Programming 1

Welcome Packet
COP 3014 Fall 2021

August 23, 2021

Programming I - Course Information

- Instructor: Sharanya Jayaraman

- Teaching Faculty, Computer Science
- Research Interests: High Performance Computing, Numerical Methods, Computer Architecture
- Other Interests: Cinema, Food, Spongebob
- Trivia: Built a Functioning Processor in Minecraft, can name 100 MCU actors.

Teaching Assistant: Silei Song

- Ph.D candidate in Computer Science
- Interested machine learning and data analyze.
- Like cooking and manga. Guitar starter.
- Live to serve Queen Snowy and Duke Cheese (pictured above).

Teaching Assistant: Arunima Mandal

- Graduate student TA
- Currently doing research on Protein Folding.
- Like Competitive programming, music, playing table tennis, badminton and volleyball!!

Teaching Assistant: Alexander Kostandarithes

- First Year Graduate Student
- Interested in Network Security and Artificial Intelligence.
- Fluent in French and major history buff.

Teaching Assistant: Ashwati Nair

- First year PhD Student
- MS from The Ohio State University.
- Interested in data management, data mining and analytics especially Graph data
- I like to watch cooking videos but won't try out much myself.

Teaching Assistant (grader): Aishwareeya "Ash" Rath

- Computer Science Senior
- I am interested in artificial intelligence and machine learning and want to one day work or do research in this area of CS.
- Outside of school, my favorite way to spend time is watching comedy shows.

Teaching Assistants

- Alex Chiciu

Learning Assistant - Emily Schall

- Computer Science BS Senior
- Interested in virtual reality applications, video games, and UX/UI. Had an internship this Summer as an Android dev!
- Has two precious (evil) cats. Avid gamer and enjoys anime.

Learning Assistant: Joshua Kane

- Computer Science second year
- Fascinated with AI and working on deep learning problems like successful small dataset GANs.
- Say hi to me if you ever see me; I love meeting and talking to people!

Learning Assistant: Ebony Bland

- Junior in Mechanical Engineering
- My favorite hobby is cooking. My best dish is lasagna.

Learning Assistant: Grace Brill

- Junior Studying Computer Science
- Undergraduate researcher in the Fuentes Lab (MTRECG).
- Interested in the intersection of sustainability and computing.
- Distance swimmer, triathlete, and avid lifter. Favorite book is "The Price of Salt" by Patricia Highsmith.

Learning Assistant: Franchesca "Fran" Bellevu

- Fourth year Industrial Engineering major
- From Miami, FL
- I am very goofy, understanding, and sarcastic (interesting combo, I know).
- I enjoy photography, painting, and aromatherapy. I love RB music, and I am a math wiz as well!

Learning Assistant: Richard Garcia

- Mechanical Engineering Junior
- I have good knowledge in video games.
- I am interested in programming because of my brother, who will major in Computer Sciences.

Learning Assistant: Romail Khan

- Cyber Criminology Junior
- I enjoy playing video games and I mainly play Valorant, COD, Apex Legends, or anything that is fun to play with friends. I also stream them sometimes. I also like to watch movies, tv shows, and anime.

Learning Assistant: Steven Matiz

- Computer Science Junior
- I love concepts revolving around artificial intelligence and image processing.
- Outside of academia, I love chess, soccer, Netflix or anime, and hanging out with friends (safely, w/ masks).
- I'm also very friendly and approachable, so don't be shy if you have any questions. I sincerely hope I can be of assistant to everyone!

Learning Assistant: Alexandra Velez

- Computer Science Junior
- Interested in robotic optimization research and web developing.
- Aunt to five amazing kids! And an amateur German Shepherd puppy trainer.
- I research in the ORL and interned for Facebook this summer as a SWE!

Learning Assistant: Tyler Welsh

- Computer Science Junior
- My dream job is to be a Software Engineer at a FAANG company (Facebook, Amazon, Apple, Netflix, or Google etc)!
- Outside of the computing discipline I referee semi-pro and collegiate level Soccer. My hometown is Jacksonville, FL and I also love the beach.

