Introduction to Programming with Data

Fall 2019

Instructor: Mindy McAdams
Email: mmcadams@jou.ufl.edu

Office hours will be held virtually via Slack as necessary. In order to schedule office hours, please email or message me privately — give at least 24 hours notice if possible and include at least two available meeting times (for your schedule) and appropriate time zone details. (I live Florida.) There will be two required office hours per semester, one of which must be scheduled in the first two weeks of the course; the other is the project check-in during the final four weeks of the course. Scheduling synchronous online time with me more often is encouraged, as this can be a great way to review concepts, ask questions and confirm understanding.

Course Website: http://elearning.ufl.edu

Course Communication: For most communication, the course will revolve around our Slack team, with a mixture of group channels for things like #help and more general chats hosted in channels like #general. If your question is more involved than a simple chat message (i.e., more than three sentences), please use email instead. Be sure to include the course number and a relevant topic in the title of any email you send to me.

Course Description: Introduction to Programming with Data provides a hands-on overview of how to program for data analysis. Using Python, students will learn how to write code for easy collection, analysis, and sharing of data. The course offers an introduction to programming best practices, while quickly getting started with practical data evaluation tasks like tabular reporting and data visualization techniques.

Course Objectives

In this course, students will:

- Create Python scripts to fix a problem, such as how to automatically send emails, how to generate CSV reports and how to scrape a website for data.
- Apply code debugging basics to determine how to fix problems while programming.
- Evaluate data visualization techniques and determine best practices for sharing data visually.
- Analyze data in tabular format using Pandas and NumPy.
- Assemble SQL queries for data extraction from a database.
- Identify statistical methods of data analysis and describe why they are useful and significant.
- Use Jupyter Notebook to share Python skills and data analysis techniques.

- Develop tests to evaluate Python functions.
- Define Python data types and several methods available to each type.
- Execute Python scripts via Jupyter Notebook and the shell or command prompt.
- Examine text data using basic Natural Language Processing techniques such as bag-of-words.
- Evaluate an API for use and execute commands against that API using the Python requests library.

Course Goals

Why is this course important? Learning how to program is an important skill for those who wish to harness data or manage others who do so. Python is a popular language for data science objectives and is easy to learn. Whether or not students plan on using their programming skills at work or even becoming a data scientist, it's essential to understand what is both easy and hard to accomplish using programming if a person is to oversee or interact with technical teams. Finally, being self-sufficient and able to answer their own data questions with some insight will help keep students stay ahead of the curve and somewhat independent when managing projects or fulfilling data-related tasks.

Course Expectations

In order to cover the wide variety of topics included in this course, students will need to apply themselves thoroughly to the coursework and bring a willingness to try new things and a curiosity for data. In this course, students will apply self-guided learning techniques, such as how to debug without an expert by their side and how to use StackOverflow and group chats to solve programming problems. Throughout the course, students will be expected to ask questions and help others who are stuck. These are great skills beyond the scope of programming and will help students succeed in most data analysis tasks they perform or advise in the workplace.

This course requires students to perform a pre-class assessment and have a laptop with an operating system that allows them to install applications and programs (i.e. administrative access). If you are running Windows, you will need Windows 10 or later. If you are running Mac OS X, you will need 10.8 or later (Mountain Lion). If you are running Linux, please insure you can install Python 3.

Ownership Education

As graduate students, you are not passive participants in this course. This class allows you to not only take ownership of your educational experience but to also provide your expertise and knowledge in helping your fellow classmates. In Slack, you should pose questions to your classmates when you have a question as it relates to an assignment or an issue that has come up at work. Your classmates, along with your instructor, will be able to respond to these

questions and provide feedback and help. This open communication and accountability also allows everyone to gain the same knowledge in one location rather than the instructor responding back to just one student which limits the rest of the class from gaining this knowledge.

Required Text: *Data Wrangling with Python,* by Jacqueline Kazil and Katharine Jarmul

Required Installations: You will need to have Python and several other libraries installed on your computer. I will also provide a shared server for some exercises (code assignments); but it is highly recommended you set up your local computer to run all programs for testing, project work and your own use. Installation instructions are in Canvas.

If you run into trouble during any installation, feel free to email—however, I encourage you to first try searching and solving your problem. Becoming more familiar with the inner workings of your computer and how to fix computer problems is a great first step in learning to program and a skill you will hone throughout this course.

Additional Readings: Listed in each weekly module on Canvas

Prerequisite Knowledge and Skills

Students should have intermediate knowledge of how to install and debug programs on their own computers. We will be practicing these skills often throughout the course, so it's okay to be a bit slow at first. I encourage students to *as soon as possible* try walking through the Python and Miniconda installations covered in the Week 1 module in Canvas. This will be good practice for getting to know a bit more about your computer.

As part of the pre-course training, you should step through two of the introduction-to-Python video courses that are linked in the Canvas course.

