IS 210: Software Application Programming I

Syllabus

College:	CUNY School of Professional Studies
Program:	Online Bachelor's Degree in Information Systems
Course Name:	Software Application Programming I
Course Code:	IS 210
Credits:	3.0
Required:	Yes
Prerequisite:	None
Instructor:	Javier Guillen

Table of Contents

Course Overview

Description

The ability to write software programs is a critical skill in the IS field. Students are introduced to the fundamental concepts and terms of computer science that are necessary to program software, with an emphasis on problem-solving and algorithm development. Concepts such as data types, control structures, modular organization, and object-oriented programming, using practical examples that highlight the design, implementation, and testing phases of programming, are explained. Important topics such as program documentation, input/output considerations, and information assurance are stressed. Students build several well-documented and well-designed integratable code modules to present in class.

Exams: Mid-term, Final

Projects: Final Course Project

Objectives

At the end of this course, students will be able to:

- Understand the basic components of a software application and computer system
- Explain the elements of computer programming languages
- Research and develop an algorithm for a real-world problem
- Develop programs for simple to intermediate problems

Instructor

Chad Heuschober, Software Development Manager, CUNY School of Professional Studies

Contact Information

E-Mail	Javier.guillen@sps.cuny.edu
GitHub	Javiguillen74

Textbooks

Required Text

Lutz, Mark, *Learning Python*, O'Reilly Media, Fifth Edition (July 6, 2013)

ISBN-13: 978-1449355739

ISBN-10: 1449355730

For purchase: http://www.amazon.com/Learning-Python-Edition-Mark-Lutz/dp/1449355730/

This test is also available as a free e-Book through the Baruch Newman Library. To access the resource visit the Baruch Newman E-Reserves Login Page [22]

The password is: heuschober210

Optional Texts

Pilgrim, Mark, *Dive Into Python*, Apress, (July 19, 2004)

ISBN-13: 978-1590593561

ISBN-10: 1590593561

For Purchase: http://www.amazon.com/Dive-Into-Python-Mark-Pilgrim/dp/1590593561

The material in the book is also available for free at: http://www.diveintopython.net/

Shaw, Zed A., *Learn Python the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code*, Addison-Wesley Professional, 3 edition (October 11, 2013)

ISBN-13: 978-0321884916

ISBN-10: 0321884914

For Purchase: http://www.amazon.com/Learn-Python-Hard-Way-Introduction/dp/0321884914

The material in this book is also available for free at: http://learnpythonthehardway.org/book/

Technology Requirements

Required

Participation in this course requires the following basic technology:

- A computer with reliable broadband Internet access on which you are an administrator
- A web browser compatible with Blackboard 9.x
- A GitHub user account

Development Environment

All software tools used in this course are open-source and available for free. CUNY SPS will provide a Virtual Machine environment with Ubuntu for OS, and Python 2.7 and IDLE installed.

Required Software

- Python [4] >= 2.7.5
- Git [6] >= 1.1.9.1
- *rst2pdf* >= 0.93

Required Python Modules

- pip [5] >= 1.5.4
- nose [7] >= 1.3.3
- pylint [8] >= 1.3.0
- mockito [9] >= 0.5.2
- pep8 [10] >= 1.5.7
- unittest2 [11] >= 0.5.1
- unittest-xml-reporting [12] >= 1.9.0
- docutils [13] >= 0.12

Text Editor

In addition to the above, you will also need some form of plain-text editor for code creation. Generally, students will be encouraged to use the IDLE python editor.

Other editors that could be considered satisfactory for this course include:

- TextMate [21] for Macs
- Notepad++ [16] for Windows

Students are cautioned against using Windows Notepad for code development as it does not support UTF-8 encoded files.

Students with a particular focus in software development and/or scripting may wish to consider command-line editors like Vim [14] or GNU Emacs [15] for their prevalence in the industry.

