Introduction to Computer Science I

Course Overview

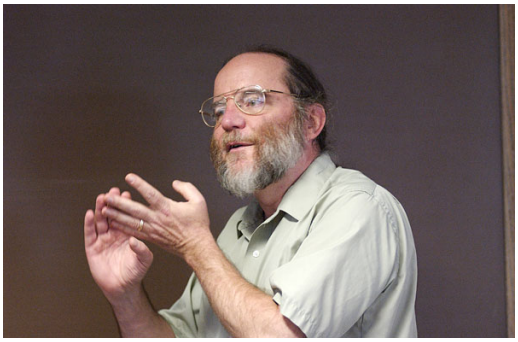
Computer Science 111
Boston University

Vahid Azadeh-Ranjbar

Welcome to CS 111!

Computer science is not so much the science of computers as it is the science of solving problems using computers.

Eric Roberts



Eric S. Roberts is an American computer scientist noted for his contributions to computer science education through textbook authorship and his leadership in computing curriculum development.

Welcome to CS 111!

Computer science is not so much the science of computers as it is the science of solving problems using computers.

Eric Roberts

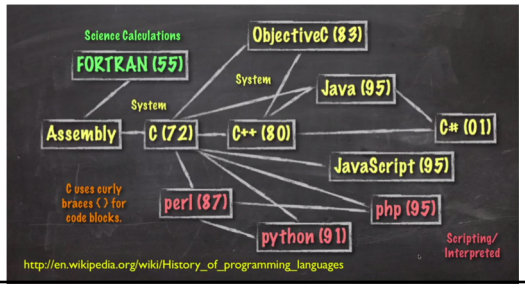
- This course covers:
  - the process of developing algorithms to solve problems
  - the process of developing computer programs to express those algorithms
  - other topics from computer science and its applications

Welcome to CS 111!

Computer science is not so much the science of computers as it is the science of solving problems using computers.

Eric Roberts

- This course covers:
  - the process of developing algorithms to solve problems
  - the process of developing computer programs to express those algorithms
  - other topics from computer science and its applications



Welcome to CS 111!

Computer science is not so much the science of computers as it is the science of solving problems using computers.

Eric Roberts

- This course covers:
  - the process of developing algorithms to solve problems
  - the process of developing computer programs to express those algorithms
  - other topics from computer science and their applications

Computer Science and Programming

- There are many different fields within CS, including:
  - software systems
  - computer architecture
  - networking
  - programming languages, compilers, etc.
  - Theory
  - AI
  - Machin Learning, Deep Learning
  - Data Science
- Experts in many of these fields don't do much programming!
- However, learning to program will help you to develop ways of thinking and solving problems used in all fields of CS.

A Breadth-Based Introduction

- Five major units:
  - weeks 0-4: computational problem solving and "functional" programming
  - weeks 4-7: a look "under the hood" (digital logic, logic circuits, etc.)
  - weeks 7-9: imperative programming
  - weeks 9-12: object-oriented programming
  - weeks 12-end: topics from CS theory such as **AI**
- In addition, short articles on other CS-related topics.
- Main goals:
  - to develop your computational problem-solving skills
    - including, but not limited to, coding skills
  - to give you a sense of the richness of computer science

A *Rigorous* Introduction

- Intended for:
  - CS, math, and physical science concentrators
  - others who want a rigorous introduction
  - no programming background required, but can benefit people with prior background
- Allow for 10-15 hours of work per week
  - start work early!
- Less rigorous alternatives include:
  - CS 101: overview of CS
  - CS 103: the Internet
  - CS 105: databases and data mining (next fall)
  - CS 108: programming with a focus on web apps
  - for more info: http://www.bu.edu/cs/courses/divisional-study-courses

Course Materials

- Free online textbook: *CS for All*
  by Christine Alvarado, Zachary Dodds,
  Geoff Kuenning, and Ran Libeskind-Hadas
  - https://www.cs.hmc.edu/csforallbook/
- In-class software: Top Hat platform
  - bring a phone or laptop to every lecture
  - create your account and purchase a subscription ASAP
    (see Lab 0 for more details)
- Also bring to every lecture:
  - a notebook (loose-leaf or spiral)
  - a pen or pencil

Traditional Lecture Classes

- The instructor summarizes what you need to know.
- Readings are assigned, but may not actually be done!
- Dates back to before the printing press.

