Introduction to Python

(Winter 2026)

Term project (50% of course total)
Due date: Tuesday March 3rd, 2026 @ 11:59pm.

Description:

The final term project serves four purposes: 1) as a way for the course instructor to evaluate your knowledge of the course topics; 2) as an opportunity to develop code that can help your personal research projects; 3) as an opportunity to develop your report-writing and communication skills and 4) as a platform for you to receive constructive feedback on your coding journey and goals.

You may already have a project in mind where you need to import, format, and analyse your own or someone else's data. You can use this project to produce a coding script or pipeline that will evaluate that data. Alternatively, you may be expecting a specific kind of data from your experiments, and you can generate mock data that will undergo the same kind of analyses you'd like to perform. Furthermore, you may need to explore different kinds of data analyses – perhaps there is a specific package you'd like to learn how to use.

Aim for your target audience to have low to beginning-level knowledge in Python. Often your collaborators, PI, or others reading your future data analyses, may not share your possibly advanced level of coding skill. When working with packages outside the purview of this course, explain your choices, and give a brief breakdown of what these packages or functions do. Communicate your intent and steps clearly!

All these directions can provide the right kind of material for a final term project. Take the time to think about what your needs are so that you can produce something useful for yourself or your future self! Contact the instructor well ahead of the due date if you are unsure about your topic or need inspiration for one.

Formatting and files:

Throughout the course, we will be working with the Jupyter Notebook format. You will see the use of both markdown (formatted text) and coding cells to convey the concepts we learn each week. Markdown cells can help explain steps prior to coding and comment on output produced by your code afterwards. Coding cells run your code and accomplish the tasks you outline in your background. The final term project should take advantage of these techniques to produce a format somewhat like the publication of a manuscript or online tutorial.

You'll want a background *including* a brief outline of what you'll be doing and the coding methods/paradigms you will employ to do it. Your "results" section is the bulk of your coding steps where you can show off a bit of everything you've learned in class. The discussion section wraps up any lingering outputs, future directions, or thoughts on where you might improve upon your code. You can even discuss what you wanted to accomplish but weren't able to and why! Finish up with any references from your background or discussion sections.

When submitting your final project please include, as an archive (zip file):

1) Your Jupyter notebook (ipynb)
2) A PDF version of your notebook (used to add comments for you)
3) Any data files/images required to run your code – please keep data files under 100 Mb total by subsampling your data.

Marking Rubric:

	Standard GOOD (80%)		Suggestions for exceeding the standard (a non-exhaustive list; do NOT do all)
Background */30*	The background information sufficiently outlines the biological knowledge necessary to understand the data relevant to the project. There is a **clear** discussion of the use case for the technique (e.g. coding context, audience, bioinformatics analyses, applications for the technique). There is **sufficient** technical explanation regarding the technique to assist users in understanding the **key** parts of the instructions/code and the output. A **brief** justification is given for why this technique was chosen (e.g. how is it better than alternatives?). **Clear** connections are made between the technique and research questions in science. If applicable, references to applications that require this technique or have used this package are **cited**. The background **does not exceed** 1000 words.		There is a compelling case for the use of this technique, and for the value of Python to achieve it. The background discussion for the technique is enough for someone outside the specific scientific discipline to comprehend and to appreciate the value of the technique. Concise but elegant theoretical background is given that effectively communicates how the tool works. More than one example of the technique at work is given and discussed as background. Extension activities of the technique are cited or proposed. A concise outline of the project workflow is described with brief justifications if required.
Instructions and code technique */30*	Instructions demonstrate the **essential** function(s) required to perform the technique and their **basic** usage. Instructions are exemplified using **appropriate** data. There is evidence that the tutorial and instructions run **correctly** without errors. If warnings exist, those are adequately addressed in the tutorial. Instructions demonstrate how to perform **basic** visualizations of the results of the analysis (e.g. to produce plots and/or tables). Instructions demonstrate how to summarize **key** results of analysis.		Instructions demonstrate more advanced function(s) usage that support the technique or expand its capabilities. Instructions demonstrate more advanced usage of function(s) (e.g. the usage of different parameters). Instructions demonstrate sensitivity of function(s) to parameter input. Instructions are exemplified using a tailor-made data set assembled to highlight the key functionality of the technique. Instructions are elegant, efficient, and/or use more advanced Python programming features (e.g. loops, if-else flow control, user-defined functions, etc.) while still effectively communicating the technique to users.
Commentary on code and output */30*	Code and its output (i.e. tables, plots, etc.) are **both** included and these are **contiguous** with discussion of results in a single logically-flowing document. Commentary guides the reader to clarify on the purpose and expected output of the ***upcoming*** code cells. Commentary **correctly** describes what is happening at the essential lines of the code (i.e. as inline code comments beginning with # AND/OR as text in the document immediately adjacent to the code). Commentary provides **sufficient** interpretation of the key elements of output immediately ***following*** the output to appreciate its meaning. Commentary **connects** the interpretation of the key elements of output to scientific phenomena of interest		Comments and instructions suitably bridge the project from section to section, producing a clear understanding of the project workflow. The instructions are thoroughly and clearly interpreted and is suitable for someone with only basic skills in Python to appreciate. The output is thoroughly and clearly interpreted and can be appreciated by someone without technical knowledge of the discipline.

