



Course Syllabus  
Object-Oriented Programming for Information Science

**INST 326**  
Fall 2020

## Catalog Description

An introduction to programming, emphasizing understanding and implementation of applications using object-oriented techniques. Topics to be covered include program design and testing as well as implementation of programs.

Prerequisite: Minimum grade of C- in INST126.

## Extended Course Description

This course introduces object-oriented design and programming concepts and methods using the Python programming language. Object-oriented programs are built as collections of “objects”, which are software representations of real-world entities and concepts. Objects combine data (attributes) with functionality (methods), and work through communicating with each other as the code is executed. By encapsulating code complexity within objects, OOP allows use and reuse of existing code in a relatively simple and easy manner. Advanced OOP concepts such as inheritance facilitate development of complex code without sacrificing robustness and possibility of code reuse. We apply computational thinking approaches such as abstraction, decomposition, algorithmic design, generalization, evaluation, and debugging.

This course also provides opportunities to develop an understanding of how programming is situated in and reflects broader social structures, constructs and issues, e.g. race, class or gender. Programming is often viewed as a value-neutral technical skill. However, the social and cultural impacts of information and technology are central concepts in our field, and the growing awareness of issues like algorithmic bias, ethical/unethical uses of algorithms and disparities in opportunities in tech jobs require that any informed professional needs to understand the larger context of programming. This is important to be ethical professionals and to be successful in the workplace. Through readings, discussion and writing, we will critically examine issues of racism, sexism and other forms of power and oppression that are pervasive in programming and related technical activities, and discuss what companies and individuals are doing to improve programming practices and professional work environments.

**Instructor Gabriel Cruz**  
[gcruz12@umd.edu](mailto:gcruz12@umd.edu)

**Class Meets**  
Online - Asynchronously

**Teaching Assistants and AMPs**  
Rishika Saini (Grader)  
[rsaini@umd.edu](mailto:rsaini@umd.edu)

Joshua Cohen (UTA)  
[jcohen43@terpmail.umd.edu](mailto:jcohen43@terpmail.umd.edu)

Melissa Lima (UTA)  
[mlima@terpmail.umd.edu](mailto:mlima@terpmail.umd.edu)

Mason Drenner (AMP)  
[mdrenner@terpmail.umd.edu](mailto:mdrenner@terpmail.umd.edu)

Maxwell Morris (AMP)  
[mmorri24@umd.edu](mailto:mmorri24@umd.edu)

Andrew Ferguson (AMP)  
[afergus1@umd.edu](mailto:afergus1@umd.edu)

**Prerequisites**  
Minimum grade of C- in INST126.

**Course Communication**

- Official course announcements, discussions, and course related questions on ELMS.
- Email extension requests and other issues to Instructor Cruz. Include [INST 126] in the subject line.

*NOTE: I will try to respond promptly during work hours (~8am-6pm). Do not expect timely responses on weeknights and weekends.*

## Student Learning Outcomes

After finishing this course, students will be able to:

1. Explain OOP concepts, principles, and methods;
2. Design, program, and debug Python applications to solve non-trivial problems;
3. Test and assess the quality of object-oriented code;
4. Write clear and effective documentation;

## Office Hours

Due to the nature of the course, all office hours will be held online. All office hours are by appointment only to prevent waits or scheduling conflicts. Additionally, all office hours are limited to 30 minute sessions per student or student group. Students may not have back to back office hours or more than 2 office hour sessions per week with any member of the instructional team unless otherwise approved by the instructor of the course.

## Teaching Notes

This course builds on a basic understanding of procedural programming, so you have to understand data types, variables, loops, conditionals, etc. and how to use them to write and debug a program. If you are fluent in a language such as JavaScript, Java, C#, Visual Basic, etc. you can readily apply your knowledge to learn Python. If you know a bit of Python already, you might find the first part of the course a bit of review. If you are interested in being challenged, I invite you to talk to me about identifying more challenging exercises or developing a more ambitious final project. I want you to learn as much as you can from this course.

Each weekly module will typically follow this pattern, with some exceptions:

### Preparation

- Do assigned readings; watch assigned videos; complete any exercises or quizzes which are due.
- We will use a mixture of lecture, discussion, and lots of hands-on activities to help you apply the materials;
- We will make extensive use of paired and group work.