In addition, if you'd like to learn a little more about the command line, which is very useful for debugging and working better with your computer and computer internals, I recommend taking a look at these resources:

- Windows MS-DOS: https://www.youtube.com/watch?v=MNwErTxfkUA
- Windows PowerShell (Windows 8.0+):
- https://www.youtube.com/watch?v=IHrGresKu2w (I only recommend watching this if you already know DOS)
- Unix-based (Mac, Linux): https://www.codecademy.com/en/courses/learn-the-command-line
- More in-depth Unix-based (bash): https://github.com/Idnan/bash-guide

Teaching Philosophy

As a primarily self-taught developer, I strongly favor practice and project-based learning for computer science. Throughout this course, we will touch upon the deeper theories and academic approaches to data science and computing; however, the course will have a strong emphasis on practical use cases and projects. I believe this allows you to quickly apply and excel at programming to help you do your work; while still allowing for questions, growth and curiosity towards the academic field. This approach will be reflected in our weekly readings, which will blend the research in the field with the daily applications.

Instructional Methods

This course will involve several different instructional methods as a way to address different learning styles and approaches. If you find your learning style is not adequately addressed, please feel free to offer feedback via email at any time. The methods are as follows:

- Video lectures
- Required readings
- Discussion threads, posting and voting
- Asynchronous group discussions (via Slack)
- Coding projects (alone and in small assigned groups)
- Peer and self-assessments and code reviews

Course Policies

Attendance Policy

Due to its online nature, the course will not have in-person meetings or attendance in a classic sense. Students are required to:

- Check Slack for regular updates at least twice a week
- Check the course discussion boards and participate in postings, voting and discussion threads at least once a week
- View and read all required content by the due date
- Send required projects in by their respective due dates
- Respond to group coordination and emails in 24 hours or less

If a student fails to meet the above attendance requirements, there will be a reduction in the participation portion of the student's grade. I encourage students to install necessary applications (such as Slack) on their mobile device or set up alerting to ensure you can promptly respond to fellow students and instructor messages without long delays.

Late Work and Make-up Policy

Deadlines are critical to this class. All work is due on or before the deadline given in Canvas. Assignments are not accepted via email unless requested by the instructor. If an illness or a personal emergency prevents you from completing an assignment on time, advance notice and

written documentation are required. If advance notice is not possible because of a genuine emergency, written documentation will be required. No work for "extra credit" is accepted. Minor inconveniences such as technical issues, family vacation or minor illness are not a genuine emergency.

NOTE: Assignment deadlines in Canvas are usually set for 11 p.m. If you submit after the deadline, your assignment is late. Your inability to upload at the last minute is not a valid excuse for lateness.

<= one hour late	15% penalty
> an hour late, but <= 12 hours late	25% penalty
> 12 hours late, but <= 24 hours late	50% penalty
> 24 hours late, but <= 48 hours late	70% penalty
> 48 hours late	no credit, or 100% penalty

If you have an emergency or pre-approved schedule when you will be unavailable to complete assignments (such as offline or limited access to internet), you must let the instructor know at least five days in advance.

Requirements for class attendance and make-up exams, assignments, and other work in this course are consistent with university policies that can be found in the online catalog at: https://catalog.ufl.edu/ugrad/current/regulations/info/attendance.aspx

Emergency and Extenuating Circumstances Policy

Students who face emergencies, such as a major personal medical issue, a death in the family, serious illness of a family member, or other situations beyond their control should notify their instructors immediately.

Students are also advised to contact the Dean of Students Office if they would like more information on the medical withdrawal or drop process: https://care.dso.ufl.edu/submit-medical-petition/medical-withdrawal/

Students MUST inform their academic adviser before dropping a course, whether for medical or non-medical reasons. Your adviser will assist with notifying professors and go over options for how to proceed with their classes. Your academic advisor is **Natalie Lee**, and she may be reached at natalielee@jou.ufl.edu.

Coursework

Most non-coding coursework will be submitted via Canvas. There are several other services we will use throughout the course for submissions, including:

- Coding projects—GitLab
- Code reviews—GitHub

- Code assignments—Canvas and JupyterHub server

The weekly coursework deadlines and where to submit work will be posted to Canvas and updated as needed throughout the course.

Deadlines: This class, like others, involves many deadlines. Here is a reminder. The new week starts on Mondays.

Group tasks	11 a.m. EST Sunday in the week assigned
Course discussions	11 p.m. EST Thursday in the week assigned
Code assignments	11 p.m. EST Sunday in the week assigned
Final coding project	11 a.m. EST on the last Wednesday of the semester

Grading

Your work will be evaluated according to this distribution. There will be opportunities in some of the assignments for extra credit. Those opportunities will only count for students with regular on-time completion of other assignments (i.e. >= 75% of work turned in on-time and complete).

