

Success Guide for COMP-1410

Curtis Bright, updated Fall 2023

Introduction

Welcome to COMP-1410, the second programming course offered by the School of Computer Science at the University of Windsor. This will be your first dive into the real “heart and soul” of computer science. More than just a second course in programming, in this course we start putting the *science* into computer science. For CS majors, the material in this course will be foundational to your future courses. Later courses that crucially rely on this material are COMP-2540 (Data Structures and Algorithms), COMP-2560 (Systems Programming), and COMP-3300 (Operating System Fundamentals). For non-CS majors, an understanding of computer science can provide you with a competitive edge in nearly any career, especially as computers become more and more universal.

Students tend to find this course more difficult than COMP-1400, because more topics are introduced, they are studied in greater depth, and new ways of thinking are introduced. As a result, this is not a “bird” course, but with the proper habits and strategy it is well within the ability of all university students to complete. This guide was written to help put you on the proper path.

Logistics

Regardless of your past history, if you want to do well in university courses your best strategy is to develop successful habits. University is a time in your life when you—perhaps the first time—will be given the freedom to set your own habits. No one will chase after you if you don’t attend the lectures or submit your coursework. It is all-to-easy to slip into the tempting habit of thinking “I can skip today and catch up later”. Regardless of if this is true or false, experience shows that the students that *consistently* make an effort do the best in the course.

Each part of the course is designed to help you learn the course material: the lectures and textbook explain the concepts; the labs and assignments give you an opportunity to practice and receive feedback; and the midterm and final exam allow you to demonstrate your understanding.

Lectures

Lectures introduce the primary material in the course. Attending lecture *in and of itself* will not help you much—computer science is not something you can pick up by mere listening—but this is not an excuse to skip lecture.

Going to lectures reserves three hours of dedicated time each week during which you are thinking about course material, with the aid of someone who understands it thoroughly. It's a rare student who is disciplined enough to skip lectures and adequately substitute for that experience. Of course, not every minute of lectures is “quality time”, but you won't know what you're missing if you don't attend.

Survival Guide, University of Waterloo

Lectures in this class tend to be interactive—I ask many questions of the class throughout the term. The students that do the best in the course are the ones that attend lecture, week after week, participate by providing their input, and even ask their own questions about the course material.

After lectures you should spend a significant amount of time reviewing the material and practicing the concepts on your own. The amount of time is different for everyone but a good rule of thumb is to spend at least 2–4 hours outside of lecture for every hour in lecture. You should aim to understand everything on the slides and be able to write *by yourself* all the functions we implement in class.

Labs

Labs are run weekly and give you the opportunity to put into practice the material covered in lecture. Labs are run by graduate students and upper-year undergraduate students. Each week there will be a “lab assignment” to complete. The purpose of the labs is to help you understand and apply the course material. They typically can be solved by a direct application of the material from lecture, so if you can't complete the lab it is a sign you need to catch up. Lab instructors are also happy to answer your questions about the material and assist you while you complete the lab assignments.

Assignments

Assignments are approximately monthly and they also provide you the opportunity to put the material you've learned into practice. Compared to the labs, assignments will be longer,

require more thought, and you may have to apply the course material in creative ways. Assignments help you solidify your understanding by testing if you can apply the material in a more general setting. A counterintuitive fact is that ***the solution is often the least important part***. The concepts and techniques that you study and practice in the process of completing the assignment are far more important. This means that looking for solutions online is a counterproductive strategy for success.

Most of the learning in CS takes place through your working through problems on your own. In many cases the final answer isn't particularly relevant; it's going through the process of arriving at the final answer that's important. Doing the assignments is the key to doing well in the course. Skipping assignments would be like trying to learn to play a musical instrument without practicing, attempting to master a language without speaking it, or claiming to be an outstanding artist or athlete based solely on theoretical understanding of the underlying principles of the field.

Survival Guide, University of Waterloo

A student once approached me with a concern about their coursework. They said they spent many hours on the assignment and were frustrated that they couldn't solve it. Eventually, they were dismayed to find that the solution was very short. They worried that because it took them so long to come up with something so short that they were missing something. In fact, this experience is totally normal and to be expected—nearly everyone learning this material for the first time has a similar story. When a related question was asked on the final exam (which is worth far more marks than the assignments), guess who answered it correctly: the students that spent hours wrestling with the material or the students that found the answer online?

