COMS250 – Special Topics in Computer Science, Introduction to Python (3 Cr., 2:2 hrs.)

Texts: Hill, Christian, Learning Scientific Programming with Python, 2016, ISBN: 9781139871754. In addition, numerous online texts and PDF files will be provided in the course LMS.

Semester: 15 weeks.

Instructor: Peter K. Schoch, Assistant Professor of Physics and Computer Science, 973-300-2342, pschoch@sussex.edu and office in L104 with office hours posted on the door.

Course Description: This course is designed to introduce the student to the programming language Python. Python and its various packages are essential tools for many programmers, engineers, researchers, and data scientists across academia and industry. In addition to learning the basics of Python, an introduction to parts of NumPy, SciPy, and Matplotlib will be covered.

Prerequisite: MATH112: Pre-Calculus II (grade of C)

Attendance: Attendance will be determined by the signing of an attendance sheet located on the instructor's podium. Because attendance is so important, prompt arrival is required. Unless an adequate reason can be given for an absence: late homework will not be accepted, make-up quizzes will not be given, and make-up tests will not be given.

Grades: The grade for the course will be based upon the following scheme:

Homework --	20%
Project work --	40%
Quizzes and Tests --	20%
Final Exam --	20%
Total	100%

The final letter grade for the course will follow the numeric scheme:

A	92	-	100
B+	86	-	91
B	81	-	85
C+	76	-	80
C	71	-	75
D	65	-	70
F	0	-	64

Homework requirement:

We have (approximately) 14 weeks of lecture time available to us during the semester. We shall be attempting to cover approximately 9 chapters of material during this time. While that may not seem like much, I assure you that the course is chock-a-block with material at a very challenging level.

The homework grade is an 'effort' grade. You are not expected to be able to solve all of the homework problems flawlessly straight out of the box. Nor are you expected to solve them in isolation. If that was the case, you needn't be taking the class! In the real world, you look things up, you 'experiment', and you consult with your peers to arrive at a solution. (Note that I am careful to state a solution, as many programming problems can be solved in several ways.) You are expected to attempt all of the problems, before asking questions about solution methodologies.

Project requirement:

In the real world, problems are not broken into "little chunks" that can be solved in 10 – 20 lines of code. Most of them require a complex and robust solution process. However, it is also important to note that those self-same solutions also reuse large portions of code, from sometimes disparate sources, to avoid re-creating the wheel. During the course of this class, I will be presenting you with less cookbook, and more robust challenges to solve using the skills you will be acquiring. You have the option of working alone on these projects or forming yourselves into teams of two people, with my approval, to work on them.

If you choose to work on the problems alone, you have the entire Internet as your resource, and me for consultation. However, you may not consult anyone else in the class

If you choose to work with another person, you both still have the entire Internet and me as resources, but now you can also collaborate to arrive at the final solution to the problem. Along the way, it will be incumbent upon you to document who worked on what portions of the problem to justify your grade. Failure to turn in this documentation along with the final project solution will negatively impact your grade.

Quizzes and Tests requirement:

All of these will be administered using the course LMS. They will test your understanding of the fundamental concepts covered in that section of the coursework.

Final Exam requirement:

COURSE COMPETENCIES/LEARNING OUTCOMES

In a manner deemed appropriate by the instructor and approved by the department, students will be able to:

1. Write Python language programs. (A.S. Science/Mathematics – Computer Science Option, Program Goal # 1)
2. Appreciate the importance of data structures in Computer Science. (A.S. Science/Mathematics – Computer Science Option, Program Goal # 1)
3. Use structured techniques in designing programs. (A.S. Science/Mathematics – Computer Science Option, Program Goal # 1)

4. Understand where testing fits in the software development process. (A.S. Science/Mathematics – Computer Science Option, Program Goal # 1)
5. Test programs adequately with appropriate input data. (A.S. Science/Mathematics – Computer Science Option, Program Goal # 1)
6. Implement Object-Oriented programming features using the Python language. (A.S. Science/Mathematics – Computer Science Option, Program Goal # 1)