# Syllabus

**cs1120: Introduction to Computing**
*Explorations in Language, Logic, and Machines*
University of Virginia, Spring 2016

**Meetings:** Mondays, Wednesdays, and Fridays, 1:00-1:50PM in Mechanical Engineering 205 (MEC 205).

**Course Objective.** The goal of this course is to teach students with no prior experience in computing to think like computer scientists, and to empower students to build useful and delightful software.

The course is designed to enable students to appreciate, use and understand ideas at the core of computer science. This is the first course in the Interdisciplinary Major in Computer Science (BACS) major, but the course covers ideas that will be useful and interesting to students, whether or not they major in computer science.

This course satisfies the prerequisite for cs2110 and cs2220, as well as for the BACS major. It also satisfies the engineering school introductory computing course requirement.

**Expected Background:** This course is open to students with no prior background in computer science or programming.

The only background we assume is:

- Language: reasonable proficiency in reading and writing English
- Math: understanding of whole numbers and addition, subtraction, multiplication, division and exponentiation.
- Logic: familiarity with logical and and or and not.
- Computer Literacy: ability to use email, browse the web, and edit text files.

**Materials:** The coursebook is *Introduction to Computing: Explorations in Language, Logic, and Machines* by David Evans. You can download the book from *http://www.computingbook.org* and print it yourself or order a printed copy. We will be using selected chapters from the coursebook this semester, but not the entire book. In addition to the book, additional readings will be assigned. We will also use other materials, including parts of the Udacity CS101: Introduction to Computer Science (Building a Search Engine) open on-line course.

## Staff

**Coach:** David Evans (evans@virginia.edu). My office is Rice 507.

**Assistant Coach:** Yuchi Tian

**Office Hours:** The course staff is here to help you. Please don't wait until too late to take advantage of the available help or ask for help. For nearly everyone, this course will involve learning new ways of thinking that are dramatically different from those you are accustomed to. It is natural to miss things, and then become lost since new ideas build on previous ones. A few minutes in office hours can enable real understanding and save hours of frustration (as well as provide the teachers with lots of insight about

what is going on in the class); if that isn't a compelling enough reason, my office has a great view of the Blue Ridge and occasionally has stray candy for office hours visitors.

The current office hours schedule (which may be revised based on your registration surveys) is:

- Tuesdays, 3:30-4:30pm (Dave, Rice 507)
- Wednesdays, 4:00-5:30pm (Yuchi, Rice 514)
- Thursdays, 11:00am-noon (Dave, Rice 507)
- Fridays, after class-3:00pm (Yuchi, starting in classroom; moving to Rice 514)

## Communications

**Course Website:** *xplorecs.org*. All course materials will be posted on the course website. This page is updated often and students are expected to visit it regularly (almost every day). All lectures, notes and assignments for the course will be posted on the web site. We encourage students to ask questions about the classes and assignments by posting comments on the course blog, and the course staff will read and respond to questions there. You may also want to subscribe to email notifications for when the site is updated (details provided later), but we will generally not send class-wide emails except in exceptional circumstances.

**Slack:** *https://cs1120.slack.com*. We will use a slack group for "real-time" communication for quick help and discussion. Unlike the course site, which is public and visible to the world, this group will only be visible to people in the class.

**Email:** Feel free to email me (*evans@virginia.edu*) with any personal questions or issues that should not be posted publicly. For general questions or issues that might be relevant for others in the class, it is better to post publicly to the course website or slack group (both so that others may be able to answer your question, and others may be able to see a response that will be useful to them).

**Calendar:** The course calendar is available as a Google calendar. Students are encouraged to incorporate this into your own calendar. If you use another calendar program, you can incorporate this calendar using ical.

## Honor

We believe strongly in the value of a community of trust, and expect all of the students in this class to contribute to strenghtening and enhancing that community. The course will be better for everyone if everyone can assume everyone else is trustworthy. The course staff starts with the assumption that all students at the university deserve to be trusted.

To ensure that expectations are clear to everyone, all students are required to sign the course pledge.

## Topics

Computer science is the study of information processes. Computer scientists study how to describe, predict properties of, and efficiently implement information processes. Students in the course will learn to design and create computer programs, and to understand and reason about how those programs execute.

Most of what we know about describing information processes stems from three simple ideas:

- You can define things in terms of themselves (recursive definitions).
- You can treat procedures and data as one and the same (universality).
- When you give something a name, it becomes more useful (abstraction).