Reading reactions / chat participation	20%
Assignments	35%
Group contribution	10%
Final project	35%

The final letter grade will be determined as follows:

92–100 points	A	72–77 points	C
90–91 points	A–	70–71 points	C–
88–89 points	B+	68–69 points	D+
82–87 points	B	62–67 points	D
80–81 points	B–	60–61 points	D–
78–79 points	C+	59 points or fewer	E

https://catalog.ufl.edu/UGRD/academic-regulations/grades-grading-policies/

Total scores will be rounded, meaning 91.5 is rounded up to an A (92), and 89.4 (or 89.4999) is rounded down to a B+ (89).

Weekly Lectures

This course will have weekly video lectures shared via Canvas. These videos will be a mixture of content produced by the instructor, as well as other free online videos that explain and demonstrate the course content for the week. I ask that you watch all videos and complete all reading before attempting the assignment content—even if the content is review for you. It's

likely there are some tips and other pointers covered that you will be assessed on at a future point.

In addition to the video lectures, there might be optional live lectures where we will meet in a live setting, such as Google Hangouts. This could be to share a conference or meetup talk I find engaging, or simply to have a group discussion on several topics and a demonstration of a tool via shared conversation. I will have these live events recorded for those that cannot attend in real-time.

It is highly recommended you watch and read each module *in order*. I have arranged the topics so they build upon one another and reference previously covered content. This will help your learning progress intuitively. If you find yourself asking numerous unanswered questions, keep track of them. If they remain unanswered at the end of the module, please post them to the module discussion board or group chat. This helps me know you are watching and following along and allows me and your fellow students to engage and react by sharing our thoughts, answers and related questions.

Assignments: Course Discussion

Each week we will cover a new set of materials with video content, code practice and examples and reading materials. Both I and the weekly moderator (see the following Weekly Moderator section) will pose several questions about the assignments—both questions that can be answered by the content and those that require you to form some opinions and thoughts related to the content. Your responses to *at least two* of these prompts are required.

I expect your response to be well-thought out and presented coherently. Your response to each prompt should be at minimum one paragraph (4–7 sentences) and at maximum three paragraphs. Your initial responses will be due by Thursday (11 p.m. EST) but will be evaluated at the beginning of the following week, giving you time to continue responding if another prompt or comment inspires a more thoughtful or interesting response. Your Communicative & Collaborative section of the rubric requires a response to your classmates; please take time to review and respond to at least one classmate during the module before 11 p.m. EST Sunday evening (of that same module).

Weekly (co-)moderator: The weekly moderator assignment will be rotated so that all students have a chance to minimally co-moderate. The moderator will be evaluated on the questions and prompts chosen to share with the group; and the ability to keep the conversation focused and interesting—encouraging students to keep writing. The moderator will also be in charge of finding one interesting related piece of content (video, article, tweet, GitHub repository, blog post) to share with the class. A list of suggested blogs for useful content and several newsletters are included in the Canvas resources for the course. The weekly moderation will be evaluated according to the moderation rubric and be part of the percentage of the reading reaction grade. In the event there are more than 15 students in class, there may be some modules that have more than one weekly co-moderator.

Discussion Post Rubric

	Exceptional (90-100)	**Proficient (80-89)**	**Basic (70-79)**	**Poor (<70)**
Ideas	Main idea is clear; relevant details support and add value to the conversation; the topic is relevant to the discussion at hand and supports the main idea.	Main idea is clear; the supporting details are present, but some details don't add value; the topic is related, but doesn't add compelling information to the conversation.	Main idea is unclear or unsupported by details. The topic is related, but not relevant.	Main idea is missing as are supporting details. Topic is missing or irrelevant.
Organization	The writing is clear, focused and organized. Details are added in a logical order with attention to transitions between items.	The writing is clear, somewhat focused and shows some organization. Details are added in a supporting order, but transitions might be missing.	The writing is unclear or unfocused. There is basic organization (i.e. paragraphs) but little attention to order of details or presentation of support.	The writing is unclear, unfocused and lacks basic organization. Details may be missing or presented with no attention to order or transitions.
Word Choice	The writing uses insightful words, with technical words and concepts from the module used appropriately and when they add meaning.	The writing uses clear terms and technical words from the reading; several are perhaps misused or added without meaning.	The writing does not include more than one technical word from the module or misuses nearly all when present.	The writing misuses all technical words included, or they are completely missing.
Conventions	The writing follows good conventions for conversational posts, with few if any grammatical or spelling errors.	The writing follows most writing conventions, and would require at least one round of edits and responses to be exceptional.	The reader is distracted by grammatical and spelling errors. Minimal 2 rounds of edits to be exceptional.	The writing is rife with grammatical and spelling errors.
Communicative & Collaborative	The writing promotes conversation and responses. The writer incorporates other responses and ideas, while still communicating their* own ideas.	The writing allows for conversation and responses, however it doesn't explicitly incorporate questions, mentions or others' ideas.	The writing includes one or fewer references to others' responses or references to others are not relevant. The writing is more statement and fact oriented and lacks attention to ongoing conversation.	The writing lacks any references towards others' ideas or responses.

