HOW TO SUCCEED IN COMPUTER SCIENCE CLASSES

This is an introductory level course in computer science (CS), focused on databases and Structured Query Language (SQL). For some of you, this might be among the first computer science classes you've ever take. For others, you might already have completed several semesters worth of prior coursework, or have considerable on-the-job experience. Regardless of your background, though, it's worthwhile to think about what exactly we'll be doing in this and how you can get the most of out of it.

Like most computer science classes, this one will contain a mix of abstract/theoretical issues with more practical/applied issues:

1. On the more theoretical side, we'll be talking about the definitions and properties of key concepts like **databases, data models, relations (or tables)** and **normalization.**
2. On the more practical side, we'll be focused on learning the programming language **SQL,** and learn to create **entity-relation diagrams**.
3. Many topics, such as database design (a major focus of the class) have aspects of both the theoretical and the practical.

Many students have a preference for either the more theoretical or the more practical end of computer science, and that's perfectly fine! However, it's important to think about *why* both sides matter. Without knowing something about the "theory" behind databases, it can be (practically!) difficult to figure why things have gone wrong, or to fix things that have broken. By contrast, without knowing something about the "practical" end of things, the *reason* that we study certain theoretical concepts (such as those underlying the "relational model" of databases) can easily get lost. Keeping this in mind can help keep you motivated when (sometime in mid-semester) you start asking yourself questions like "Why do I have learn this, anyway?" 😊.

FOCUS ON LEARNING THE MATERIAL, AND NOT (TOO MUCH) ON YOUR GRADE

There is nothing wrong with wanting to get a good grade in this class, and the overwhelming majority of students who follow the advice on this handout will pass the class with *at least* a C (and many will get As and Bs). However, focusing too much on your grade can be an unneeded source of stress, and can get in the way of your actually learning the material (and weirdly enough, of getting a good grade!):

1. On some of the "easier" assignments, you might find it easy to get a "good grade" on the assignments, even without doing the reading or looking closely at the problems. In these cases, I *strongly* encourage you to go back and look over the problems again, and make sure you really understand how the problem "works" and not just what the "right answer" is. Later material in the class will build on the early material, so it's important to focus on really understanding the material, and not on getting through it as fast as possible.
2. On some of the "tougher" assignments, you might run into an occasional problem that you just can't "crack," even after you've spent 30 minutes (or occasionally, 3 hours) staring at it. This is OK (it happens to professional computer scientists all the time, actually…), and having occasional problem will *not* doom you to failing the course. In these cases (after giving it your best shot), I recommend going back over some of the easier problems that you *do* understand, or looking at some problems in the textbook, etc. I always find it best to end a homework session by working on something I "get," even if I know this won't necessarily help my grade.
3. Remember that, for the toughest assignments, there is absolutely nothing wrong with getting a 50%. This score doesn't mean "you've failed to learn anything" but rather "you've already learned a lot, even if you haven't learned everything." The grading system for the class is designed to ensure that, if you diligently do your work, you *can* pass the class.

PRACTICE COMPUTER SCIENCE (ALMOST) EVERY DAY

In general, a three-credit college course should take around nine hours per week (including lecture, reading, and homework). For many students, computer science classes (like math or logic classes) tend to take a bit more time than do other sorts of courses. With this in mind, I'd encourage you to plan out the time you will spend on your computer science homework at the beginning of the week (as opposed to trying to "fit it in" whenever you have time). Some general pointers:

1. **Make CS a priority.** If you've had trouble in "math-y" classes in the past, I'd encourage you to do your homework *before* doing your homework for other classes, cleaning the house, etc. Make sure to do CS homework when you are awake and aware and not when you are tired and grumpy (this really does make a difference!).
2. Getting better at CS requires **frequent practice** (it is somewhat similar to learning to play a musical instrument, or practicing a sport). You'll learn much more (and find it much more enjoyable) if you spend an hour or two every night practicing SQL queries than if you sit down and try to do an entire homework assignment four hours before it is due.
3. **Take a break.** A good study session is around two hours, with a 5-minute breaks every half hour or so where you get up and walk around (don't spend the breaks surfing the internet!). After this, give yourself at 30 minutes to relax before going back to your homework. Trying to write code straight from 9 pm to 3 am is a recipe for disaster.
4. **Plan your study time.** One good way to get your nine hours in: plan on spending two hours on the class Mon-Fri (with a break thrown in), and then take the weekend off.

One benefit of "spreading out" your work on this class is that you give your brain time to process all the new things you'll be learning. In many cases, I've found that problems that seemed *impossible* when I went to bed on Tuesday night all the sudden seem pretty doable by Wednesday morning.

DO THE HOMEWORK! READ THE BOOK! ATTEND CLASS!

I know every teacher you've ever had has told you this, but in this class, it really is necessary to do as much of the homework as you possibly can if you want to succeed in the class. Many of the assignments and lecture "build" on previous ones, so missing one or two assignments can snowball pretty quickly. This does NOT mean you have to get an "A" on every assignment, but it does mean you have to sit down and give it your best shot.