### Application (homework):

- There will be weekly assignments to help you practice, reflect, and extend your understanding. **All homework assignments are to be completed on your own unless otherwise stated in the assignment.**

Over the course of the semester, we will also examine selected broader issues of programming and coding—the social and organizational context, issues related to gender, race, disability, etc. This will help you prepare for situations that you are likely to encounter in your professional work. These are noted in the schedule as "Critical perspectives".

Here is my suggested general strategy for working on assignments:

- Start early—don't wait. That will give you time to work through the problems and get help as needed.
- When you run into a problem, spend 5-10 minutes trying to solve it on your own.

- Then take a break. Sometimes this will allow you to come back and see something you missed. Letting your subconscious work on it for a while (unsupervised, so to speak) will often lead to useful ideas.
- If you've spent 20-30 minutes and are still stuck, post your question on ELMS. We are here to help each other, so don't beat your head against a brick wall—ask for help! When you post, provide as much information as you can. When helping your fellow students, please do not do their work for them. Help them understand underlying principles and programming techniques and let them arrive at solutions to homework problems on their own.
- I will be monitoring and will respond as soon as I am able, usually within a day (longer during weekends, travel, etc.).
- If you see a question on the discussion board that you can answer, or if you have an idea, please respond. Don't wait for me. You will be helping your colleagues.

## Textbooks & Readings

There is no book required for this course. Instead, we will make use of the following freely available websites/tutorials:

- **Python for Everybody: Exploring Data Using Python 3**
  - Charles R. Severance  
<https://www.py4e.com/book>  
Download: [http://do1.dr-chuck.com/pythonlearn/EN\\_us/pythonlearn.pdf](http://do1.dr-chuck.com/pythonlearn/EN_us/pythonlearn.pdf)
- **Object-Oriented Programming in Python**
  - University of Cape Town  
<http://python-textbok.readthedocs.io/en/1.0/> (Links to an external site.)  
Download: <https://media.readthedocs.org/pdf/python-textbok/1.0/python-textbok.pdf>
- **The Python Tutorial**
  - Guido van Rossum and the Python Software Foundation  
<https://docs.python.org/3/tutorial/> (Links to an external site.)  
Download: <https://docs.python.org/3/download.html> (Links to an external site.) (as part of the official Python documentation)

Other readings (generally available online, or through Library subscriptions) may be assigned as needed. Note that all of these sources can be downloaded for offline access. Please take a moment and download copies now, so that you're prepared in the event of internet issues during the semester.

## Required Technology

- Laptop: we will do live programming exercises during most classes, so bring your laptop and be prepared to write code. Any reasonably current operating system can be used. If you don't have access to a laptop, contact me before the first class.
- Python: Python programming language (3.6 or newer). Python is freely available from <https://www.python.org/downloads/>.
- Editor: Please install Visual Studio Code, which is freely available from <https://code.visualstudio.com/>. Please also install the following extensions (see [this page](#) for instructions on how to install extensions):
  - [Python](#)
  - [Python Indent](#)
  - [File Utils](#)

## Grades

Grades are not given, but earned. Your grade is determined by your performance on the learning assessments in the course and is assigned individually (not curved). If earning a particular grade is important to you, please speak with me at the beginning of the semester so that I can offer some helpful suggestions for achieving your goal.

All assessment scores will be posted on the course ELMS page. If you would like to review any of your grades (including the quizzes), or have questions about how something was scored, please email me to schedule a time for us to meet.

Every graded element of the course (assignment, test, quiz, etc.) is assigned to one of the following weighted categories:

Category	#	Weight Each	Category Weight
Homework	5	6%	30%
Quizzes	2	5%	10%
Final Project	1	30%	30%
Exercises (complete with a partner)	10	1%	10%
Critical Perspectives	2	5%	10%
Comprehension Checks (2 lowest get dropped)	12	10%	10%

Final letter grades are assigned based on the percentage of total assessment points earned. To be fair to everyone I have to establish clear standards and apply them consistently, so please understand that being close to a cutoff is not the same this as making the cut (89.99  $\neq$  90.00). It would be unethical to make exceptions for some and not others.