* *they* is used to better generalize for non-cis pronouns

Moderator Rubric

	Exceptional (90-100)	Proficient (80-89)	Basic (70-79)	Poor (<70)
Leadership	The moderator demonstrates leadership qualities such as passion, open-mindedness, authenticity and inspiration in their* prompt and in follow up responses. They are patient and encouraging with responses.	The moderator demonstrates at least one quality of leadership in their* prompt. They encourage at least two responses. response.	The moderator writes a prompt, but it doesn't clearly demonstrate leadership qualities. They encourage at least one	There is no prompt, or the prompt lacks any qualities. The few (if any) responses lack encouragement or thoughtfulness.
Management	The moderator posts at least two prompts in a timely and organized fashion. The follow-up posts and responses show ability to encourage others' participation and a value for others' feedback.	The moderator posts at least one prompt before the deadline (but not earlier). The follow up posts are not encouraging or are too late (i.e. Sunday night before the due date) to promote conversation.	The moderator posts only one prompt. The follow up is either lacking encouragement or too late.	The moderator fails to either post prompts or responses in a timely fashion.

* *they* is used to better generalize for non-cis pronouns

Assignments: Programming Journal

Throughout this course, I ask that you keep a programming journal where you write down things that you learn and questions you have as you explore programming with data for the first time. Your format should follow that outlined in this blog post: https://linbug.github.io/self-improvement/personal%20tracking/imposter%20syndrome/2017/09/30/How-I-hacked-my-imposter-syndrome-using-personal-tracking/

You will document what the problem, question or confusion is. It is okay if you don't always have an answer for what you thought would happen, the exact category, or what you learned, but I would like you to take time researching each issue (spend an hour *or less* each week).

Although you will not be explicitly turning these journal entries in, **you will be using two of these entries at the end of the course** in your final assignment—so having many to choose from is ideal. In addition to their usefulness as points in this course, this process will be helpful

for your growth as a data mining and analysis practitioner. It will show you what you have learned as you look back through earlier entries and give you a standard process for managing the inevitable "Oh no, I don't know this" moments.

Code Assignments

Most assignments will be submitted via the JupyterHub server or GitHub/GitLab. The specifics (including data, problem set or challenge) of each assignment will be available on the Monday of the module week. The due date for each assignment will be Sunday by 11 p.m. EST. Each assignment will have some element of code included as well as some writing (usually documentation, problem solving reflection and / or debugging notes). Sometimes you will be asked to use peer or self-assessment with the same rubric as part of these assignments.

I encourage you to evaluate each piece submitted with the following rubric *before* actually submitting the work. Consider this rubric a *checklist* for items that each assignment will require. Information about PEP-8 and how to easily lint your code for issues are available in the Canvas resources. Note, that several pieces of the rubric (documentation, testing) depend on later modules and will therefore be required after the related module is completed.

Code Assignment Rubric

	Exceptional (90-100)	**Proficient (80-89)**	**Basic (70-79)**	**Poor (<70)**
Clarity, legibility, logic of the code	The code follows PEP-8 standards and is legible. Variable and function names are clear and meaningful. The logic, data types and layout are easy to follow.	The code follows most PEP-8 standards and is somewhat legible. Variable and function names or logic (regarding data types or organization) are sometimes unclear.	The code follows some PEP-8 standards. Variables and functions are often named unclearly. Some logic is applied to data types, but at times it might be unclear.	The code shows no regard for PEP-8, legibility or normal logic flows.
Code or project structure	The code (or repository) has been organized in a manner to allow readability and easy extension. If it is a repository, the folders and files follow common conventions and the structure itself is well-documented	The code (or repository) is readable and at least somewhat extensible (ability to import and use). If a repository, most folders and files follow common conventions and the structure is documented.	The code is readable but would take work to improve for extensibility. If a repository, the folders and files follow some conventions, but not enough to easily share with others. The structure may or may not be documented.	The code is disorganized and not very legible. It might function, but it is not easily shared, readable or extensible. If a repository, little organization exists and common conventions are not followed.

	Exceptional (90-100)	**Proficient (80-89)**	**Basic (70-79)**	**Poor (<70)**
Correct functionality, end result	The code offers exceptional or above-and-beyond comprehension and functionality for the end result. This can be in the form of a "one step further" approach or applying new data to ensure general usability.	The code functions properly and returns the proper result.	The code has at least one error in the result, but also at least one proper step or intermediate result is achieved.	The code is missing or achieves improper or false results.
Library knowledge	The code shows a clear understanding of when and how to apply outside or standard libraries. The author has taken time to learn and apply outside libraries for the problem at hand.	The code uses outside libraries demonstrated in the module appropriately. Only one or two sections of the code could benefit from more library utilization.	The code utilizes at least one outside library, but is not effective or efficient in its use. There are more than two sections of code which could benefit from more library utilization.	The code uses no outside libraries or applies them to irrelevant or improper uses.
Mathematical / statistical reasoning	The code demonstrates a strong grasp of mathematical and statistical logic. The data types and output chosen reflect appropriate mathematical reasoning and can be strongly supported by work in the field.	The code demonstrates a basic grasp of mathematical and statistical logic. Most data types and chosen output reflect mathematical reasoning.	The code shows little grasp of mathematical or statistical logic. The data types chosen show a rudimentary understanding of the mathematical concepts.	The code has few (if any) elements showing any mathematical or statistical reasoning. Data types are misused or rarely used.
Documentation	The documentation is clear, concise and covers relevant topics. There is both module level documentation and class / function level. If necessary, the inline documentation clarifies the logic choices.	The documentation is clear, concise and covers most relevant topics. There is complete module level documentation although not all classes and functions are documented.	The documentation is unclear or too short or long for manageable reading. Several modules, functions or classes lack documentation.	The documentation is incomplete, missing or illegible for at least one if not all module, class, function and inline code sections.
Testing	There are unit or data tests and validation provided that are clear, easy-to-use and documented.	There are unit or data tests and validation provided. Some lack clarity or documentation.	There are missing unit or data tests or they are so unclear and undocumented that they cannot be used.	There are no unit or data tests.