- Many technological developments since then!

How do you learn new technology?

Let's say you get a new gaming system….

- Do you read the manual from beginning to end before you open the box?
- Do you try and figure out all you need to know before you begin?

NO!!!

You learn technology as you engage with the technology.

Recognizing this, a new approach to teaching CS was developed.

Lectures in this Class

- Based on an approach called *peer instruction*.
  - developed by Eric Mazur at Harvard
- Basic process:
  1. Question posed (possibly after a short intro)
  2. Solo vote (no discussion yet)
  3. Small-group discussions (in teams of 3)
     - explain your thinking to each other
     - come to a consensus
  4. Group vote
     - each person in the group should enter the same answer
  5. Class-wide discussion
     - why is the correct answer correct?
     - why are the wrong answers wrong?
     - possibly some clarification/explanation by me

Benefits of Peer Instruction

- It promotes active engagement.
- You get immediate feedback about your understanding.

And more importantly

- I get immediate feedback about your understanding!

- It also promotes increased learning.
  - **explaining concepts to others benefits you!**

Drawback of Peer Instruction

- Less time to catch up on sleep.

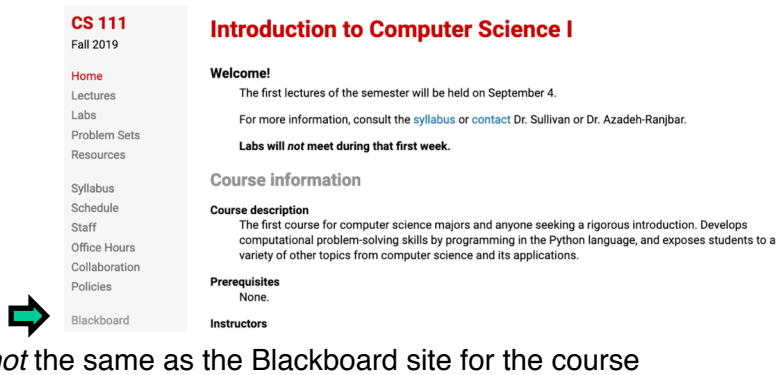
Some people talk in their sleep.
Lecturers talk while *other* people sleep.

– Albert Camus

Preparing for Lecture

- Short reading(s) and/or video(s)
- Short online reading quiz or other exercise
  - complete **by 10 a.m.** of the day of lecture (unless noted otherwise)
  - won't typically be graded for correctness
  - your work should show that you've prepared for lecture
  - **no late submissions accepted**
- Preparing for lecture is essential!
  - gets you ready for the lecture questions and discussions
  - we won't cover everything in lecture

Course Website
www.cs.bu.edu/courses/cs111



- *not* the same as the Blackboard site for the course
- use the Blackboard link to access:
  - the pre-lecture readings/videos
  - the pre-lecture quizzes/exercises

  } posted by 36 hours before lecture
  - the lecture notes – posted after lecture

Labs

- Attendance is required
  - begin next week
- Will help you prepare for and get started on the assignments
- Will also reinforce essential skills
- **ASAP: Complete Lab 0** (on the course website)
  - setup Top Hat account/subscription
  - setup a CS account before your first lab session
  - some other tasks to prepare you for the semester

Assignments

- Weekly problem sets
  - most have two parts:
    - part I due by 11:59 p.m. on Thursday
    - part II due by 11:59 p.m. on Sunday
- Final project (worth 1.5 times an ordinary assignment)
- **Can work on some (but not all!) problems with a partner**
- Can submit up to 24 hours late with a 10% penalty.
- **No submissions accepted after 24 hours.**

