Introduction to Programming for Information Science

INST 126
Fall 2019/0103

Learning Outcomes

This is an introduction to programming for information science majors. This course provides a path for students with diverse backgrounds to successfully learn programming. You are not expected to have any computer programming experience. You will learn how programmers analyze and solve computational problems and develop your own "computational thinking" skills. You will learn basic programming concepts and skills using the Python language (https://python.org/), which is particularly well suited for data collection, analysis and management. This is a hands-on course - you will be writing, analyzing, testing and debugging code, culminating in a project that tackles a challenging real-world problem.

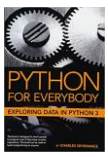
Throughout the course, we will examine issues like algorithmic bias, ethical/unethical uses of algorithms and disparities in opportunities in tech jobs, which you need to understand to be an ethical professional and to be successful in your work. These issues reflect the broader social and cultural context in which we produce software. The social and cultural impacts of information and technology are central concepts in our discipline. Through reading, discussion and writing, we will examine how issues of racism and sexism reflect inequitable forms of power that are pervasive in programming and technical settings, and ways that these issues are being addressed.

After successfully completing this course, students will be able to:

- Explain fundamental programming principles, concepts and methods;
- Develop small-scale computer programs by applying fundamental programming concepts such as variables, data types, assignments, arrays, conditionals, loops, functions, and input/output operations;
- Test and assess the quality of small-scale programs;
- Write clear and effective in-code comments and other documentation;
- Apply computational thinking techniques to analyze problems and develop computational solutions;
- Explain how programming is situated in and reflects broader social and organizational structures, and the ethical and equity issues this entails.

Required Resources

Course website: https://umd.instructure.com/courses/1267936



Python for Everybody ($10 print, free online)
https://www.py4e.com/book

Charles Russell Severance
1st edition (2016). ISBN #1530051126

Dr. Philip Piety
ppiety@umd.edu

Office Hours
Thursday 9-11 or by appointment

Teaching Assistants
Pramod Chundury
pchundur@umd.edu

Office Hours

Prerequisites
MATH115 w > C- or math eligibility of > MATH140. Must not have completed INST326 or CMSC131.

Course Communication
ELMS announcements will be used to send time-sensitive information.

We will use Slack for much of our course-related discussions:

- Post *course-admin-related* questions in #admin,
- Post *course-content-related* questions in #content.

We will try to answer your questions within two business days, so please ask assignment-related questions early.

If you have a private question, contact Pramod on Slack or pchundur@umd.edu. If your question isn't resolved, Dr. Piety on Slack or at ppiety@umd.edu.

Campus Policies

It is our shared responsibility to know and abide by the University of Maryland's policies that relate to all courses, which include topics like:

- Academic integrity
- Student and instructor conduct
- Accessibility and accommodations
- Attendance and excused absences
- Grades and appeals
- Copyright and intellectual property

Please visit www.ugst.umd.edu/courserelatedpolicies.html for the Office of Undergraduate Studies' full list of campus-wide policies and follow up with me if you have questions.

Specific to this course, academic integrity means the following: programmers often use online resources to learn approaches. What will constitute a violation of academic integrity is if an answer or portion of an answer is copied and pasted or substantial portions (i.e., more than two lines) are only lightly edited from another source. We define another source as either another student in a current or past course, material found online (e.g., in videos or forums), or from a tutor hired outside of the class.

If we find a copied answer on an assignment, the student will receive a zero the first time and be reported to the office of student conduct if any further copied answers are found. For exams, students will be immediately referred to the office of student conduct.

Course overview and expectations

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 sub-conscious work on it for a while (unsupervised, so to speak) will often lead to useful ideas.
- If you've spent 20-30 minutes and still are stuck, post your question online. 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. Often it helps to post a screenshot with the problem.
- The TA and I will be monitoring and will respond as soon as we are able, usually within two business days.
- If you see a question on Slack that you can answer, or if you have an idea, please respond. Don't wait for us. You will be helping your colleagues.

Get Some Help!

Taking personal responsibility for you 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 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 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), and if you just need someone to talk to, visit 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 for information about resources the campus offers you and let me know if I can help in any way.

Assessments

1: WORKSHEETS (10% OF FINAL GRADE)

These assessments help you assess your preparation for the concepts to be discussed in class. They will typically be quizzes submitted on ELMS, consisting of a mixture of multiple choice, matching, and short answers. They are designed to be straightforward to do well on as long as you have done the reading.

2: LEARNING CHECKS (12%)

We will use learning checks throughout the posted course material to ensure that you are following along and understanding course content. They should be straightforward to do well on if you are paying attention and have done the reading. The lowest three grades are dropped: consider this a "freebie" for when unexpected life circumstances get in the way.

3: EXERCISES (12%)

Each week, a substantial portion of how you spend your time will be devoted to a hands-on exercise that you will turn in to be graded. These will typically be hands-on programing exercises that apply the concepts and skills discussed in class. Sometimes these include discussion exercises as well. "Correctness" of your programs will be an important component of your grade, but you should have sufficient support to do well on them in class.

4: HOMEWORKS (25%)

Includes coding problems (of course), but will also include analysis questions, brief reflective writing, and other activities. For these assessments, "correctness" of your solutions will be an important part of your grade.

5: EXAMS (25%)

These are diagnostics for you to assess your understanding of the programming basics that are necessary as you proceed through the course and prepare for term project. You will want to address any weaknesses these diagnostics identify to ensure you are well prepared for the project.

There will be 2 exams: the first covers material from Weeks 1-5; the second exam covers material from Weeks 6-10. Each exam will consist of 2 parts: 1) a closed book exam, and 2) coding problems, due within four days of being released.