We won't be requiring an integrated development environment (IDE) in class, but many programmers find them useful. Some popular IDEs are:

- PyCharm [17]
- Ninja-IDE [18]
- PyDev for Eclipse [19]

GitHub

GitHub [1] is an industry-leading social coding website that provides nearly unlimited public repository hosting for open projects. Due to is popularity, exceptional design, and utilization of best-in-class technologies like Git [6], GitHub [1] has become the standard in open source project management. In addition to its project management features, GitHub [1] emphasizes the social aspects of collaborative design and implementation and, as a result, GitHub [1] profiles have become an invaluable tool to MIS hiring managers on par with, and even sometimes exceeding, such tools as resumés, cover letters, and ePortfolios.

Students will be find all assignment material in a GitHub repository, and should download the code directly into their Virtual Machines as per instructions. Student submissions will be to Blackboard, however. No submissions to Github will qualify for grading.

Why Python?

Many programming languages and computational tools (Excel, MATLAB, R, C/C++, Java) could be used to introduce programming. For many new to programming, Python is a good choice. It is a complete programming language, and it is simple to learn and easy to write and read. You can get started with it right away, and it comes with many great libraries that will allow you to do almost anything you want. Lastly, Python is a highly structured language with strong conventions, nearly all of which are transferable as best practices in other languages.

Learning Assessment

Students should gain understanding of the course topics and will be assessed through four primary categories: homework, quizzes, exams, and a student course project.

- Homework will be assigned on a weekly basis and enforces learning concepts through discussion and practice.
- Weekly quizzes will test student comprehension of the prior week's learning material.
- The mid-term and final exams will test concept synthesis and application.
- The final course project will challenge students to demonstrate mastery of course objectives in original contexts.

This course is designed to teach practical programming fundamentals using the Python programming language. The late mathematician Paul Halmos said it best when he stated, “The best way to learn is to do…” at the beginning of his paper on teaching practices. Therefore, a strong emphasis of this class is homework assignments.

Grading Weight

Categories of student assessment will be weighted by percentage in the final grade. The following table establishes grading weights for this course:

Categories of Assessment Grading Weight

Category	% of Final Grade
Homework	40%
Quizzes	20%
Mid-term exam	15%
Final exam	15%
Course Project	10%

Note

Due to this weighting, point values are not equivalent from category to category (eg, 10 homework points != 10 points on the final exam).

For reference, the following table contains the CUNY SPS Undergraduate Grading Scale.

CUNY SPS Undergraduate Grading Scale

Letter Grade	% Range	GPA Equivalent
A	93-100	4.0
A-	90-92.9	3.7
B+	87-89.9	3.3
B	83-86.9	3.0
B-	80-82.9	2.7
C+	77-79.9	2.3
C	73-76.9	2.0
C-	70-72.9	1.7
D	60-69.9	1.0
F	< 60	0.0

Course Schedule

Course Schedule

Activity
Week 01: Getting Started
Week 02: Programming in Practice
Week 03: Variables and Operators
Week 04: Conditionals
Week 05: Basic Functions
Week 06: Modules and Packages
Week 07: Lists and Tuples
Week 08: Loops
Week 09: Midterm Exam, Final Course Project Planning
Week 10: Dictionaries
Week 11: Advanced Functions

Week 12: Objects and Classes, Part 1
Week 13: Objects and Classes, Part 2
Week 14: Error Handling
Week 15: I/O and Third-Party Modules
Final Exam

Please note that the above calendar does not include the official CUNY SPS academic calendar [20]. To view the CUNY SPS academic calendar [20] see: http://sps.cuny.edu/academic_calendar.html

Time Allocation

The following table provides a summary of the type of work expected of each student:

Estimated Student Time Allocation

Type of Work	Estimated Time	Summary of Requirements
Readings	1-2 hrs/wk	2-3 chapters weekly
Videos	1-1.5 hrs/wk	1-2 videos weekly
Homework	6-9 hrs/wk	Programming assignments and/or discussions
Quizzes	12-24 min/wk	15-question timed quizzes
Exams	6-10 hrs each	A combination of a small programming task and concept test.
Course Project	5-10 hrs	A self-directed programming project

Lesson Components

Readings

Students will cover between two and three chapters per week in the primary course text. Additional materials will be provided to supplement the primary course text. Students who are already familiar with the lesson topic are still encouraged to read the source materials as they will become the basis for future quizzes, homework, and exams.