Learning Assistant: Rotchy Moricette

- Senior Computer Engineering student
- I am interested in Robotics, and Microprocessors/Microcontrollers.
- I am into movies and video games, my favorite movie is The Great Gatsby, and my favorite video game franchise is the Assassins Creed franchise.
- I also enjoy power lifting, playing football, and cooking.
- My first programming language was Java.

Learning Assistant

- Osairam Mohsin

A Fresh, Semi-Flipped, Student-Centric Approach

- Course structure redesigned to make a more immersive student experience
- Focus on learning and conceptual understanding
- Directing effort to facilitate success by concentrating on the intangibles - Learning How to Learn!
- Spotlight on Resilience and Retention
- We're all here to have fun while learning a practical, profitable skillset!

A Deep Dive into the World of Computer Programming

Course Rationale - Why are we here?

Really knowing how a computer works can be

- Rewarding: You see the result right there. As close to instant gratification as we can get.
- Profitable: The average pay for the 75th percentile of people with a Bachelor's Degree in Programming - $100,000
- Unlimited in Potential: There is no dearth to the number of fields where programming would serve you well.

Topics Covered in the Course

The course is meant to be an introductory course in programming using the C++ programming language. You will learn

- C++ syntax - Input/Output through Structures
- C++ design philosophies - what's under the hood of most applications
- Software Engineering practices and conventions - Why things are done a certain way
- Problem Solving - Design, Techniques and Strategies
- Problem Statement to Maintainable Solution - How to analyze a problem, design a solution, build the software product and test ti to ensure robustness.

Programming is more than a Watch-And-Learn Concept

Throughout the course, we will focus on Software Development Concepts, which rely on experimentation. We learn by doing.

- Incremental Thinking
- Problem Decomposition
- Extrapolation and Code Reuse
- Evaluating “close” solutions and gap analysis
- Build ->Test ->Refactor cycle
- Change Propagation
- Output matching vs stable solutions

There are no “wrong” solutions. Just solutions to another, slightly different problem. File them away, they might be useful later - no knowledge is “useless” knowledge

A Journey of Self-Discovery

But, we also focus on a lot of intangibles

- Conceptual Understanding >Memorization
- Leave behind the standardized testing approach and embrace the practical learning approach
- Focus inward - Discover how you learn
- Learn to Recognize Progress
- Learn how to Explore and Extrapolate
- Learn Resilience and Perseverance
- Recognize the need for assistance and learn how to request it

Excellence is the gradual result of always striving to do better

You need to put in about 10 hours of effort a week - including the lectures and discussion sessions. This translates to about 30 minutes every day outside the classroom.

