Introduction to Programming for Physics
PHY 110C
58913

Instructor Contact Information

Student Facilitator: Crystal Riley
Student Facilitator Email: crystal.sno.riley@gmail.com
Faculty Supervisor: Dr. Michael Marder
Faculty Supervisor Email: marder@chaos.utexas.edu

Course Website www.cs.utexas.edu/~csriley/phy110c

Course Description and Learning Outcomes

This course is designed to help students with a passion for the sciences get involved in projects and research by formally introducing them to basic programming concepts. There are many times in the sciences, when brilliant, ambitious minds are turned away due to lack of programming experience. This is a hands-on, book free, lecture limited course. Programming is a science learned by doing, not reading. This course is designed to be a laidback, collaborative atmosphere to have fun programming. This course will have emphasis on data processing and analysis, but will still give participants a strong basis to begin most programming projects. We will be coding Python for most of the course, however, we will build programming constructs, resources, and how to think like a programmer, allowing us the ability to begin development in his/her programming language of choice.

Course Structure

Once a week, the student facilitator will lead interactive seminars on basic computer science topics. In class we will introduce programming constructs, review examples, and solve small coding problems. Participants are encouraged to bring laptops to class (but anything that can run a python interpreter is fine) to participate in the development of these solutions. If a participant does not have access to any of these things for completion of the projects, out of class arrangements can be made. Some participants may become comfortable with concepts at different paces, and are welcome to aid those around them, or try to conquer more difficult problems related to the concept. At the end of lecture, a project will be assigned, applying the newly introduced concept into a more complex solution on a larger scale than what can be completed in class.

Projects

Every week there will be an assignment that the participant will have one week to

complete. These assignments are projects designed to be completed outside of class but will be similar to the problems we do in class. These will have multiple parts and will be an extension of the problems solved in class. Programming is cummulative, therefore later projects will likely be longer, including topics covered earlier in the course, while earlier projects will be shorter. These will be designed to be completable within 2 hours, so if there is confusion about a particular concept, please email me or come to office hours to get help. These projects are structured with designated goals; however, many of them will be open ended and expanding on projects is encouraged and rewarded.

Collaboration

Collaboration is encouraged, especially when participants are stuck on a particular concept. In class, collaboration is completely open. However, limitations do apply to the projects. Participants are not allowed to view or exchange one another's project code. Conversing about concepts, drawing abstract representations of the code, and asking about clarifications on specific functionalities are all allowed. All work submitted must be the participants original work, don't cheat. Researching similar concepts covered (on the internet, in books, etc) is encouraged, but citations are required when using another person's concepts in participant's code.

Grading Policy

Projects: 60%
In-class collaboration 40%
There will be no final exam for this course

Not all assignments will be graded out of the same number of points. Later labs will be weighted heavier, being that they are longer and cover more concepts. Participants who take projects above and beyond can receive greater than maximum points. The lowest project grade will be dropped. Projects are graded 70% on correctness, 20% on readability and 10% on documentation. This will encourage not only correct code, but developing reusable and maintainable code.

Attendance is mandatory, please show up. Each day will be 3% of your total grade. The participant will be given 1 of 3 pts for being prepared to class. To receive all points for a class day, the participant must actively engage in discussion and problem solving in class. Missing 4 class days will result in 0 pts awarded for attendance and a failing grade in the course. Note, this class is Pass/Fail only, meaning 60% is considered passing for this course.

Documented Disability Statement

Any student with a documented disability who requires academic accommodations should contact Services for Students with Disabilities at 512 471 6259 (voice) or 1-866-329-3986 (video phone) or http:www.utexas.edu/diversity/ddce/ssd as soon as possible to request an official letter outlining authorized accommodations. Students with a letter of accommodation

should meet with the instructor before class starts to discuss their accommodations.

Academic Integrity

Each student in this course is expected to abide by the University of Texas Honor Code. Work submitted by a student for academic credit should always have references for information or opinions they have accessed from other sources. For university policy:http://deanofstudents.utexas.edu/sjs/acint_student.ph

Tentative Schedule

Week 1: Setup Python, introduce interpreter invocation, and printing/formatting output
- Setting up Interpreter
- Begin ASCII Art Project

Week 2: Data types and operations
- Integers, Floats, Characters, Strings and their Operations
- Begin Accounting and Ledger Project

Week 3: Input forms and conditionals
- Peer presentation: if & elses, file reads & writes
- Begin Ad-lib Story Project

Week 4: Loops and vPython
- for each, while, range, break, continue
- Begin Kinematics Project

Week 5: Functions and lambdas
- function definition, evoking, arguments, lambdas
- Begin Ordered Logic & Cryptography Project

Week 6: Data structures - tuples, lists, arrays
- tuples, lists, arrays, sort, add, del, index
- Begin Critters Project

Week 7: Data structures - stacks, queues, sets, dictionaries
- LIFO, FIFO, list comprehensions
- Begin 4-in-a-row & board games Project

Week 8: Objects and classes
- classes, instances, objects, inheritance, scopes
- Begin Card Game Project

Week 9: Recursion - Trees, heaps
- basic recursion, trees, heaps
- Begin Permutations and Maze Project

Week 10: Modules and Packages - Data and tables, NumPy + SciPy
- importing and directories
- Begin Data Analytics Project

Week 11: Modules and Packages - Large Scale Data
- NumPy + SciPy routines
- Begin Advanced Calculator Project

Week 12: Modules and Packages - Visualization

- vPython
- Begin Gravity Simulator Project

Week 13: Learning new languages

- basics of Java
- basics of C
- Begin Language Loop Project

Week 14: Catch up