6: TEAM PROJECT (16%)

The team project will give you an opportunity to apply what you learn in class (especially computational thinking practices and testing/documentation best practices) in a more realistic application than toy problems. You will form a small (~3-5 people) team to **formulate and develop a solution to a real-world problem that interests you** (some samples will be provided to you). The exact problem you choose is open-ended, as long as it conforms to at least one of three application areas that are relevant to information science: data analysis, search, or web services.

There will be **5** components to the team project:

- 1 initial project proposal, where your team submits a document that describes a formulation of your problem in computational terms, along with your team's collaboration plan (proposed roles, time/scheduling expectations).
- 3 progress updates (including current state of the code).
- 1 final project submission.

Assessment on this project will be process-centric: your grade will heavily focus on assessing the extent to which you are effectively using computational thinking techniques, including a computationally formulating problems, using decomposition and abstraction, and practicing good software engineering techniques like quality testing (e.g., testing for edge cases) and effective documentation. It is possible that some of you may choose problems that turn out to be too difficult to solve (although we will work hard to make sure that your chosen problems are scoped well enough that you have a good chance of succeeding): if this happens, don't worry! As long as you are doing a good job of following good *processes*, you should be able to do well on this assessment.

I also recognize that social loafing can be a problem: to incentivize everyone to pull their weight, I will require you to 1) submit a collaboration plan with your initial proposal, and 2) complete (confidential) peer reviews of your teammates. Your grades will be directly impacted by these peer reviews, and they will have a cumulative effect (i.e., will eat into more and more of your grade if your effort is repeatedly rated low by others).

Devices in Class

Based on research and past experience, I do not allow any devices in class, including phones, except when we are actively programming. There is one exception: there will be one row in the back for people who decide or need to use devices. For more information about the science behind the policy watch: http://youtu.be/WwPaw3Fx5Hk

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 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.

Grades

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 exams), or have questions about how something was scored, please email the TA to schedule a time to meet. If you still have a question, you can contact me and we will meet. Any formal grade disputes must be submitted in writing and within one week of receiving the grade. Because of the substantial amount of grading done in programming courses and need to return students their work in a timely manner, **late work will not be accepted for course credit. Please plan to have your work submitted well before the scheduled deadline.**

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 ≠ 90.00). It would be unethical to make exceptions for some and not others. **I do not round grades. I will not respond to email requests for a grade bump at the end of the semester.**

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%		

Course Schedule

Note: This is a tentative schedule, and subject to change as necessary – monitor the course ELMS page for current deadlines.

		TOPIC	MAJOR ASSIGNMENTS
Wk 1	Tues 1/28	Introductions, computers, computation (chapter 1)	**WS 1 due Wednesday 1/29**
	Thurs 1/30	More computational thinking and introduction to Python Download Anaconda and run Hello World in Spyder	
Wk 2	Tues 2/4	Variables, expressions & statements (ch 2)	**WS 2 due Wednesday 2/5**
	Thurs 2/6	Conditional execution (ch 3)	
Wk 3	Tues 2/11	Functions (ch 4) Iteration (ch 5) CritT: Techniques for learning programming and CompT	**WS 3 due Wednesday 2/12**
	Thur 2/13	Functions & Iteration continued In-class exercises	
Wk 4	Tues 2/18	Strings (ch 6)	**HW 1 due Monday 2/17** **WS 4 due Wednesday 2/19**
	Thurs 2/20	In-class exercises	
Wk 5	Tues 2/25	Files (ch 7) Directory hierarchies	**HW 2 due Monday 2/24**
	Thurs 2/27	Exam 1	**Exam 1**
Wk 6	Tues 3/3	Files review Lists (ch 8)	
	Thurs 3/5	In-class exercises	
Wk 7	Tues 3/10	Dictionaries (ch 9) Tuples (ch 10)	**WS 5 due Wednesday 3/11**
	Thurs 3/12	Exam follow-up Mid-semester evaluations Dictionaries (ch 9) (cont)	
Wk 8	Tues 3/17	**Spring break- no classTBD.**	
	Thurs 3/19		
Wk 9	Tues 3/24	Web basics & search engine algorithms	**HW3 due Monday 3/23**
	Thurs 3/26	Search engines continued Project intros	**WS 6a due Thursday**

		Data analysis with Pandas	
Wk 10	Tues 3/31	In-class exercises	**HW4 due Wednesday 4/1**
	Thurs 4/2	Charting & graphing Project kickoff CritT: Search engine bias	
Wk 11	Tues 4/7	Exam 2	**Exam 2**
	Thurs 4/9	Charting & Graphing continued CritT: Google echo chamber	
Wk 12	Tues 4/14	Networked systems (ch 12) CritT: Programming ethics Project work	**Update 1 due Wednesday 4/15**
	Thurs 4/16	Exam follow up Project work	
Wk 13	Tues 4/21	Using web services (ch 13)	**WS 8 due Monday 4/20**
	Thurs 4/23	Project work CritT: Team dynamics	**Update 2 due Thursday**
Wk 14	Tues 4/28	Using web services (continued) Project work	**Update 3 due Tuesday (in-class update)**
	Thurs 4/30	Design ethics Project work	**Update 4 due Thursday**
Wk 15	Tues 5/5	CritT: Limitations of computational thinking Project work: projects + demos	
	Thurs 5/7	Project Demos Course evaluations	
Wk 16	Tues 5/12	Project Demos Course evaluations	
		No final exam - The final project report is used instead of a final exam. All materials are due by May 14th 11:59 pm.	Final project - deliverables & report