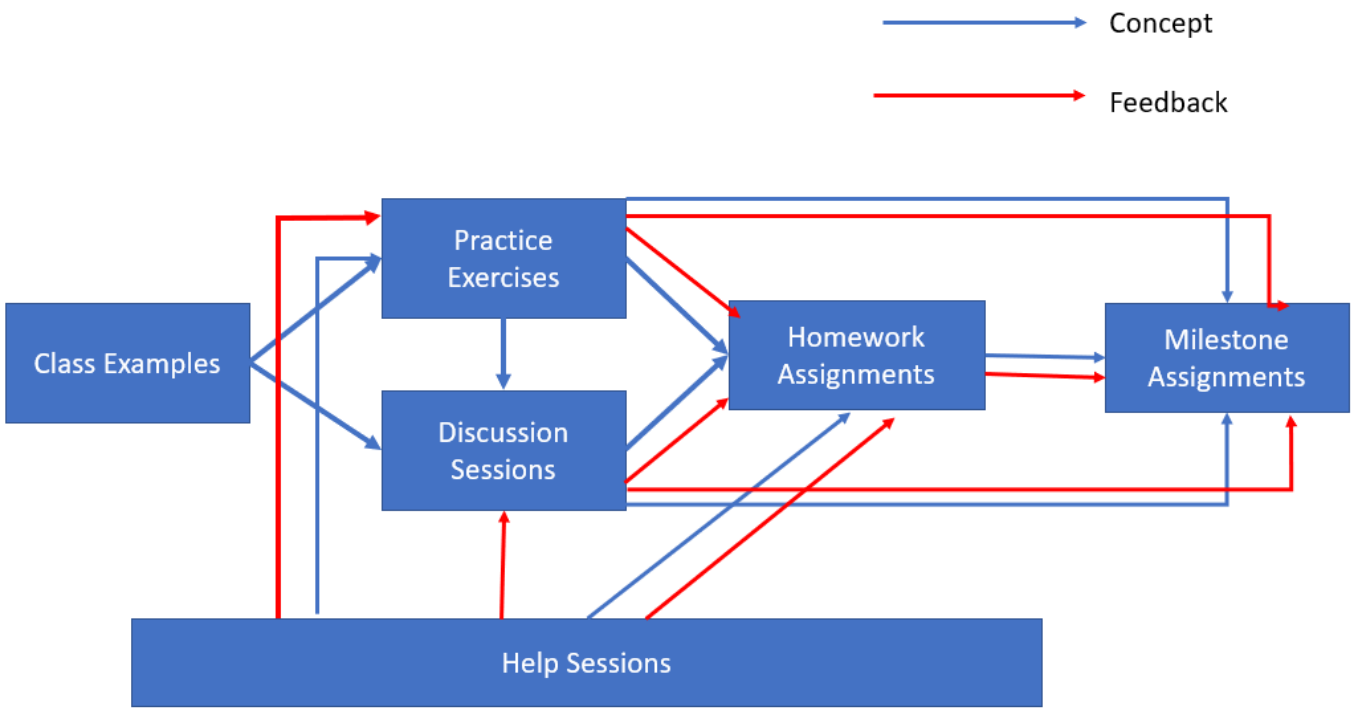
4-tier Scaffolding

We will use a 4-tier approach for the course

- Tier 1: Class example - A problem will be introduced and the solution + solving techniques will be demonstrated in class.
- Tier 2: Practice Exercises - A similar problem or an extension of the problem will be given as a Practice Exercise
- Tier 3: Homework Assignments - An amalgamation of a week or two's worth of problems will be given as a homework assignment
- Tier 4: Milestone Assignments - The concepts will be tested with a smaller but similar problem on the milestone assignments.

4-level feedback loop

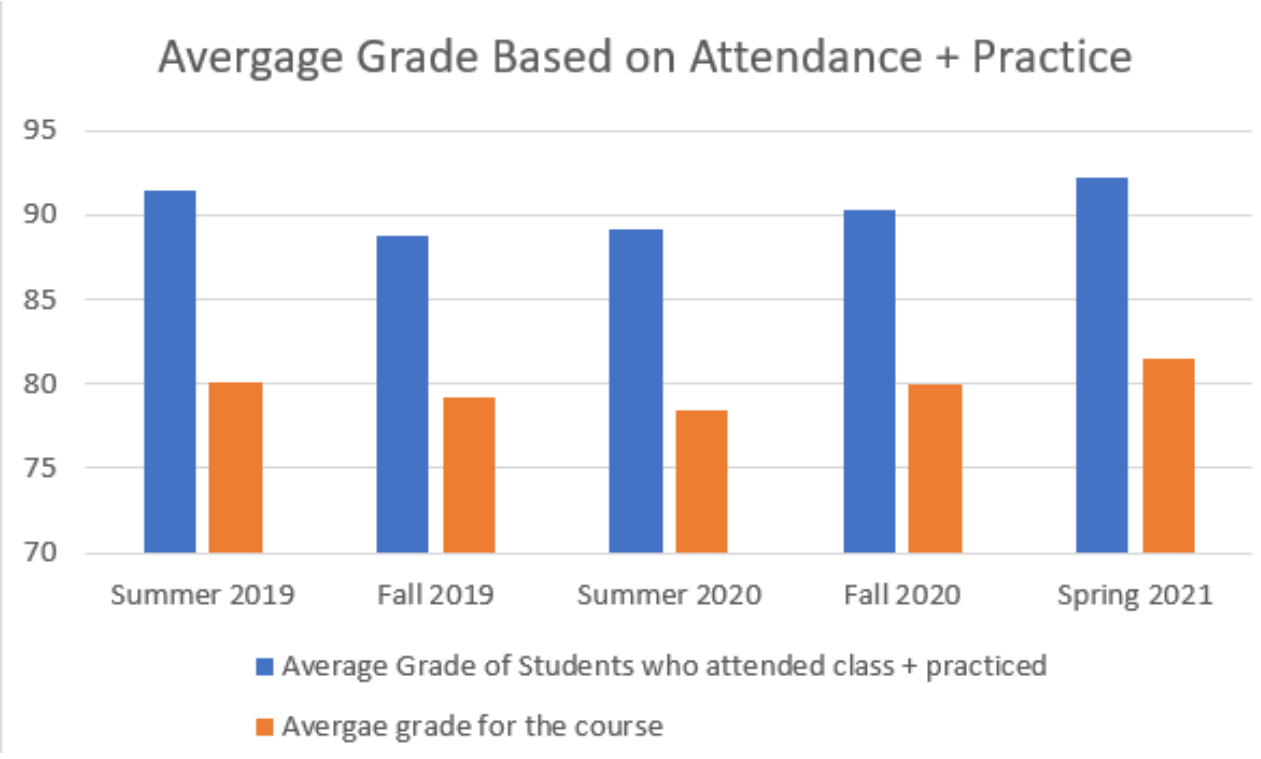
We will offer feedback and help at each of the tiers, resulting in a 4-level feedback loop, which each level offering room for corrections, from a lower to a higher stake assessment.



