Quick Facts

What does this course offer?

This class is a rapid introduction to the Python programming language and expands your programming skills acquired in previous courses/experiences. You will learn how professionals solve problems in code in general and specifically in Python. This course presents many experiences to move you from knowing about coding to being a practiced programmer.

I assume that you enter this course with solid programming ability in some language beyond HTML. We will cover 20 weeks of materials in 10 weeks, and you will not likely keep up if you are not prepared with some prior experience. If you are unsure, check out some of the resources on D2L to ensure you are in the correct place.

Success in this class looks different than many others. You will need to learn new ideas and skills, but real success comes in applying those skills in creative ways. The best path to this is your deliberate practice within various skills. You must become resilient to failure and always learn from your mistakes. Programmers make and fix mistakes all day long to claw towards useful solutions. More than anything, if you can build this mindset – learning through failure rather than avoiding it – you will be well-positioned to be a potent computing professional.

Learning aids

The primary textbook is *Introduction to Computing using Python: An Application Development Focus, Second Edition, Ljubomir Perković, John Wiley & Sons, 2015*.

Links to the free library book and for purchase and at the library are on D2L

You may also purchase an account with CodeLab as extra practice and a great way to earn points. The cost is less than many textbooks ($20-25) and well worth the extra practice it offers in Python.

Logistics and Details

The syllabus merely provides an overview of the breadth of information for the course. This class will likely be unlike any you have taken and thus may need more explanation than one document can easily provide. You will find the details of the course on our D2L page, including:

- Introductory info (see Getting Started)
- A general schedule (see the Course Plan)
- "Lecture" content (readings and YouTube)
- Work submission details and scores

Becoming familiar with and frequently engaging with D2L will help you get the most from our meeting session, including:

Live Class – Wednesday 11:50-1:20 CST
Virtual Lab – Monday 11:50-1:20 CST (Discord)

Engaging with the course materials and attending sessions strongly correlates with most students' success!

How will I be graded?

This class has a very different grading approach than traditional courses. Rather than earning a "percentage correct" on a fixed list of assignments, you will choose work that interests and challenges you. Each assignment earns you points, which build up towards your desired grade. You can specialize in topics that interest you and (somewhat) avoid those you don't prefer. Most importantly, you get multiple attempts to learn complex topics and have the entire term to improve your grades.

The Course Plan describes the expected points required for each grade and the minimal points expected to show your mastery of the core

concepts. To earn top grades, you must complete the minimum points in each topic/task.

Behaving like a Professional

The workplace expects professionals to plan and complete their work with minimal oversight. This course will help you start to form those habits and practice owning your learning and productivity by engaging in the following:

- Building and updating a **Course Plan**
- Communicating your progress each week via **Status Reports**

Like most projects, your work in this course can continue to improve until the final delivery. You must complete Course Plans and Status Reports on time, but the final deadline for nearly everything else is the last day of the term. The Course Plan and Gradescope show the 'suggested' timeline for work, but you may need more time. This is OK! The Course Plan helps you to track, and the Status Reports communicate your progress (or delays). These everyday workplace tasks are essential habits to form and earn you points!

Some students struggle without direct supervision and clear punishments. In fact, nearly every student who struggles in my classes does so because of procrastination. If you believe you are a hopeless procrastinator and need help learning how to manage your time, see the Personal Accountability Contract (under Professionalism in the Content section of D2L) for support.

Attendance/Engagement

Each week we will have one in-person and one 'virtual' session (some courses also include an in-person lab with a TA). Generally, you are not required to attend individual sessions, but ideally, you do! You are responsible for watching the recorded announcements at the start of each session you miss. You will be required to attend sessions if you fall behind, or else you may lose the ability to submit past due work or other consequences (see the video on D2L for more details).

Since many courses have a formal asynchronous section (or many students have demanding schedules), I place all 'lecture content' online to start the course. Regardless of your situation, having these materials online lets you work as fast or slow as you wish. If you prefer to attend live classes, I strongly suggest watching these materials before our weekly time, as we will use our class time somewhat differently. Our in-person sessions will focus on demonstrations and activities for the week's topic. The lab time will focus on your questions and collaborating with others. Again, students who attend or at least engage are much more likely to succeed!

Professionals have Integrity

The job of programming fundamentally serves others. As a programmer, you should apply your knowledge and skills to serve your clients best, and as Google says, "do no evil". Integrity is an internal quality rooted in your everyday decisions. The courses I teach are vital to your professional development, so I hope you have little incentive to 'cheat'. Learning this invaluable material is the best way to succeed in the course and your career! I trust +you but still have checks to validate your work.

The most common breach of academic integrity is submitting the work of others as your own (i.e., plagiarism). You can collaborate with others without merely copying their work. I encourage you to work alongside others, so long as you report who you work with and where you find help online (or otherwise). Every coding assignment in the course requires a cover sheet (see D2L for details) where you will report these collaborations.

