AUGUSTA UNIVERSITY
SCHOOL OF COMPUTER
AND CYBER SCIENCES

Augusta University
School of Computer and Cyber Sciences
Summer 2019

Introduction to Computers and Programming
CSCI 1200

Instructor:	Neea Rusch
Email:	nrusch@augusta.edu

Office Hours	See me after class
Course Dates:	May 20 – July 15, 2019
Final Exam Date:	July 12, 2019 (Friday, 6:00 – 8:00 pm)

Lecture: University Hall (UH) 162

Mondays	5:30 – 6:00 pm
Tuesday	5:30 – 6:00 pm
Wednesdays	5:30 – 6:00 pm
Thursday	5:30 – 6:00 pm

Lab: University Hall (UH) 162

Mondays	6:10 – 7:20 pm
Tuesday	6:10 – 7:20 pm
Wednesdays	6:10 – 7:20 pm
Thursday	6:10 – 7:20 pm

Learning Outcomes

By the end of this course, students will be able to solve computer programming challenges using correct, well-structured programs written in the Python programming language. Students will have a basic knowledge of and experience with variables, simple data types, lists, dictionaries, iteration and selection structures, functions, classes, and file operations. The student will develop computational thinking skills and practices.

Course Description

CSCI 1200 – This course emphasizes analytical thinking and teaches problem solving through an introduction to basic programming structures. It covers design of well-structured algorithms using appropriate logic structures with simple data types and data structures.
(Source: Augusta University 2018 – 2019 Course Catalog)

Format and Procedures

This course consists of lectures and lab periods. Lab exercises and a programming projects will augment and reinforce the lectures.

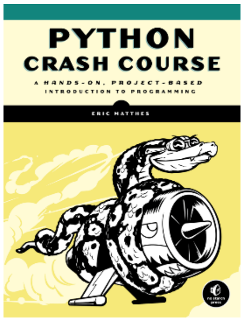
Teaching Philosophy

CSCI 1200 is a three (3) credit hour course to introduce students to computers and computer programming using the Python programming language. I assume no previous programming experience. The lab exercises allow the students to learn and practice programming skills and techniques and prepare for projects. Projects provide an opportunity to demonstrate the learned skills. The emphasis of all class activities is understanding ‘why’, not just ‘how’.

Course Requirements

- Class Attendance. Class attendance is a significant indicator of course success. I expect you to be in class. I recognize the need to miss classes due to unforeseen circumstances.
- Class Participation. The next key indicator of course success is to pay attention and participate. You are expected to complete all course work in a timely manner.
- Reading. Our required textbook is:

> Matthes, *Python Crash Course*, © 2016; No Starch Press;
> ISBN-10: 1-59327-603-6
> ISBN-13: 978-1-59327-603-4



This textbook is available at the JagStore and online in including via Safari Books Online (https://www.safaribooksonline.com).

- Homework. Problem sets will be provided for each book chapter. Homework problems help reinforce theoretical understanding and facilitate learning. You are highly encouraged to work through each problem set.
- Assignments. Lab exercises and projects afford you the opportunity to apply the skills and techniques we discuss in lecture. I encourage discussions of concept and design.

Write your own code. To do otherwise is to plagiarize.

Grading

This course consists of five modules where each module covers two book chapters. There will be one (1) project and one (1) exam per module. At the end of the course there will be a final programming project and cumulative final exam. Homeworks and labs are not graded, but help you to succeed with projects and exams that are graded.

- **Projects**. Projects measure your learning in programming. Each project is based on a module and focuses on topics introduced in the respective module.

- **Final Project**. A capstone-like end-of-course project allows you to apply the tools and skills you learn throughout the course. The last week of the course, lecture and lab periods, will be dedicated to the final project.

- **Exams.** Exams assess your understanding of the topics covered in the course. The exams will include multiple-choice, true-false and short-answer questions. You will take your exams on Brightspace/D2L in classroom under instructor supervision. Under no circumstances are you allowed to take exams independently or without supervision. Failure to show up for an exam at the scheduled date and time without prior approval by instructor will result in automatic grade zero (0).

- **Final Exam**. The Final Exam is cumulative and assesses your understanding of the topics covered in the course. The Final Exam will include multiple-choice, true-false and short-answer questions.

These graded components are weighted in the following manner:

Projects (5)	30%
Final Project	10%
Exams (5)	40%
Final Exam (1)	20%
Total	100%

Letter Grade Methodology

A – 90% and above
B – 80 – 89%
C – 70 – 79%
D – 60 – 69%
F – Below 60%

- **Extra Credit**. Opportunities for extra credit may be introduced throughout the course. Extra credit tasks typically involve demonstrating understanding of advanced course topics or tools.