Course Structure - Mimic the most successful students

The course has been redesigned to offer credit for mimicking the practices of the most successful students in the course.

Students who regularly attend class, participate in discussions and solve the practice problems achieve 1 letter grade higher on average.



Course Structure - Before the Lectures

- Programming is about adapting and extrapolating familiar/existing solutions to solve problems we've never seen before!
- This can be achieved only with a thorough understanding of programming concepts, the capabilities of the language, and the medium.
- To this end, we expect the students to prepare for the lecture, by watching the pre-lecture videos.
- These videos will be posted to Canvas about 24 hours before the next lecture. They are usually about 10 minutes long.
- The videos will cover the theory of the topics for the next lecture and include 2 questions.
- The grade on these questions will count towards your class participation grade.

Course Structure - During the Lecture

- The class is very incremental. So, skipping a few classes will get you into trouble. You are expected to attend class.
- It would be extremely difficult to catch up through binge learning. You would be missing out on several hours of practice.
- Attendance and Class Participation count towards 8% of your grade.
- Every lecture will feature
  - Demonstration of the control structures using various mini real-life analogies
  - Demonstration of underlying programming techniques (simultaneous)
  - In-class exercises, individual and group activities for the attendance and class-participation grade

Course Structure - Practice Exercises

- Students will turn in practice problems after every lecture for credit
- Graded on a 3-tier scale
  - 0 points for no attempt
  - 2 points for an attempt
  - 3 or 4 points depending on the completion and correctness of the attempt
- Practice problems are a combination of
  - Reading Exercises from the textbook
  - Multiple Choice questions
  - Code Reading/Writing/Debugging problems
  - Short Answer questions
  - "Journaling", where the students will use an application to drag and drop lines of code in a control structure, examine the effects and record their observations
- Practice problems are expected to take 20-30 minutes to complete and must be turned in before the next lecture. These count towards 8% of the final course grade.

Course Structure - Discussion Sessions

The once-a-week lab sessions are discussion sessions.

- Students will "present" the solutions to one of the 3 the practice exercises from the previous week to the LA's.
- This is individual work, even if the LA will work with groups of 7-8 students.
- No slides/prepared speech required. Students will explain the solutions they turned in the previous week.
- LA's will discuss the solutions with the students, asking for clarifications, suggesting improvements, strategies, etc.
- LA's will record their observations under the supervision of a TA. The TA would forward the observations to the instructor, who will submit a grade. These count towards 8% of the final course grade.
- The grade for the practice exercises will be released along with the grade for the presentation.

Course Structure - Homework and Milestone Assignments

- The Homework Assignments blend several learning outcomes for a higher stake assessment.
- We expect about 8 homework assignments for the course, once on each major topic.
- Homework Assignments will be handed out every 2 weeks or so, and students will have about a week to 10 days to complete them. They will involve writing about 2 or 3 programs per assignment.
- The 3 Milestone Assignments are timed assessments of learning. They will consist of multiple choice, code reading, code writing, code debugging and short answer questions, meant to mimic an interview.
- Milestone assignments will be on paper, in class.
- Homework Assignment and Milestone Assignments count towards 35% and 39% of the course grade respectively.