	Exceptional (90-100)	Proficient (80-89)	Basic (70-79)	Poor (<70)
Debugging notes	The author has included appropriate debugging notes in the README either in the form of a narrative (how I solved X) or an FAQ. These notes are ready for public consumption and use.	The author has included some debugging notes in a file. They have at least one outside reference but are difficult to follow.	The author has included only one note or reference re: debugging and it is not included in a separate document or resource.	There are no debugging notes or they are indistinguishable from other notes.

Final Project

The final project will be a group project with group work (in the form of peer review, code review, issue assignment and delegation) being a large part of the tangible grade. Of course, a functional product is also a requirement, and non-functional code will be returned for further work. There will be several stages to this project and reviews along the way, to ensure the groups are working well and progress is steady and obvious (I am hoping this also discourages procrastination). Final projects will be presented and reviewed in the final week of the course and will all be on GitLab. The code and READMEs will be reviewed using the Code Assignment Rubric.

The overall project must meet the defined requirements (posted on Canvas), and the group work will be reviewed using the following Data Science Team Rubric. For the rubric, team members will be evaluated based on the overall team performance (see “As A Team” criteria) as well as individually (see “As a Member” criteria). These evaluations will be based on review of the GitLab project as well as peer and individual reviews.

Teams will be assigned by the instructor after a survey. If you have concerns about your team, a particular team member or your ability to contribute to the team, please contact the instructor immediately. You will be evaluated based on your ability to work together as well as the individual strengths you bring to the table; so I encourage you to attempt to resolve team issues internally before bringing them to the instructor (i.e., as you would for a true data science team if the instructor was instead the CEO/CTO/CIO).

Data Science Team Rubric

	Exceptional (90-100)	Proficient (80-89)	Basic (70-79)	Poor (<70)
As a Team: Collaborative & Communicative	The team communication is collaborative and timely. Issues that are brought to light are quickly addressed. Peer feedback is constructive and positive.	The team communication is collaborative or timely. Issues are eventually addressed. Peer feedback is lacking constructive or positive elements.	The team communication lacks timeliness or collaboration. Peer feedback is missing or overwhelmingly negative.	Team communication is infrequent, incomplete and lacking collaboration.
As a Team: Use of Tools	The team utilizes all tools available within GitLab in appropriate capacity; including Issues, planning, and documentation.	The team uses most of the tools available in an appropriate capacity.	The team uses at least one tool in an appropriate capacity.	The team does not utilize the tools available or uses them in unclear, inappropriate ways.
As a Team: Accepts Challenges & Able to Delegate	When faced with inevitable challenges, the team is adaptive, uses issues to report and inform other members and appropriately and clearly delegates work and review.	When facing challenges, the team is responsive; but may not always utilize issues or communication to handle problems or delegate issues.	When facing challenges, the team is unresponsive or unclear. One or two members end up carrying the group to the finish.	The team has little to no ability to delegate or respond to challenges or does so in an incomplete or untimely fashion.
As a Member: Responsible	The team member takes on tasks and issues and helps delegate what they cannot do well. They ensure the work assigned to them is completed in a timely manner.	The team member responds to delegated or assigned tasks (and might at times volunteer). They ensure their work is completed in time.	The team member responds to some, but not all, delegated or assigned tasks. The work is completed mostly on time.	The team member shows little to no responsibility towards their team or tasks.
As a Member: Responsive	The team member is both timely and open-minded when responding to others' mentions, comments and feedback.	The team member is often timely and open-minded when responding to others; with some visible deviations.	The team member is at least twice not on time or responsive to others feedback, comments or mentions.	The team member is often unresponsive to other members of the team.
As a Member: Productive	The team member commits meaningful and contributive code and comments frequently.	The team member commits and comments at least once a week. Some contributions or comments lack thought or attention to detail.	The team member commits and comments at least every other week. The contributions are mostly helpful.	The team member does not contribute as a productive member via code and / or comments.