Most code submissions require a 'ledger' of your work to ensure you are practicing and not merely copying. The ledger is my technology invention that tracks the change to your code. It shows your time spent working on a problem and the ups and downs. The ledger highlights if you succumb to the temptation to copy code as an incentive to just put in the work using any help you need. One of your first assignments will introduce the ledger so remember 'it is watching,' but is there to help you, not trap you!

My Philosophy of Learning

Like many professors, I also do research. My work investigates how people think and learn about computing. My classes are very different as I try to embrace new ideas in my classes. I fundamentally believe that given the dedication, enough time, and the right circumstance, anyone can learn programming. One student described my approach: "you give students enough rope to hang ourselves with; in a good way!". They meant I give you freedom and opportunity and let you make the most of it while trying to help you avoid falling behind.

I want to teach you to learn complex technical topics independently. My most successful colleagues were undaunted by the unknown and embraced new concepts and technologies. Computing moves so quickly that you often must learn tools before anyone has time to document them! Learning how to learn is a critical skill for any fast-paced creative job.

I believe that each student brings unique strengths alongside gaps in their experience. I want you to leverage your strengths and fill those gaps without guilt or pressure. I won't punish you with arbitrary late penalties if you need time to understand a topic better. I won't hold you back from moving quickly or even finishing the course early. I believe I will challenge you in unexpected ways that will give you advantages in your career.

Classroom Management and Policies

Getting Help

Too many students feel they must learn entirely on their own. I encourage and expect you to reach out to any resources that help you learn. Ask peers, former students, family, friends, tutors, and most importantly, the instructional team for help. You must simply document the support you receive (human or online). After leaving school, professionals must hold themselves accountable and find answers through research and collaboration. If you have not started to make that attitude shift, I hope to show you how to do it.

Another shift in mindset I hope to promote is critical to software… mistakes are a part of the game. You will seldom (never!) get your software perfect on the first attempt. Programmers cannot afford to just "do their best by the due date". A better mindset is to keep working. Inch towards daylight but don't flounder too long without seeking help. Learn from your mistakes in any way you need, as that is where true learning comes from.

I hope to be a vital resource to help many of you, not by lecture alone but by working with you one-on-one. Not everyone learns at the same speed or struggles with the same concepts. Individualized learning is so important to me that I have an entire day to promote it! Official class times are not the only time to seek help. You can sign up for virtual office hours using a tool linked on D2L. You can also reach out to me anytime via Discord messages or email. Look at D2L for more info on seeking help.

Unlimited Resubmissions

In programming, failure is always an option. Failure is never final so long as you correct your mistakes. I hope everyone scores perfectly on each assignment (making grading much easier!). First-time perfection is an unrealistic and unfair expectation. The path to finding answers in programming is finding and removing problems (a.k.a. debugging). Good software is not written correctly the first time but gets better gradually over time. You are not only allowed but ***expected*** to correct mistakes and resubmit your work.

Situational Grading

The course plan is our contract for how you earn a grade. If you exceed the requirements stated, then you earn that grade. Those requirements include:

- Exceeding the point total for the grade
- Completing the minimums in each category (on the second tab)
- Completing work with integrity.

You must accomplish all three to get an A in the course. Lower grades indicate you may have some gaps in your knowledge of the subject and thus may have tolerance for missing some of the requirements. For example, you can still earn a B if you miss some minimums. You can miss several minimums and still earn a C, but you must complete some work on every topic.

The point system has checks and balances that can alter your final grade. Since you can collaborate to earn points, you must still show you learned something through quizzes, traditional tests, and your work through your Cover Sheets and ledgers. Remember, getting a job is not about passing tests but convincing an interviewer that you know your stuff!

The best way to show you are doing the work is to stay engaged. Students that regularly submit work, ask questions, and generally communicate with me need little other evidence to show they put in the work and learned. Staying engaged is not only the best way to learn but the best way to be recognized for your effort.

Please commit to a full term of hard work and keeping in touch (perhaps outside your current comfort zone), and you will do fine. I generally give the benefit of the doubt and adjust the final grade thresholds after work is completed. So do your best to earn points (honestly), and you will be rewarded.

Academic Honesty Violations

Your work is checked at several levels for independent submissions. Gradescope includes a tool to compare your work to others. The Cover Sheets and Ledgers tell the story of your work. Merely matching submissions may not be a problem if your collaboration story is faithful. If it is not, then any questionable submissions will be zeroed out (but you can generally resubmit).

A final check of academic honesty happens after the term ends. If your work shows patterns of plagiarism, you will be referred to the universities for an academic honesty review. You can also appeal any zeroed submissions, but this policy requires a referral to the academic honesty review board.

Safety and Respect

One non-negotiable request is to respect the policies, statutes, laws, rules, or mandates that any recognized authority adds to the classroom or virtual environment. Any breaches of these will be reported to the school as appropriate and may lead to your removal from the class if not other University ramifications.