PRACTICE ACTIVE READING (AND/OR LISTENING)

Reading a computer science textbook (or lecture notes) is NOT like reading a novel, or even like reading a textbook for your other classes. If you try to read without taking notes, you probably won't get much out of it (even if you try rereading it multiple times). Instead, you should plan on reading (or listening) *actively,* making sure you really understand what is going on, and asking yourself *questions* about the text (and/or lecture). Here is my general strategy for reading:

1. **Skim** for general format—what topics are dealt with? How much space is devoted to each? (This will often give you a rough idea of what will be covered on the homework.)
2. Look for **key terms and concepts**—what terms/concepts are defined? What are the definitions (I often write these down on a separate piece of paper)? Make sure to pay attention to the *specifics* of the definition, and not just to the "general idea." In technical definitions, *every word* of a definition matters.
3. **Read** the textbook chapter in its entirety, paying special attention to how the sample problems are solved. It's *completely normal* to spend 10 (or sometimes 20) minutes being confused by a sample problem. Just go through it slowly until you start to "get" what is happening.
    a. In general, studies have found that **re-reading** textbook material doesn't increase substantially increase understanding. It's better just to read once (Actively! Slowly!) than to just go over it again and again.
4. If you get confused on a concept, it's OK to look at Wikipedia or StackOverflow (I know I do!). Many times, it can help to see a concept explained several different ways.
5. Try to do a few of the **problems** in the textbook on your own, and see if you can get the right answer (nearly every textbook gives the answers to some of the textbook problems).
6. **Understand. Don't memorize.** In college-level computer science courses, you shouldn't try to memorize "solutions" to problems. Every problem will be a bit different, and you'll have to use your understanding of the concepts to come up with your *own* solution to the problems. Once you understand the concepts, remembering them won't be nearly as difficult.

ASK FOR HELP!

When learning computer science, teamwork plays a big role. Here are some suggestions:

1. When you get confused, ask for help! In particular, attend **office hours,** or go to the **campus learning center** (where they have CS tutors). In many cases, working through problems with an "expert" for 15 or 20 minutes can help you figure out (pretty quickly) what has been going "wrong." A note: It is very difficult to do effective CS tutoring via e-mail. So, I encourage you to meet with me in person or via Zoom.
2. Find a **"study buddy."** While the homework should reflect your own work, there is nothing wrong with getting together with a classmate (or two) to bounce ideas off. Research has consistently shown that one of the best ways of learning new concepts is to (a) listen to your peers' explanations of them (since they are closer to your "level" than the instructor is) and (b) to try to *teach* new things to your peers. Plus, working with other people makes the homework more fun!
3. If something is interfering with your ability to succeed (such as illness, family emergency, etc.), let people (such as your instructor, your academic advisor, etc.) know! We really are there to help!

As a side-benefit, students who make connections with their instructor/classmates/tutors tend to deal better with adversity (when things get tough, they have someone to talk with), and to get better grades overall.

GENERAL STRATEGIES FOR COLLEGE SUCCESS

To close with, here are two general strategies that should be of use to you in *any* college class:

1. **"Fake it until you make it."** Many students have the feeling that they are "bad" at Computer Science (or English, Math, Philosophy, etc.) even when they are doing OK in the course. They think "it shouldn't be so hard for me!" or "I'm getting the right answer, but I don't really understand how!" *This is a completely normal feeling when you are learning something new.* The fact that (certain) problems are tough

doesn't mean you are missing something—they are *supposed* to be tough, and it's *good* to struggle with them. Just keep doing what you are doing, and it will turn out OK.

2. **Work less. Sleep more. Exercise. Eat Healthy.** While we all know these things on some level, it's important to remember that things like work-related stress, poor sleep, a lack of exercise, or unhealthy diets really do make a (big!) difference in college. Students who work 25 hours a week and who sleep 8 hours a night will find it easier to succeed in college than those who work 50 hours a week and sleep 4 hours a night. Obviously, it's impossible for most of us to be perfect in these sorts of things (life gets in the way!), but there is usually *something* we can do to make things a bit easier for ourselves.

REVIEW QUESTIONS AND ACTIVITES

1. Get to know a fellow student in the class by doing the **Five Fingered Introduction.**
    a. **(Index finger)** Your basic info, including (at the very least) your name, major, year in school, and professional goals (and a bit about how this course relates to these goals).
    b. **(Middle finger)** Someone (either real or fictional) you look up to as a role model (especially when it comes to computer science), and an explanation of why this is.
    c. **(Ring finger)** Someone or something that you "love" or care about deeply. (What motivates you in life? In your career?).
    d. **(Pinky)** Are there are any "shortcomings" that you want to improve upon in this class? Which skills, if any, do you want to work on? Do you tend to procrastinate too much? Be too argumentative?
    e. **(Thumb)** What is a special talent, ability, or experience that distinguishes you from other people? Tell us a little bit about this.
2. Take 10 minutes, and answer the following questions:
    a. What are your **learning goals** for this course? That is, what skills/knowledge do you want to gain (besides getting a good grade)?
    b. How are you going to structure your studying for this class? What days/times/locations work best for you?
    c. If something goes wrong (you get stuck on a homework problem, fall behind because of illness, etc.) what is your plan for addressing it?