Videos

Each lesson will contain one or more videos to help contextualize the readings and offer tips, tricks, and best practices not covered in the readings.

Homework

Homework will primarily consist of small directed programming tasks that directly correlate with covered course materials. These assignments will be occasionally interspersed with discussion board assignments.

Each assignment will build upon concepts covered in the previous assignment. Since the homework assignments build on each other, keeping up to date is essential.

Homework will be assigned weekly. Each week will have a total of 30 points of assigned homework. This homework averaged and weighted to represent 40% of the final grade. The breakdown of points will be posted with the grading rubric with each assignment.

Programming

Programming tasks will form the bulk of student homework assignments. These will vary from raw creative tasks to analysis and modification of existing programs. Each question will be graded on a rubric that favors precision and completeness, with passing unit tests as the most important criteria for grading. Partial credit is not guaranteed.

Discussion

Students will occasionally be directed to discuss specific programming concepts as a component of that week's homework. Regular participation in such discussions advances student understanding and makes for a more rewarding class experience.

When a lesson contains a discussion component, each student will be directed to post one to two times according to the directions. The student first posts a new discussion topic (a.k.a. thread) in response to a question posted by the instructor. If the assignment requires two responses, the initial student response is due by early that class week (See the course schedule for actual due date). If posted after the due date it will be subject to the late policy for this course. The final response is due by the end of the week (See the course schedule for actual due date). All posts should add to the conversation, express evidence that the original post was fully read and take the conversation further. Generic responses such as "I agree" or "Good job" do not meet class discussion response requirements.

Discussion assignments will be be weighted in the *Homework* component of the final grade. Students are cautioned not to rely on discussion assignments to carry their grades in this course as the overwhelming majority of homework will be practical programming assignments.

Submission

Student programming assignments will be communicated through GitHub [1], an industry-leading code management repository. For each lesson, students will download starter code from GitHub [1], make changes as instructed, and submit the code via Blackboard. Submissions via GitHub will not be graded.

Assignments are due each Sunday at 11:59 PM EST.

Grading Policy

This course employs a grading rubric that favors completeness and accuracy when awarding points. Homework is divided into a number of tasks each graded independently and weighted accordingly. You may only receive credit for a given task if:

- Your code runs and does not produce an error unless explicitly directed to do-so.
- You have achieved the main objective of the task and demonstrated your knowledge of the core concepts being assessed.
- You have fully documented your work according to the documentation guidelines introduced in week #2.
- Your code follows the syntax and style guidelines introduced in week #2.
- Your code passes lint tests.
- Your code passes unit tests.

- Your code follows best practices after their introductions in the course material. This will most regularly apply to such principles as "Once and Only Once" (OAOO) and "Don't Repeat Yourself" (DRY).

Why So Strict?

Believe it or not, this is just a form of preparation for field work. The tasks you will have to perform in class will have only a fraction of the complexity you will encounter in any future programming endeavors and the above goals should all be achievable. By practicing rigor from the beginning you will develop excellent programming habits to take forward in your academic and professional careers.

Programming is a task that requires precision and attention to detail. Mistakes in programming have been known to wreck ships, misdeliver medical supplies, crash markets, and break websites. This is why programming accuracy scales start at 99.9% and move up to 99.999%.

Late Work Policy

No late homework will be accepted or is eligible for credit barring exceptional circumstances negotiated through and documented by the CUNY SPS Office of Student Services.

The material of each lesson of this course builds upon the concepts and tools introduced in prior lessons. As a result, students who fall behind will be incapable of completing the current assignment or comprehending current course materials.

Quizzes

Most regular weeks, students will take a 12-minute, 15-question, open-book, open-internet quiz covering the previous week's material. These quizzes are timed and are worth 15 points each and are weighted, in total, to represent 20% of the final grade. Quizzes will be available from Monday through Sunday most weeks. You will have two attempts per quiz; the highest final score will be kept for grading purposes.

Late quizzes will not be available for credit.

Course Project

This project should use the accumulated body of knowledge covered during this course. Students will select, with advisement, a relevant programming topic and design, document, and implement a program to satisfy their objective. Students will then be asked to critique each others projects. This project is worth 15% of your final grade.

More details regarding this project will become available part-way through the course.