Course Structure - Help Sessions

- There is a wide variety of help and support available for the students. We are invested in your success and will assist you in every way possible.
- The instructor, TA's and LA's will hold help sessions in their offices every week. We expect to support about 35 hours a week. 4 of these will be virtual.
- We will help through email. When you send us email, please attach (not copy-paste) the file you are working on. Please also include a subject line to ensure your email doesn't go to spam, and describe the issue you are having.
- You may set up appointments (subject to availability) 24-hours in advance.
- FSU ACE also offers free tutoring for this course.

Testimonials from Former Students

Unlike most other classes where studying will guarantee a good grade, coding takes time and practice in order to excel in your courses. Learning to code is a lot like learning how to ride a bike; you can't expect to succeed unless you put in a little time each day. As long as you continue to practice over time, you will notice yourself continuously getting better!

- Benjamin Zech
Computer Science Junior

Testimonials from Former Students

COP3014 is an essential building block that helps to develop core skills for later classes since the concepts will continuingly build upon each other. I had a lot of fun in this class because I got to practice my previous skills as well as learn new things that challenged me.

- Emily Schall
Computer Science Senior

Testimonials from Former Students

As a software engineer, I utilize everything I learned in COP 3014 on a daily basis. The skills and concepts learned in this class allow me to effectively and efficiently resolve challenges I face across multiple languages, frameworks and other various areas of the practice. What you will learn here, will act as the foundation for your eventual career in software engineering and beyond.

TL;DR ->Pay attention, take notes, start the assignments early, ask questions.

- Nicholas Feanny
Software Engineer, Virtru

Testimonials from Former Students

I am a graduate student who has been at FSU for 8 years, and I still use the knowledge I learned from this course.

You learn the foundations of programming and build upon them as you continue your degree, and they become second nature as you learn more and more. Take this course seriously!

- Timothy Barao
PhD Student, FSU

Testimonials from Former Students

This class is an important first step to demystify the complexities surrounding the technology that we use in our everyday lives. In addition to providing some of the fundamentals of programming, this class will also teach you how to think logically, problem solve, and how to persevere when the answers aren't immediately known. All of these skills and more will extend beyond the scope of the class which means that any student, regardless of major, will find this class worthwhile.

- Jeremiah Cummings
- Software Designer, UF

Testimonials from Former Students

When I started programming, most of my questions were of the form "Can I ____?" where I wanted to know everything I could do. A wise man then responded with "Can you?" Tinker, experiment, and throw code at the wall to see what sticks. Software is your logical ability and reasoning made manifest in a (more or less) tangible form. Introductory programming courses are not just practical puzzle solving, they are a unique chance for introspection.

Programming is a beautifully frustrating art. The computer is very rarely in error; it has done exactly what you told it to do. Rather, your instructions solved a problem which differs from the one in your head. The process of refining your specificity improves your ability to communicate concepts and methodologies to any target, as well as train logical problem solving.

- Preston Hamlin
- Senior Engineer, AMD

Course Expectations

This is a hard class

- While this class is required for several majors as a capstone, it is also an introductory class for CS majors.
- You have signed up to learn programming in C++. At the end of the course, you should be a competent entry-level C++ programmer.

Course Expectations

Reading

- Please read through the entire write-up a couple of times to understand the requirements before asking questions.
- Most of the assignments/ problem statements will be long. Jumping the gun without reading the whole thing could be detrimental.
- Most problems will have a "story" of sorts introducing it. One need not understand the pop-culture references there to understand the problem itself. You would be given Sample Runs that explain program behavior. The story is just to provide you with a way to relate to how this concept could be used.

Course Expectations

Start Early!

- You will be given a week to 10 days for homeworks. Please start early. You need that amount of time to complete them.
- Starting early also gives you time to ask for help if you get stuck.

Course Expectations

Retention

- The class (and the major) is very incremental. Material introduced in one class will be applied through the rest of the course. Retaining material is important. There is no modularization of material.
- Please make sure you understand a concept before we move on to the next. We are ok with repeating material.

Course Expectations

Ask for help!