Final Presentation + Findings Rubric

	Exceptional (90-100)	**Proficient (80-89)**	**Basic (70-79)**	**Poor (<70)**
Presentation: Organized	The presentation has a clear beginning, middle and end with attention paid to building on previous concepts and findings.	The presentation builds on previous concepts, but has an unclear beginning or end.	The presentation either does not build on concepts or findings, or does so in a disjointed manner.	The presentation pays little attention to organization with little or no building on previous findings and concepts.
Presentation: Audience–Appropriate	The presentation content and language used shows attention and care given the audience. Time is spent on topics engaging to decision makers and peers.	The presentation content and language is appropriate for the given audience. Topics are related to decision makers and peers.	The presentation uses some appropriate language and content, but also includes material too easy or too difficult for the target audience.	The presentation content and language do not fit the audience. The topics are either not present or do not include topics for decision makers or peers.
Presentation: Data Visuals	Data visualization is featured prominently in the presentation and the quality of the visuals improves the content. The visualizations are clear and effective at communicating the findings. The visuals are included in an engaging way and the audience wants to further explore the data based on these visuals.	Data visualization is included in the presentation and the visuals add to the content. The visualizations are effective at communicating the findings. The visuals are included appropriately, but perhaps not in an engaging way.	Data visualization is included in the presentation, but at least one visual is challenging to understand or ineffective at communicating the findings.	Either data visualization is not included in the presentation, or a majority of the visuals are ineffective, inaccurate or unethical.
Findings: Clear and Well-written*	The presentation deck and supplemental final findings materials are well-written and follow convention. The materials make the findings clear and easy to understand. Technical and mathematical words are used appropriately and add meaning.	The presentation deck and supplemental final findings materials are written clearly and follow most conventions. There are a few unclear places or words used improperly, but the overall usage and clarity is not hindered by these mistakes.	The presentation deck and supplemental final findings are lacking clarity or are poorly written. Technical and mathematical words are sometimes misused.	The presentation deck and supplemental final findings materials are incomplete or are unclear or error-prone.

	Exceptional (90-100)	Proficient (80-89)	Basic (70-79)	Poor (<70)
Findings: Statistically Accurate & Reproducible	The findings and visualizations included in the material and slide deck are accurate and easily reproduced. They show an understanding of the statistical principles and are well chosen to convey the outcome.	The findings and visualizations included in the material and slide deck are accurate and able to be reproduced.	The findings and visualizations included in the material and slide deck have minor inaccuracies or are not able to be reproduced.	The statistics and visuals are either missing or are have major inaccuracies.
Findings: Actionable	The findings have a clear actionable result, even if this result is in the form of a new design plan. The importance and relevance of the findings and the topic as a whole is made clear to the audience.	The findings have a clear and actionable result, but perhaps lacking a few areas of follow through or plan. The importance of the findings is referenced, but perhaps not made clear to the audience.	The findings have either an unclear result or little engagement in "what happens next." The importance of the findings is either missing or unclear.	There is little or no attention paid to what to do with the findings. The importance of the findings is lacking clarity or is entirely missing.

* What does "well-written" mean? See the Discussion Post Rubric: Organization, Word Choice, and Convention.

Any clarification needed on the rubrics should be done before the end of the first week of class. If you have questions or suggestions or need help, please message the instructor or post in group chat to clarify the problem or question immediately.

University Policy on Accommodating Students with Disabilities

Students requesting accommodation for disabilities must first register with the Dean of Students Office (https://disability.ufl.edu/), which will provide documentation to the student. The student must provide this documentation to the instructor when requesting accommodation. You must submit this documentation prior to submitting assignments. *Accommodations are not retroactive*; therefore, students should contact the office as soon as possible in the term for which they are seeking accommodations. Students with disabilities who may need accommodations in this class are encouraged to notify the instructor and contact the Disability Resource Center so that reasonable accommodations may be implemented.

Stress and Other Personal Issues

Sometimes, happenings outside of the classroom can affect our performances within it, including virtual ones. Please do not hesitate to take advantages of university resources in this area, for whatever reason. If it is affecting you, it is important to get the support you need, which includes some online services:

UF Counseling and Wellness Center
352-392-1575
https://counseling.ufl.edu/

Netiquette: Communication Courtesy

All members of the class are expected to follow rules of common courtesy in all email messages, threaded discussions and chats.
http://teach.ufl.edu/wp-content/uploads/2012/08/NetiquetteGuideforOnlineCourses.pdf

Class Demeanor

Mastery in this class requires preparation, passion, and professionalism. Students are expected, within the requirements allowed by university policy, to attend class, be on time, and meet all deadlines. Work assigned in advance of class should be completed as directed. Full participation in online and live discussions, group projects, and small group activities is expected.

Feedback

My role as instructor is to identify critical issues related to the course, direct you to and teach relevant information, assign appropriate learning activities, create opportunities for assessing your performance, and communicate the outcomes of such assessments in a timely, informative, and professional way. Feedback is essential for you to have confidence that you have mastered the material and for me to determine that you are meeting all course requirements.