That's not to say you should spend an unlimited amount of time working on an assignment if you aren't getting anywhere. It's better to spend an hour or two on it and then come back to it tomorrow (or go to the course staff for help) then it is to try to solve it in one sitting. This is one of the reasons why starting early on assignments is a good habit. You're handicapping yourself if you try to solve them in one sitting.

Midterm and Final

There are two tests in the course, a midterm exam that takes place around the 5th week of the course and a final exam. Together they form 80% of your grade in the course. The best way of doing well on the exams is to complete all the labs and assignments by

yourself—and to read, study, and understand the lecture notes in detail. The exams are designed to test an understanding of the course material. “Surface-level” understanding is dangerous, and puts you at risk of failing. Thus, the exams serve two purposes: to measure how well you are doing in the course, and to motivate you to complete (and understand) the labs and assignments.

The labs and assignments are ***critical*** for doing well on the exams. Even though they are worth a small amount of your final grade they are of oversized importance when you consider that doing well on the exams is a result of understanding the labs and assignments.

Grading System

The grading system for this course is not commonly used at the University of Windsor. It is designed to be less “high-stakes” than the typical grading system and to allow labs and assignments to be *learning opportunities* as opposed to ways to earn marks. It is also more interactive and designed to be more like a conversation than a formal test.

Each week you will get a lab assignment to complete that consists of two parts, a practice problem and an assessment problem. You are encouraged to work on the assignment prior to the lab and during the lab and if you need help the lab instructors will show you how to solve the practice problem. For the assessment problem you are to complete the code and test it yourself. If you can do this successfully, you will demonstrate your solution to a lab instructor. They will examine your tests and your implementation to verify that it meets requirements. If it does, they will ask you to explain various details about the implementation and your understanding of the code. For example: *What is the purpose of this line? Why did you write it like that? How else could you have written it?* The purpose of the questions is to assess your understanding and uncover any misconceptions that you have. They will also give you comments and suggestions to improve your submissions in the future.

If you are able to answer the questions to their satisfaction then you will get full marks for that question. If not, they will tell you the part of the code that you need to understand more fully or rewrite. They may tell you to come to their office hours in order to give you more time to work on and understand the question. You should then attend their office hours to revisit the question. Once you’ve demonstrated a complete understanding you will get full marks for that question—if you **attended the lab** and the lab instructor agreed to let you continue the assessment during office hours.

Even if you are not able to complete the assessment, you will still get a 50% participation mark as long as you show up to the lab and make an effort to work on the problems and discuss your work with a lab instructor. If you do not understand how to start, you should discuss with them the difficulties that you are having. The reason we are providing these participation marks is to encourage you to start a conversation with the course staff as soon as possible. The course staff can provide you guidance on the resources available and the steps that you can take to improve your understanding. You need to start these conversations early, as if you cannot complete the lab assessments by yourself it will be impossible to pass the course.

Cheating

Cheating is submitting anything that you didn't write yourself. The majority of students are honest, but in the past we've always had a problem with students trying to pass off code they didn't write themselves, usually either copied from online sources, other students, or AI-powered language models.

The reason why you shouldn't cheat is simple: in the past, almost all of the students that we catch cheating do very poorly on the midterm and final exam and fail the course. This is because the midterm and final exam are designed to test comprehension obtained only by wrestling with the labs and assignments yourself. Submitting an answer from a chatbot is counterproductive, even if the answer is correct, because while you may get the mark for the lab assessment (about 1% of your final grade) you are much more likely to get related questions wrong on the final exam.

Beware the “illusion of knowledge”! It is *very easy* to get the correct answer from an online source and think “ok, that makes sense, now I understand how it works” but when you need to actually apply this knowledge you are unable to do it. This is an all-too-common trap that is easy to fall into. The way out of the trap is to complete the assessments yourself with the assistance of the course staff. Unfortunately, each term there are a number of students that learn this lesson the hard way, but it doesn't have to be you if you commit to not cheating from the start.