Supplemental Activities
You are highly encouraged to complete reading assignments, homeworks and lab exercises to succeed in this course. These activities help reinforce your learning and prepare you for the activities that will be graded.

- **Reading and Homework**. You are expected to read each chapter *prior to* the first time it is discussed during lecture. I provide chapter-related problem sets to test your high-level understanding of chapter concepts and topics. These problem sets are not graded, but they help you to identify important topics in each chapter and verify your understanding of the theory. You are highly encouraged to complete the homework in order to succeed in this course.

- **Lab Exercises**. I design lab exercise to allow you to practice the programming techniques and structures we discuss in lecture. I expect you to attend lab periods each week to complete the lab exercises. Lab exercises are a required prerequisite for succeeding in class projects.

Project Submissions
During this course you are expected to submit programming projects for grading. For full credit I expect you to complete each project by the assigned due date. Follow the instructions provided with each project on how to submit your work for grading.

Class Project Submission
For regular course projects, I will accept late projects up to 3 days after the due date with a 10 % late penalty. I will not accept late projects more than 3 days after the due date, the resulting grade will be a zero (0).

Final Project Submission
For full credit submit project as described in the assignment instructions by the due date. I will accept late submissions within the following timelines and with the following late penalties:
- 10% penalty if submitted within 24 hours of the due date,
- 20% penalty if submitted between 24 and 48 hours after the due date.

I will not accept a submission that is more than 48 hours late. The resulting grade will be a zero (0). Because the project is due on the last day of class there is no room for added leniency.

Exam Absence
Exam absences must be coordinated with me prior to the exam. Under certain circumstances and with prior permission, I may grant you permission to count your Final Exam grade as a missed exam grade. Note that this allowance is available only with my prior permission and is only available to replace one missed exam. Unexcused missed exams will result in a grade of zero (0).

You will take all your exams on Brightspace/D2L in classroom under instructor supervision. Under no circumstances are you allowed to take exams independently or without supervision. Failure to show up for an exam at the scheduled date and time without prior approval by instructor will result in automatic grade zero (0).

Final Exam Absence
Absence from the Final Exam without prior coordination with me, or a valid emergency, will result in a grade of zero (0) on the Final Exam.

Course Withdrawals
Course withdrawals prior to midterm will result in a 'W'. Course withdrawals after midterm will normally result in a 'WF'. It is your responsibility to initiate a course withdrawal.

Academic Integrity
Write your own code! Programming is normally a team sport. Good programmers work well with others and as part of teams.

I encourage you to study together and to discuss information and concepts with other students. This permissible cooperation must never involve one student having possession of a copy of all or part of work done by someone else, in the form of an email, email attached file, flash drive, hard copy, passed via spoken word or other form.

I may check your assignments via TurnItIn, a tool that compares text to Internet and other sources in its repository. TurnItIn produces an Originality Report that indicates the amount of original text in your work and identifies plagiarism.

***** Students will receive a score of zero (0) on plagiarized assignments. *****

Academic Accommodations
- Augusta University will make reasonable academic accommodations for students with documented disabilities. Students should contact Testing and Disability Services (Galloway Hall; 706.737.1469; www.augusta.edu/tds/) as soon as possible for more information and/or to initiate the process for accessing academic accommodations.
- If you are registered with Testing and Disability Services, and have not already done so, please see me as soon as possible to discuss your accommodations and how I may be of assistance to you throughout the course.

Course Recommendations
- Access to a modern computer or laptop is recommended, especially if you chose to work on Final Project outside the lab.
- Use of git version control system is highly recommended for class projects and lab exercises.

Software Requirements

- We will use **Python 3** in this course. You can download and install Python 3 on your own machines (at no cost) at www.python.org.
- We will also use **Geany**, a lightweight Graphical User Interface (GUI) text editor for our Python programming. Geany is loaded on our University Hall 162 lab machines, as well as in AH N126 and AH N344 lab machines. You can download and install Geany on your own machines at no cost at www.geany.org.
- **Sourcetree** is not required, but is a highly recommended, user-friendly Git GUI that helps to manage code files. You can download and install Sourcetree on your own machines (at no cost) at www.sourcetreeapp.com.

Resources

- Reese Library Cyber Resource Center (http://guides.augusta.edu/friendly.php?s=cyber). A great source for cyber resources available to the public and Augusta University students, faculty and staff. Available resources include subscription-based, online CD journals, books and other publications.

Keys to CSCI 1200 Success

- Think
- Read
- Attend class
- Write a lot of code
- Work through problem sets
- Work through lab exercises
- Complete projects on time
- Prepare and attend each exam

See me early if having issues