Policies- Syllabus

🖂 Contact Information

- Instructor: Dr. Bart
- Email: acbart@udel.edu (mailto:acbart@udel.edu)
- Office: Smith 411
- Office hours: Monday 10am-11am and Thursday 9:30am-10:30am
- Full staff information: Course Staff (https://udel.instructure.com/courses/1442292/pages/organizational-course-staff)

≣ Course Description

The focus of the course is "How to think like a Computer Scientist", learning to solve real-world problems by writing programs. Being able to design programs - or at least be able to work with people who design programs - is a valuable skill. To build your own programs you need to know two things: how to use the specific programming language and libraries needed, and the more general skill of how to design a program. You will learn how to model the information in a problem domain, design an algorithm that uses the model to solve the problem, and then dissect and improve that algorithm. The idea is to give you the proper knowledge, intuition, and habits on which you can base a lifetime of learning in computer science. We will examine the ways in which computational ideas have arisen from, and had impact on, many different disciplines. In this first CISC course, we focus on the abstract nature of computation: how Computer Science is relevant and useful even without the details of specific computing hardware. You will find that program design involves a series of smaller skills, including information design, logic design, testing, and documentation.

Course Objectives

- Develop abstract computational data models for representing real-world information
- Choose from, and write programs over primitive data, heterogeneous fixed-size data structures (dictionaries), and homogeneous variable-length data structures (lists).
- Develop test procedures for programs
- Use simple control mechanisms: function composition, conditionals, iteration, and recursion
- Abstract over, and analyze, simple programming patterns (higher-order programming) including simple parameter abstraction, parametric data, and looping patterns like map/filter/reduce.
- Basic sorting & searching algorithms, recognize simple program time/space behavior
- Explain when mutable state is needed (and why we avoid it if possible)

Assessments

Complete descriptions and instructions for completing assessments will be provided when the assignment is made. Dates for assignment of assessment and due dates for assessment completion will be indicated on the course schedule.

Assignment Type	**Contributes**
Mastery Quizzes	10%
Programming Problem Sets	20%
Supplementary Activities	15%
Projects	20%
Midterm 1	10%
Midterm 2	10%
Final Exam	15%
Code Beyond	0%

Specific policies regarding the above:

- **Lab and class attendance is required**, and active participation is required. TAs will take attendance in lab.
- Late work will not be accepted, except in exceptional circumstances (e.g., medical situations).
- Final grades cannot be more than one letter grade higher than your average exam grade.

Final course grades will be assigned based on the following schema:

Letter Grade	**Percentage Points**
A	>=93%
A-	<93% to 90%
B+	<90% to 87%
B	<87% to 83%
B-	<83% to 80%
etc.	

Textbook

There is **no required physical** textbook for this course! Instead, we will periodically have recommended readings from free, online sources.

If you would like a textbook, here are some recommendations:

- "**[How to Think like a Computer Scientist](http://interactivepython.org/runestone/static/thinkcspy/index.html)** (http://interactivepython.org/runestone/static/thinkcspy/index.html)" is online and free.
- "**[Automate the Boring Stuff with Python](https://automatetheboringstuff.com)** (https://automatetheboringstuff.com)" is online and free.
- "**[Coding for Beginners in Easy Steps](https://www.amazon.com/dp/1840786426/ref=cm_sw_r_cp_dp_T1_YcSrzbRQ6Z24N)** (https://www.amazon.com/dp/1840786426/ref=cm_sw_r_cp_dp_T1_YcSrzbRQ6Z24N)" is available on Amazon for fairly cheap.

Computer

You are **required** to bring a functioning laptop with you to every class. Only Mac, Windows, or Linux laptops[1] are allowed. Chrome Books are not allowed. Tablets are only allowed if they run full Windows or Mac OS and you have a suitable physical keyboard. Note that a desktop is not a suitable replacement for a laptop on its own. Students will be expected to have their laptop present in class and lab. This laptop must have a functioning internet connection and be able to connect to Canvas, the autograder, and other online activities.