Exams

This course contains one midterm and one final exam, each of which are weighted to represent 15% of the final grade. Both exams will include course materials covered up to that point and will consist of an open-book timed test and a programming task. Details regarding the exams will be covered in-depth as we get closer to the exam dates.

Extra Credit Discussions

On select weeks, students will have an opportunity to participate in an extra credit discussion on the discussion boards. It is important to note that such credit is offered only to those students who complete the required components of the lesson (eg, assignments and quizzes). Extra credit is offered in addition-to **not** replacement-of credit earned through normal means. Students are recommended to note the extra credit discussion guidelines in the course site for more details.

Course Policies

Student Conduct

All CUNY School of Professional Studies students are responsible for behaving in a manner consistent with the policies and standards set forth by the School. The policies are available in the Academic Polices Handbook. We encourage our students to become familiar with these polices.

Please visit the following URL for additional information: https://sps.cuny.edu/acad_policies/

Non-Discrimination

There will be no discrimination on the basis of sex, race, color, national origin, sexual orientation, religion, ideology, political affiliation, veteran origin, sexual orientation, religion, ideology, political affiliation, veteran status, age, physical handicap, or marital status.

Online Participation

You are expected to read the assigned texts and participate in the discussions and other course activities each week. Assignments should be posted by the due dates stated on the grading schedule in your syllabus. If an emergency arises that prevents you from participating in class, please let your instructor know as soon as possible.

Attendance Policy

Attendance will be counted as having posted an assignment during the week it was due (discussion, paper, quiz, etc.) A class week is defined as the period of time between Monday 9:00am and the following Monday at 9:00am except for the last week of the final period and holidays, when the week and the course will end on Sunday at midnight. The course and system deadlines are all based on the U.S. Eastern Standard Time Zone.

Online Etiquette and Anti-Harassment Policy

The University strictly prohibits the use of University online resources or facilities, including Blackboard, for the purpose of harassment of any individual or for the posting of any material that is scandalous, libelous, offensive or otherwise against the University's policies. Please see: http://media.sps.cuny.edu/filestore/8/4/9_d018dae29d76f89/849_3c7d075b32c268e.pdf

Academic Integrity

Academic dishonesty is unacceptable and will not be tolerated. Cheating, forgery, plagiarism and collusion in dishonest acts undermine the educational mission of the City University of New York and the students' personal and intellectual growth. For more information on Academic Integrity at The CUNY School of Professional Studies, see: https://sps.cuny.edu/acad_policies/acad_integrity.html

Support Services

Accessibility and Accommodations

The CUNY School of Professional Studies is firmly committed to making higher education accessible to students with disabilities by removing architectural barriers and providing programs and support services necessary for them to benefit from the instruction and resources of the University. Early planning is essential for many of the resources and accommodations provided. Please see: http://sps.cuny.edu/student_services/disabilityservices.html

Additional Help

If you need any additional help, please visit Student Support Services: http://sps.cuny.edu/student_resource

References

1(1, 2, 3, 4, 5, 6, https://github.com/
7, 8, 9, 10, 11,
12)

2 https://www.vagrantup.com/
3 https://www.virtualbox.org
4 https://www.python.org/
5 https://pypi.python.org/pypi/pip
6(1, 2, 3) http://git-scm.com/
7 https://pypi.python.org/pypi/nose/
8 https://pypi.python.org/pypi/pylint/
9 https://pypi.python.org/pypi/mockito/
10 https://pypi.python.org/pypi/pep8/
11 https://pypi.python.org/pypi/unittest2/
12 https://pypi.python.org/pypi/unittest-xml-reporting/
13 https://pypi.python.org/pypi/docutils
14 http://www.vim.org/
15 https://www.gnu.org/software/emacs/
16 http://notepad-plus-plus.org/
17 http://www.jetbrains.com/pycharm/
18 http://ninja-ide.org/
19 http://pydev.org/
20(1, 2) http://sps.cuny.edu/academic_calendar.html
21 http://macromates.com/
http://remote.baruch.cuny.edu:2048/login?url=http://ereserve.baruch.cuny.edu.remote.baruch.cuny.edu/eres/coursepage.aspx