At all times it is expected that you will welcome and respond professionally to assessment feedback, that you will treat your fellow students and me with respect, and that you will contribute to the success of the class as best as you can.

Getting Help

For issues with technical difficulties for eLearning in Canvas, please contact the UF Help Desk:

- Student Help FAQs: https://elearning.ufl.edu/student-help-faqs/
- File a support ticket: https://itsm.helpdesk.ufl.edu/sc/teaching-and-learning/learning-management-systems/course-management-systems
- UF Help Desk: http://helpdesk.ufl.edu/

IMPORTANT NOTE: Any requests for make-ups due to technical issues MUST be accompanied by the ticket number received from LSS when the problem was reported to them. The ticket number will document the time and date of the problem. You MUST e-mail your instructor within 24 hours of the technical difficulty if you wish to request a make-up.

Other resources are available at: https://distance.ufl.edu/getting-help/

- Counseling and wellness resources
- Disability resources
- Resources for handling student concerns and complaints
- Library Help Desk support

Should you have any complaints with your experience in this course please visit:
https://distance.ufl.edu/student-complaint-process/

Course Evaluations

Students are expected to provide feedback on the quality of instruction in this course based on 10 criteria. These evaluations are conducted online:
https://evaluations.ufl.edu

Evaluations are typically open during the last two or three weeks of the semester, but students will be given specific times when they are open. Summary results of these assessments are available to students at: https://evaluations.ufl.edu/results

University Policy on Academic Misconduct

Academic honesty and integrity are fundamental values of the University community. Students should be sure that they understand the UF Student Honor Code at:
https://sccr.dso.ufl.edu/policies/student-honor-code-student-conduct-code/

Academic Honesty

All graduate students in the College of Journalism and Communications are expected to conduct themselves with the highest degree of integrity. It is the students' responsibility to ensure that they know and understand the requirements of every assignment. At a minimum, this includes avoiding the following:

Plagiarism: Plagiarism occurs when an individual presents the ideas or expressions of another as his or her own. Students must always credit others' ideas with accurate citations and must use quotation marks and citations when presenting the words of others. A thorough understanding of plagiarism is a precondition for admittance to graduate studies in the college.

Cheating: Cheating occurs when a student circumvents or ignores the rules that govern an academic assignment such as an exam or class paper. It can include using notes, in physical or electronic form, in an exam, submitting the work of another as one's own, or reusing a paper a student has composed for one class in another class. If a student is not sure about the rules that govern an assignment, it is the student's responsibility to ask for clarification from his instructor.

Misrepresenting research data: The integrity of data in mass communication research is a paramount issue for advancing knowledge and the credibility of our professions. For this reason any intentional misrepresentation of data, or misrepresentation of the conditions or circumstances of data collection, is considered a violation of academic integrity. Misrepresenting data is a clear violation of the rules and requirements of academic integrity and honesty.

Any violation of the above stated conditions is grounds for immediate dismissal from the program and will result in revocation of the degree if the degree previously has been awarded.

Although it should not need to be said, I will state that any student failing to abide by the Code of Conduct towards any other student or faculty member will be immediately dismissed from the course communications and placed in mediation.

If you have additional questions, please refer to the Online Graduate Program Student Handbook you received when you were admitted into the Program.

Schedule

Week One: Course Introduction, Aug. 20–25

- Use the Python interpreter and Jupyter notebook to write and run Python code
- Select packages and install them via pip
- Learn basic programming terms and Python syntax

Required readings: See the module in Canvas
Assignments:

- Install software as directed in Canvas
- Module 1 discussion
- Code Assignment: Getting to Know JupyterHub

Week Two: Introduction to Programming, Aug. 26–Sept. 1

- Determine how to use new functions, classes and methods based on documentation
- Implement logic-based changes in programming using Python

- List different Python data types and differentiate between them

Required readings: See the module in Canvas—you'll need the required book now

Data Wrangling with Python, Chapter 1 and 2*

Assignments:

- Module 2 discussion
- Code Assignment: Data Types in Python

Week Three: Introduction to Statistics, Sept. 2–8

- Select appropriate statistical measures for analyzing a chosen problem set
- Define common statistical terms and match them with their appropriate equations
- Assemble code to answer a series of mathematical and statistical questions

Required readings: See the module in Canvas

Assignments:

- Group questionnaire
- Module 3 discussion
- Code Assignment: Using the Stats and Math Libraries in Python

Week Four: Debugging Skills, Sept. 9–15

- Demonstrate debugging basics by fixing broken code
- Effectively manage Python exceptions
- Research and explain issues found in broken code

Required readings: See the module in Canvas

Assignments:

- Group: Team communication guidelines and roles
- Module 4 discussion
- Code Assignment: Fixing Broken Code

Week Five: SQL and Databases, Sept. 16–22

- Create your own SQL database, and store and select data from the database
- Write proper SQL syntax for selections, updates and insertion
- Describe database schema
- Differentiate between different database types