Collaboration

- Two types of homework problems:
  - individual-only: must complete on your own
  - pair-optional: can complete alone or with one other student
- For both types of problems:
  - may discuss the main ideas with others
  - may ***not* view** another student/pair's work
  - may ***not* show** your work to another student/pair
  - don't consult solutions from past semesters
  - don't consult solutions in books or online
  - don't post your work where others can view it
- ***At a minimum, students who engage in misconduct will have their final grade reduced by one letter grade.***
  - ***e.g., from a B to a C***

Collaboration

- For pair-optional problems:
  - work with at most one partner per assignment
  - work **together at the same computer**
    - screen should be visible to both of you
    - one person types, while the other plans/critiques
    - switch roles periodically
  - may ***not*** split up the work and complete it separately
  - both submit the same solution and clearly indicate that you worked as a pair.

- After finishing the problems:
  - each person should have contributed equally
  - both could complete the problems on their own

Grading

1. Weekly problem sets + final project (25%)
2. Exams
   - two midterms (25%) – **Wed nights 6:30-7:30**; no makeups!
   - final exam (40%)
     - can replace lowest assignment and lowest midterm
     - **date TBD – wait until you hear the date from me; initial date announced by Registrar will be wrong; make sure you are here for the entire exam period!**
3. Preparation and participation (10%)
   - lecture preparation
   - attendance/participation – full credit if you:
     - make 85% of the votes over the entire semester
     - attend 85% of the labs
   - voting from outside classroom and voting for someone else are **not** allowed

Grading (cont.)

1. Weekly problem sets + final project (25%)
2. Exams: midterms + final exam (65%)
3. Preparation and participation (10%)

- ***To pass the course, you must earn a passing grade on each of these three components.***

Course Staff

- **Instructors:** Dave Sullivan (A1 and C1 lectures)
  Vahid Azadeh-Ranjbar (B1 lecture)
- **Teaching Fellows/Assistants (TF/TAs)**:
  Umang Desai — Louis Jensen
  William Frazier — Allison Mann
  Libby James — Lina Qiu
- **Undergrad Course Assistants (CAs)**:
  Drew Abram '21, Dora Agali '20, Marc Bernstein '20,
  Zach Bodi '22, Noah Cohen '21, Cali Dolfi '21, Amanda Durfee '20,
  Myles Hayes '21, Yongyuan (Steve) Huang '22, Ting Hsu '21,
  Heather Johnson '20, Kristen Lamb '22, Melissa Lin '22,
  Melissa Lopez '21, Nam Pham '21, Rachel Peng '21,
  Harrison Richmond '20, Chenming Shi '22,
  Christopher Trinh '20, Karen Zheng '20
- Office hours: http://www.cs.bu.edu/courses/cs111
- For questions: **post on Piazza** or cs111-staff@cs.bu.edu

Where is everything at BU placed???

Programming

- Programming involves expressing an algorithm in a form that a computer can interpret.
- We will primarily use the Python programming language.
  - one of many possible languages
  - widely used
  - relatively simple to learn

- The key concepts of the course transcend this language.
- You can use any version of Python **3**
  - ***not*** Python 2
  - see Lab 0 for details

Scratch

- Python is a relatively simple language, but...
- To allow for interesting problems right away, we're going to start with something even simpler!
- Scratch – a simple graphical programming language
  - developed at the MIT Media Lab
  - easy enough for middle school kids
  - makes it easy to create animations, games, etc.
- We're using the latest version:

  https://scratch.mit.edu/projects/editor

- Creating a Scratch account is not required for this course.

Your Current Tasks

1. Complete Lab 0 ASAP.
   - on the course website (www.cs.bu.edu/courses/cs111)
2. Review the syllabus and today's lecture notes.
   - available on Blackboard later today