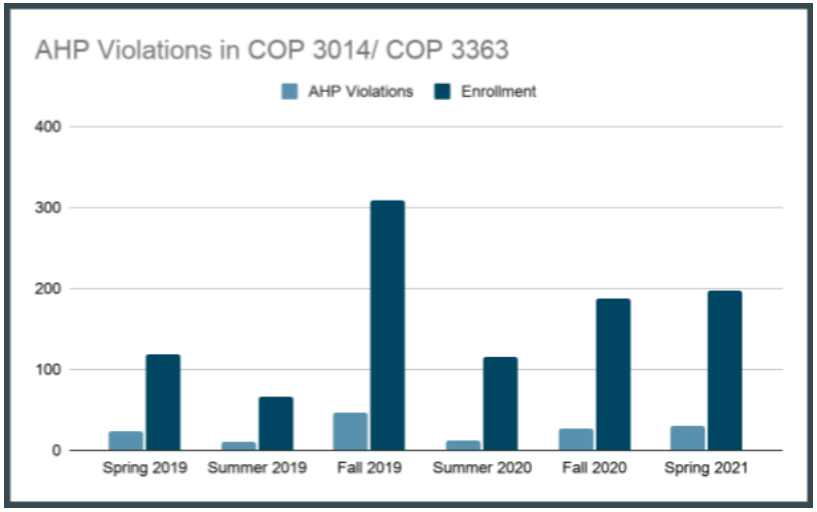
- The instructor and the TA's are available to help. Please do not hesitate to ask for help.
- We are willing to work with you to ensure you are learning the material. However, this requires that you start the assignments early.
- Asking for help is not accepting defeat. It is just asking for help. In the real world, programmers work in teams and bounce ideas off each other. For this class, the instructors and the TA are willing to play that role.

Academic Honor Code

- Of late, we have been having a lot of issues with violations of the Academic Honor Code.
- Since this is your first programming class, the rules might be a bit ambiguous. We will try to clear up any confusion here.
- What is allowed:
  - Asking the instructor, TA, LA or ACE Tutors for help
  - Discussing concepts in general, discussing the concepts used in an assignment.
  - Getting a tutor to help you with concepts and ideas.
- If you have a question about what counts as a violation to the honor code, please ask the instructor, TA's or LA's.

Academic Honor Code

Incidences of Academic Honor Code Violations (and relation to course size)



This indicates several things:

- Students are very often tempted to take the "easy-way-out", out of desperation, lack of time, etc
- It is also frighteningly easy to find (and pursue) violations of the Honor Code and the penalties are severe.

Academic Honor Code

This is a list (inexhaustive) of things that violate the Academic Honor Code

- Copying another person's solution (changing variable names won't help).
- Giving another person your solution.
- Telling another person how to solve the problem.
- Hiring a tutor to solve the problem for you.
- Asking friends/family to solve the problem for you.
- Copying solutions from the internet.
- Hiring people on the Internet (including websites like chegg) to solve the problem for you.
- Turning in older solutions if you're repeating the class.
- Working with another student (collaboration).

Student Expectations

The most common student expectations from the First-Day Quiz

- To be competent in C++ and Computer Programming
- Clear Specifications on assignments and grading
- Detailed feedback
- Availability of the instructional staff

After the First Lecture

We recommend the following steps after the first lecture

- Read the syllabus. It goes over what we just said, more formally.
- Make a commitment to effort - We have made a recommendation of how much effort is required and how to apply it. Now, you can customize it to suit you.
- Mark time for this class on your calendar. Set reminders to make sure you practice.
- Create a Journal for this class on Google Docs/ One Drive/ on you computer/ on paper. Track your progress every day!
- Bookmark the course website, and ensure you know how to contact the instructional staff.
- Think about how you can incorporate the tangible and intangible course content into your everyday life.
- Install the required software.

Software Required for the Class

- The recommended software is JetBrains CLion
  - You can find it at `https://www.jetbrains.com/clion`.
  - You can then Download and Install CLion for free with a student license. Please follow the "Install Software" document on the course website for instructions on installing and using the software.
- You can also use XCode, Visual Studio, etc. However, if you do so, please keep in mind that the TA's will use CLion to grade.
- You can also create an account on the CS department programming servers, and use a text editor and the g++ compiler to run your code.