Required readings: See the module in Canvas

Assignments:

- Group: Initial final project questions
- SQLZoo tutorials
- Module 5 discussion
- Code Assignment: Querying Customers with Python

Week Six: Pandas and NumPy, Sept. 23–29

- Identify, list and classify different NumPy data types
- Generate a pandas dataframe from given data
- Evaluate several statistical criteria from a Pandas dataframe

- Visualize data using Pandas

Required readings: See the module in Canvas
Assignments:

- Group: Final project design documentation & group discussion
- Module 6 discussion
- Code Assignment: Importing, Investigating and Grouping Dataframes

Week Seven: Data Visualization, Sept. 30–Oct. 6

- Create data visualizations using Pandas and Matplotlib
- Identify qualities of a good data visualization
- Design interactive charts using the Bokeh library
- Determine appropriate visualizations for different data types and distributions

Required readings: See the module in Canvas
Assignments:

- Group: Set up GitLab / KanBan
- Module 7 discussion
- Code Assignment: Charts with Matplotlib and Bokeh

Week Eight: Writing Your First Stand-Alone Script, Oct. 7–13

- Create a stand-alone Python script with proper documentation and command line use
- Design documentation and evaluate best practices for documenting code
- Reflect on code design principles via peer and self-evaluation

Required readings: See the module in Canvas
Assignments:

- Group: Initial review of KanBan tasks & adding assignments
- Group: Identify data sources and begin data extraction
- Module 8 discussion
- Code Assignment: Building Your First Python Script

Week Nine: Scraping Data, Oct. 14–20

- Create web scrapers with Python using BeautifulSoup and Scrapy
- Classify different scrapers according to the type of scraping performed
- Use browser tools to evaluate the ease and difficulty of scraping a website

Required readings: See the module in Canvas
Assignments:

- Group: Data exploration & visualization
- Instructor check-in: How did data exploration go?
- Module 9 discussion
- Code Assignment: Scraping Blog Posts

Week Ten: APIs, Oct. 21–27

- Assess API documentation to evaluate useful methods
- Utilize helpful library wrappers for the Twitter API to run simple queries

Required readings: See the module in Canvas
Assignments:

- Group: Initial group project documentation and file structure
- Group: Begin final presentation documentation and slides
- Module 10 discussion
- Code Assignment: Geocode API Usage

Week Eleven: Natural Language Processing, Oct. 28–Nov. 3

- Create code to preprocess text by removing stop words, resolving case and punctuation differences, and tokenizing the content
- Define important natural language processing terms such as POS tagging, tokenization, bag-of-words, tf-idf (TFIDF) and sentiment analysis, and determine good applications for each
- Evaluate a series of documents using Doc2Vec

Required readings: See the module in Canvas
Assignments:

- Group assignment: v0.1 release
- Module 11 discussion
- Code Assignment: NLP with Python

Week Twelve: Tests, Nov. 4–10

- Create unit tests to find bugs, identify faulty logic, and create secure code
- Write data validation tests to find incorrect data
- Utilize property-based testing to evaluate constraints of a given function or method

Required readings: See the module in Canvas
Assignments:

- Group project: Add testing and documentation, and finalize code structure
- Module 12 discussion
- Code Assignment: Adding Tests to Your Script

Week Thirteen: Automation Nov. 11–17

- Create new cron tasks and run them on a server
- Write a Celery task with proper asynchronous execution
- Categorize tasks as easy or difficult to automate based on criteria and questions

Required readings: See the module in Canvas
Assignments:

- Group project: Instructor feedback / code review
- Module 13 discussion
- Code Assignment: Automation Conventions

Week Fourteen: Expanding your Skills Nov. 18–24

- Use Python classes to incorporate object-oriented programming basics
- Distinguish between different types of loops in Python

- Define concurrency as a programming concept
- Develop a self-learning plan for programming

Required readings: See the module in Canvas
Assignments:

- Group: Continue with final feedback; update Kanban for final week of project work
- Module 14 discussion
- Code Assignment: Iterators, Concurrency and Self-Directed Learning

Week Fifteen: Final Project "Last Sprint" Nov. 25–Dec. 1

- Finalize project roles, tasks and responsibilities
- Plan these final steps using the Kanban board

Required readings: See the module in Canvas
Assignments:

- Group: Final Kanban board / v0.2 release
- Continued final project work
- Discussion: Retrospective summary
- Code Assignment: Code Review and Improvements

Week Sixteen: Final Project and What's Next? Dec. 2–4

- Share and analyze what you have learned throughout the course and in the production of your final project
- Evaluate others learning via peer evaluation
- Judge personal areas of interest for future study

Required readings: See the module in Canvas
Assignments:

- Final group evaluations
- Group project presentation and review
- Module 16 discussion

Disclaimer

This syllabus represents my current plans and objectives. As we go through the semester, those plans may need to change to enhance the class learning opportunity. Such changes, communicated clearly, are not unusual and should be expected.