Clicker

You are required to bring a clicker device. You need to purchase the actual clicker vs. using software from your phone or laptop. You can use the same device in multiple classes. Bring your i>clicker to class to earn participation points on in-class questions. Your i>clicker will need to be registered before the second week of class, in order to participate.

Autograder

This course makes heavy use of auto-graded programming problems, projects, and exams. These tools make it easy to quickly get feedback on programming assignments and can be very helpful for your learning. However, they are not perfect.

Our autograding tools are delivered through https://quiz.cis.udel.edu. You will need to use this website to complete a number of critical assignments including the midterm. You should configure and learn more about the Autograder here: **[Policies- Autograder](https://udel.instructure.com/courses/1442292/pages/policies-autograder)** (https://udel.instructure.com/courses/1442292/pages/policies-autograder)

Collaboration and Cheating Policy

Collaboration and cheating can be tricky to understand in computing. Much of programming involves using and building on other peoples' work. However, when you are starting out, you want to be doing things yourself to learn. Even still, getting support from your peers and the greater computing community is almost necessary. We want to encourage honest collaboration and prevent abuse. For more explicit information about this topic, please consult the following page on **Policies: Cheating and Collaboration (https://udel.instructure.com/courses/1442292/pages/policies-cheating-and-collaboration)** .

- Collaboration is expressly forbidden on Exams, certain in-class assessments, and certain other assignments. When an assignment allows collaboration, it will specifically state this.
- Mastery Quizzes and Programming Problem Sets allow you to work in defined pairs using a process called Paired Programming.
- All other assignments are individual by default, but many will be explicitly identified as collaborative. However, remember that you are ultimately responsible for absorbing the material and successfully completing the exams, so allowing a classmate to complete an assignment for you will not lead to success. Your final grade cannot be more than one letter grade higher than your average exam grade.

Course Structure

This course uses a blended approach, meaning a mixture of online and in-person activities. Every module is composed of lessons and activities. These lessons will usually be short topical videos and coding examples. The activities will include short mastery-based quizzes, a set of programming problems, and a larger scale project. Although you have flexibility in completing these assignments, it is critical that you stay on top of them.

Class time will be spent using active learning. Time spent lecturing will be minimized, because that's usually not a good way to learn. Instead, expect worksheets and think-pair-share activities: you are given a problem, you think about the answer, you talk about your answer with classmates, then we find out the answer as a class. The remaining class time will effectively be a lab where you can get individual help from the instructor, TAs, and your partner.

Regularly throughout the course, there will be two midterms and a final exam. There will be a number of practice final activities and review sessions to prepare you for these assessments. The exams take place at the UD testing center near the eastern edge of campus, and you will need to schedule your test taking session in advance. You can read more about the exams here: **Policies- Exams (https://udel.instructure.com/courses/1442292/pages/policies-exams)**

At the end of every module, there are optional extra credit activities called Code Beyonds. These do not affect the course grade directly, but instead are a way to build up good karma that can potentially positively impact your final grade. However, these assignments are extremely challenging and time-consuming; therefore, you can only complete them after completing all other assignments for a module. For more information, refer to: **Policies- Code Beyonds (https://udel.instructure.com/courses/1442292/pages/policies-code-beyonds)**

Open and Due Dates

All assignments in a module open the friday before the week that the module begins. Most assignments are due 11:59pm Friday at the end of the module, except in certain circumstances (e.g., most projects in the second half of the course are due a week later). Late work is not excepted without a university-mandated excuse (e.g,. medical, religious).

Ideally, modules will open well before their regular open date. However, some components of the course are being redeveloped as the course progresses, so it is impossible to open assignments early. We encourage students to move ahead, and warn you not to fall behind.

Backups

Make sure to keep backups of all your work! Hardware failure or accidentally losing your code is not an allowable excuse to be granted an extension. A service like Google Drive or Dropbox is an easy, no-effort way to keep an online backup of your work, without having to do anything extra beyond keeping your files in a special folder.

★ Errata

[1] Chrome OS and other platforms that do not allow you to install your own desktop software are not allowed. If you're willing to hack Linux or whatever onto your TI-84, then that's fine, but we're not going to be able to provide much tech support. Life is easiest if you stick with Mac or Windows in this course.