Although these ideas are simple, they have profound implications that it takes many years to fully appreciate.

The main topics of the course include:

**Language.** Be able to identify the primitives, means of combination, and means of abstraction for a language; describe a language using a replacement grammar or recursive transition network; determine the set of the surface forms in a language described by a replacement grammar or recursive transition network; determine what a surface form in a language means if you are given evaluation rules for the language; determine the value of a expression following the rules of evaluation. Understand an interpreter; modify an evaluator to change the meaning of a language; explain why the difference between eagar and lazy evaluation matters.

**Defining and Understanding Procedures.** Be able to define and understand procedures including procedures that take procedures as parameters, procedures that produces procedures as results. Understand recursive definitions and be able to solve problems by defining a recursive procedures and reason about the process produced by evaluating an application of a recursively defined procedure. For most of the class, we will use the programming language Python, and students will be expected to be able to write and analyze Python programs.

**Data, Mutation, and Objects.** Understand how complex data can be constructed from simple structures, and how to define and manipulate recursive data structures. Be able to define, use, and understand procedures that manipulate lists, trees, and dictionaries. Define and understand procedures that use mutation; define procedures that create objects; explain a class hierarchy; define procedures that use inheritance; explain how a method is selected given class definitions.

**Analyzing Problems and Procedures.** Describe problems precisely in terms of their inputs and outputs; express the amount of work a procedure requires using asymptotic operators; describe a problem using $O$, $\Omega$, and $\Theta$; estimate the amount of work a solution to a problem involves; classify problems into complexity classes P and NP; explain convincingly why a problem is in NP; explain what it would mean if someone developed a fast (polynomial time) procedure for an NP-Complete problem.

**Modeling Computation.** Explain how to model computation; understand a finite state machine description and explain what it does; understand a Turing Machine description and explain what it does; show that a computing model is (or is not) capable of modeling any mechanical computation; explain what it means for an axiomatic system to be perfect, incomplete or inconsistent; explain the essence of Gödel's proof; determine if a problem is computable or noncomputable and provide a convincing argument why.

## Assignments, Exams, and Grading

Most of the material in this course is tightly coupled and interdependent: it is not possible to understand later concepts well, without understanding what came before. Hence, assignments and grading will be done using a progressive model more similar to what is done in kids Tae Kwon Do classes (and other martial arts) than what most students are accustomed to in academic classes. This means you need to

satisfactorily complete the "White Belt" level assignments before you can move on to the "Yellow Belt" assignments. There will be target deadlines for each level, but if you do not successfully complete a level by the target deadline, you will be able to re-attempt it (possibly doing alternate assignments and oral exams) later.

Exams for completing each belt will vary in format and structure. Some will involve submitting assignments on paper or electronically; others will include quizzes in writing in class or taken at home; and some will involve oral exams.

The target levels for the course are:

- "White" (Monday, 25 January): informal understanding of what a computer is and what programming a computer involves, ability to write and understand expressions that manipulate numbers and strings.
- "Yellow" (Friday, 5 February): understanding conditional and looping statements, being able to defining and use procedures, and ability to combine procedures to solve problems.
- "Orange" (Monday, 15 February): using and reasoning about structured data including lists, procedures on recursive data structures.
- "Green" (Wednesday, 24 February): designing and manipulating complex data and programs that operate on it.
- "Blue" (Friday, 4 March): analyzing the cost of executions, basic of how computing machines work, being able to solve problems without guidance.
- "Purple/Brown" (Monday, 21 March): ability to organize procedures and state into classes and develop programs using multiple modules and inheritance, understanding of type systems and their tradeoffs.
- "Red" (Monday, 4 April): understanding of computability and ability to construct and reason about reduction proofs, interpreters, language semantics.
- "Black" (Monday, 2 May): deep knowledge of major concepts in computer science; understanding of intractability and its significance; ability to design and build a complex, interactive application.

Your final grade in the course will be based on the highest level you are able to satisfactorily complete. For students who demonstrate commendable effort throughout the semester, achieving the Red Belt level earns an A in the course and Blue Belt earns a B. The +/- modifiers reflect quality of work, contributions to the class, and other aspects.

Spend your energy focusing on what you are learning, instead of worrying about your grade. Although the material we cover is challenging, and the pace may seem overwhelming at times, historically all students who put effort into this class have done fine. Although success requires hard work, all students are expected to get an A in the course. Students who do especially outstanding work in the course will be offered paying summer positions in my research group.

  