	Standard GOOD (80%)		Suggestions for exceeding the standard (a non-exhaustive list; do NOT do all)
	as well as guiding/informing next steps in the workflow or future steps/improvements for beyond the project.		
Presentation */10*	Jupyter Notebook with background, data, code, and commentary, and its PDF/HTML versions, were submitted to QUERCUS. The entire document is consistently formatted to enhance readability. Large dataframe output is truncated using the appropriate commands (ie head() or tail())		Sections of the project are clearly noted/numbered and separated, including subsections where appropriate. Together the instructions and their output stand as a complete "vignette" of the technique and is of sufficient quality in terms of its technical content and presentation for posting as a tutorial on the web.

/100

Frequently Asked Questions

- My introduction is more than 1000 words. Is that okay?
    - I won't penalize you for being ***reasonably*** verbose, but I prefer that you are clear and concise. I'm not looking for a novel nor do I want to read about your cherished childhood memories that led you to this point in your data science journey. Give me what I need to know about your project so that I can understand why it's important, why you're doing it and why you're taking the direction you are with your analyses. Feel free to make your childhood story an appendix if you are *sure* it must be included somewhere.
- How do I use markdown cells vs. comments in code cells? What's the difference?
    - Your code should be commented to justify/clarify steps that are unclear. For instance, to remind the reader of what a specific variable represents or why you may be using a less conventional function. Markdown cells, on the other hand, should be used separately **to describe upcoming code cells or output from code cells**. This is where you can inform the reader of your plans like why you are about to remove or keep certain columns from your data frame; why the output was not as expected; what the interpretation of your visualization is, etc.
- How much code is enough code for this project?
    - This is **your** term project. It is meant to help you accomplish a goal which may be simple or it may be complex! I generally need enough code to get an idea of how well you understand core concepts in the course. As a rule of thumb, this can generally be captured in 20-40 code cells that utilize 10-20 different functions/ideas/paradigms. Likewise, the ratio of errors/poor coding choices within your coding cells is influenced by your total overall code length/complexity.
- Can I use packages from outside the course?

- **Absolutely**. Just be sure to explain your choices and how to use them as you use them – expected parameters and expected output.

- Can I use datasets from other manuscripts or labmates?

    - **Definitely**. If these datasets match what you plan to work on in the future or are appropriate for the tutorial/package you want to discuss, then by all means do so. Please credit appropriately in your references.

- Do I have to use every concept you taught in the course?

    - **Absolutely not**. Set out with a plan in mind, explain how you're going to accomplish your goals and use the commands you need to do that. The more you show me (to a reasonable extent), the more I can assess your skills but if you don't use some aspects like RegEx or context managers, I certainly won't penalize you. What's important is that your code works, it takes advantage of what we've learned, and it is well-commented or well-documented.

- What if my project doesn't have *enough* code in it?

    - Everyone's projects and needs can differ depending on what you are working on. How much code is enough code depends on how confident you are with your skills. The more code/examples I see, the better I can assess and forgive mistakes or inefficiencies. Less code simply means I have less to judge you on so make sure it's clear, and complete. Overall, your coding portion is 30% of your term project. 70% is dedicated to **written communication** about your project, code, results and your future directions. When in doubt, email me an outline before the end of lectures!

- My project isn't perfect, can I get an extension until it works the way I want it to?

    - **The simple answer is No**. The complicated answer is maybe, but on a case-by-case basis. Your code will hardly ever be perfect the first time you produce it. It's a work in progress and I want to see your progress. If you are having trouble with your commands, you still have two weeks to contact me or the TAs or search the internet. Again, it just needs to get the job done as best as you can. When I do mark your assignments, I'll suggest how you can fix or streamline your pipeline to meet your goals. If it's a bottleneck to your next step, you can always produce a "final" formatted file as you'd like to see it and use that on later steps if those work as planned.

- My data file is a GIGANTIC sequencing file for analysis can I submit that? If I run a subset of data, my results will look terrible!

    - When you ***submit*** your projects, you will have run the project notebook and you can use your original data file(s) when you generate the PDF with all of your output. The data file ***YOU submit*** can be a smaller subset that I use in case I need to run parts of your code. Of course, make a note of this somewhere in your intro/background too so that I know what I'm looking at. Alternatively take a screenshot of your data so I can get an idea of what it looks like going in and/or coming out of a command or series of commands. **DO NOT SUBMIT LARGE FILES!**

- Can I get any bonus marks for submitting my final project early?
    - I do not award any additional marks for early submission. However, if you are the first to email me with the title “I read the Introduction to Python term project rubric FAQs!” then you will be awarded a 0.5% to your final mark.
- What happens if I use code from the internet to help build my project?
    - Sometimes you will turn to the internet to help solve a data wrangling problem or some calculation that you need. In this day and age, it’s hard to avoid finding help from outside sources and this is especially applicable to coding. Whenever you use code (or data) from a lab member, the internet, chatGPT, or other source, you should **clearly reference these sources** and ensure that you clearly explain/comment the code so that **I KNOW** that **you understand** what you are implementing.