I/0 (ask me if you want to know how). That way, you won't need to type in the input every single time you run your program. Yet another option is input redirection, but I'm not sure how to do it in a Windows environment. So if you do know how to do it in Windows, I'll be happy to learn from you.
- Some of the problems are open ended to give you some room for creativity. It's always interesting to see what improvements you can come up with.
- If you come up with your own problems, share those with your friends or with me. This is actually a good exam preparation technique. You'll also find out that you come up with harder problems than what we can ask on exams. Sometimes, you may also come up with a problem that is similar to an exam question.