Final Grade Cutoffs									
+	97.00	+	87.00	+	77.00	+	67.00		
	%		%		%		%		
A	94.00	B	84.00	C	74.00	D	64.00	F	<60.0%
	%		%		%		%		
-	90.00	-	80.00	-	70.00	-	60.00		
	%		%		%		%		

(see last page for sample course schedule)

## Late Work

We are certainly living in difficult and strange times. However I will not accept late work unless I have approved it by prior arrangement. Please note, if you have to miss a deadline, you should inform me as soon as possible, indicating the reason and when you propose to submit your work. If you have a legitimate reason, such as a major medical or family emergency, I may agree to an extension or makeup work, which I will grade by the end of the semester. Documentation of the emergency (e.g., a doctor's letter) may be required.

As a general rule, comprehension checks cannot be made up. I recognize that life events such as illness and family emergencies may arise during the course of the semester; for this reason, the two lowest scores in this category will be dropped.

## University Course Policies

The essential purpose of the university's undergraduate course policies is to enable all of us to fully participate in an equitable, accessible and safe academic environment so that we each can be challenged to learn and contribute most effectively. They address issues such as academic integrity, codes of conduct, discrimination, accessibility, learning accommodations, etc. We are all responsible for following the policies at

<http://www.ugst.umd.edu/courserelatedpolicies.html> (Links to an external site.). You must read them and send me any questions by the first week of classes.

## Academic Integrity and Ethical Use of Other People's Work

In academia and in computer programming, building on the work of others is often acceptable and encouraged. In this class, there will be some situations in which it is appropriate to build on other people's work. For example:

- you may get help from a fellow student to understand a particular concept
- you may pair program with a student on an assignment that has been designated as a pair assignment
- you may want to use a function or an algorithm from a website or a book
- you may be writing a paper and may wish to share ideas you read in a published scholarly work

In this class, the following principles govern the ethical use of other people's work:

- You have an obligation to produce your own original work to satisfy the learning objectives of each assignment. Other people's work should complement, not replace, your own work.
- You should always give credit to individuals whose work you use. In a written document such as a critical perspectives essay, this means providing a complete, accurate entry in your bibliography as well as an in-text citation. In code, you should provide a comment including the following details:
  - the source of the code (URL if online or bibliographic citation if in print)
  - as much authorship information as is available
  - the date you accessed it
  - if applicable, the version number and title of the code

You are expected to complete all course work (homework, quizzes, midterms, reflections, etc.) on your own unless my written instructions on a particular task indicate otherwise. You may not discuss exams or midterms with anyone other than the instructor until the deadline for submitting the exam or midterm has passed for all participants in the discussion (remember, due to personal circumstances, some students may have a different deadline than you). You may work on exercises with one partner unless otherwise specified; both partners must be engaged in the coding process. The partner does not need to be the same partner as in class. You may discuss homework with other students; this includes explaining underlying concepts, assisting a fellow student in debugging (without supplying your own code to that student), and discussing algorithms. If you collaborated with one or more fellow students in one of the ways described above, your code must include a comment describing the collaboration and citing all collaborators. Please note: under no circumstances are you allowed to copy/paste, retype, or work off of, or possess a copy of someone else's solution to an assignment unless the assignment instructions include explicit written instructions to the contrary.

UMD students are required to abide by the student honor pledge: **I pledge on my honor that I have not given or received any unauthorized assistance on this assignment/examination.** You will be asked to complete the honor pledge as part of each assignment, quiz, and test in this class.

Suspected cases of cheating, plagiarism, or other academic integrity violations will be referred to the Honor Council.

## Get Some Help!

Taking personal responsibility for your own learning means acknowledging when your performance does not match your goals and doing something about it. I hope you will come talk to me so that I can help you find the right approach to success in this course, and I encourage you to visit [tutoring.umd.edu](https://tutoring.umd.edu) to learn more about the wide range of campus resources available to you. In particular, everyone can use some help sharpen their communication skills (and improving their grade) by visiting [ter.ps/writing](https://ter.ps/writing) and schedule an appointment with the campus Writing Center. You should also know there are a wide range of resources to support you with whatever you might need (see [go.umd.edu/assistance](https://go.umd.edu/assistance)), and if you just need someone to talk to, visit [counseling.umd.edu](https://counseling.umd.edu) or [one of the many other resources on campus](#).

Most services free because you have already paid for it, and **everyone needs help**... all you have to do is ask for it.

## Basic Needs Security

If you have difficulty affording groceries or accessing sufficient food to eat every day, or lack a safe and stable place to live and believe this may affect your performance in this course, please visit [go.umd.edu/basic-needs](https://go.umd.edu/basic-needs) for information about resources the campus offers you and let me know if I can help in any way.

## Names/Pronouns and Self Identifications

The University of Maryland recognizes the importance of a diverse student body, and we are committed to fostering inclusive and equitable classroom environments. I invite you, if you wish, to tell us how you want to be referred to both in terms of your name and your pronouns (he/him, she/her, they/them, etc.). The pronouns someone indicates are not necessarily indicative of their gender identity. Visit [trans.umd.edu](https://trans.umd.edu) to learn more.

Additionally, how you identify in terms of your gender, race, class, sexuality, religion, and dis/ability, among all aspects of your identity, is your choice whether to disclose (e.g., should it come up in classroom conversation about our experiences and perspectives) and should be self-identified, not presumed or imposed. I will do my best to address and refer to all students accordingly, and I ask you to do the same for all of your fellow Terps.

## Course Schedule (subject to change)

- Mondays:
  - Modules will open up along with all the readings and videos.
  - Readings/lectures are to be done at your own pace but should be completed by Friday.
- Fridays:
  - Assignments with non-code deliverables will be due by 11:59pm.
  - This includes quizzes, critical perspectives and comprehension checks.

- Quizzes will open up on Fridays and will be timed online quizzes consisting of multiple choice, free response questions and short coding challenges. They are timed (1 hour) and must be completed within that time frame.
- Sundays:
  - Assignments with code deliverables will be due Sunday nights at 11:59pm.
  - This includes exercises and homework assignments.

For a more detailed calendar, including any exceptions to those rules please refer to the following:

MODULE		TOPIC	DUE DATE	
Mon	8/31	Module 1	Introduction/Fundamentals 1	Due: Exercise 0
Mon	9/7	Module 2	Fundamentals 2	Due: Exercise 1 and Comprehension Check 1
Mon	9/14	Module 3	Intro to Git and Testing	Due: Exercise 2, HW1 and Comprehension Check 2
Mon	9/21	Module 4	Basics of OOP	Due: Exercise 3 and Comprehension Check 3
Mon	9/28	Module 5	Advanced Data Structures	Due: Exercise 4, Comprehension Check 4 and HW2
Mon	10/5	Module 6	Serialization and File I/O	Due: Quiz 1, Exercise 5 and Comprehension Check 5
Mon	10/12	Module 7	Regular Expressions	Due: Critical Perspective 1, Exercise 6 and Comprehension Check 6
Mon	10/19	Module 8	Remote Collaboration	Due: Exercise 7 and Comprehension Check 7 and HW3
Mon	10/26	Module 9	Advanced OOP	Due: Exercise 8 and Comprehension Check 8
Mon	11/2	Module 10	Databases and SQL	Due: Exercise 9 and Comprehension Check 9 and HW4
Mon	11/9	Module 11	Data on the Web	Due: Quiz2, Exercise 10 and Comprehension Check 10
Mon	11/16	Module 12	Web Scraping	Due: Comprehension Check 11 and HW5
Mon	11/23	Module 13	Data Analysis	
Mon	11/30	Module 13	Data Analysis	Due: Comprehension Check 12 and Critical Perspectives 2
Mon	12/7	Module 14	Bringing it all Together	Due: Final Project Presentation Link
TBD				<b>FINAL PROJECTS DUE</b>

**Note:** This is a tentative schedule, and subject to change as necessary – monitor the course ELMS page for current deadlines. In the unlikely event of a prolonged university closing, or an extended absence from the university, adjustments to the course schedule, deadlines, and assignments will be made based on the duration of the closing and the specific dates missed.