COP 3223H Syllabus
Honors Introduction to C Programming – Fall 2016

Course Prerequisites: No formal requirements except a background in precalculus mathematics
Class Time: MWF 12:30 – 1:20 pm
Class Location: ENG1 – 286
Course Web Page(s): http://www.cs.ucf.edu/courses/cop3223/fall2016/
http://www.cs.ucf.edu/courses/cop3223/fall2016/section201
Lecturer: Arup Guha
Office: HEC – 240 **Email:** dmarino@cs.ucf.edu
Phone Number: 407-823-1062
Office Hours: TR 10:30 - 11:30 am, MTWR 2:15 - 3:15 pm

Course Description: COP 3223 provides an introduction to programming for those with ***no prior programming experience***. Two languages are definitely going to be taught: Python and C. In Python, we'll look at variables, I/O, if statements, loops, functions, strings, lists, sets, dictionaries and the python turtle. In C, we'll cover variables, I/O, if statements, loops, functions, arrays, strings, and structs. If the class is picking up C well, we will also learn a few bits and pieces of C++.

Recommended Book: Programming Knights: An Introduction to Programming in C by Guha, ISBN-13: 978-1-256-92763-1.

The goal in writing this book was to bring the cost down for students and also to provide a book that mirrored how the course is taught. This book will be cheaper than a new copy of the previous book, but potentially more expensive than a used copy of the previous book. The previous textbook (before the custom text) was:

C Programming by K.N. King, ISBN 978-0-393-97950-3.

A good student will most likely be able to do well in the class by consulting my online notes and examples, as well as doing google searches to fill in missing holes that inevitably may come up. Thus a textbook is probably only necessary for students who prefer that sort of pedagogical structure completely laid out in book form.

Other Important Course Policies:

1) Cheating will not be tolerated. **If a student is caught cheating, then the grade on that assignment for all students knowingly involved (the person providing answers as well as the one taking the answers) will be a -25%.** (**Note, this is less than 0%.**) Since discussion of concepts with other students is often helpful, cheating must be more clearly defined. In particular, the following items are cheating: copying a segment of code of three lines or more from another student from a printout or by looking at their computer screen, taking a copy of another student's work and then editing that copy, and sitting side by side while writing code for assignments and working together on segments of code. In all of these situations, **BOTH people responsible**, the one from whom the three lines of code are taken as well as the person who takes those lines of code are engaging in academic misconduct. For example, if someone makes an electronic copy of their code accessible to ANYONE in the class (except for themselves) before 48 hours after an assignment is due, they are automatically culpable of academic misconduct. It does not matter if the recipient of the code doesn't use it, uses it a little, or copies it directly. Furthermore, based on the severity of the case, the entire course grade for the student may be lowered an entire letter grade. ***If you get stuck on an assignment, please ask either a TA or me for help instead of getting help from another student.*** Part of the learning process in programming involves debugging on your own. In our experience, when a student helps another student with an assignment, they rarely allow the student getting help to "figure out" problems on their own. Ultimately, this results in a lack of debugging experience for the student receiving help. The goal of the TAs and instructors is to provide the facilitation necessary for students to debug and fix their own programs rather than simply solving their problems. **But, you are encouraged to work together on any non-graded programs to enhance the learning process, particularly in your study groups.**

2) In order to take a make-up exam, you must request one from the instructor. The instructor will grant requests using his own judgment by applying the following general rule: "Make-up exams will only be given if the reason for missing the exam was out of the student's control." For example, being hospitalized unexpectedly is out of a student's control, but oversleeping or going to happy hour is not out of a student's control. ***If possible, it is recommended that the instructor be contacted before the exam.***

3) NO LATE ASSIGNMENTS WILL BE ACCEPTED. If a work/military/family/personal reason prevents you from turning in an assignment on time, please contact me as early as possible and I will make alternate arrangements. If the situation is serious enough, I may suggest pursuing an Incomplete or dropping the course.

4) Both the course web page and WebCourses will be crucial elements of the course. ***It is your responsibility to check both of these before every class meeting for any updates that may be posted.*** Some clarifications may only be given in class and won't be posted online at all, so make sure you keep up with announcements in class.

Important Note for Beginning Programmers

The amount of graded work is minimal compared to what is actually necessary to learn how to program in C comfortably. I strongly suggest that you write ***more*** programs than are actually assigned for you to turn in. The course web page will contain several suggestions of programs to write. Most lectures will contain sample programs that you can cut and paste, compile and edit. You may show me extra programs you have written at any time. Since these aren't graded, I can give you more feedback and help with them than on assignments. Also, it is encouraged that you write extra programs with others in the course; often times learning is facilitated by working with others. **You must work alone for all graded programming assignments, however.**

As the note above indicates, this class is far more time consuming than other introductory classes for many students. Unlike other introductory classes which simply ask students to read and regurgitate information, this class requires students to learn a new language, and then use that language to solve non-trivial problems that students have not seen before. It takes most students some time to get comfortable and confident with their programming skills. For each student, the amount of time and practice necessary to reach this level is different. Although we don't want to scare anyone, we do want to be up front about the fact that this class is a lot of work, and it's a different type of work than many other classes. (For example, it's possible you may get stuck on one error for a couple hours without making any progress. Once you fix that error, you might be able to finish your whole program in another thirty minutes.) ***I would imagine that for a good introductory student, on average, this class will require a total of 10 hours of work per week.***

Important Note for Experienced Programmers

Although you may understand all the general concepts of programming taught in this class, if you've never programmed in C, you may not know some of the very specific details that differentiate C from other languages. I write test questions that highlight these subtle differences and other subtleties of the C language. Make sure you read the textbook carefully so that you are aware of these details. We have had instances of a beginning programmer receiving the highest grade in the course, beating out many others who had been programming for three or four years because the beginner actually read the textbook and paid attention to those details that the experiences programmers missed.

Grading

The final letter grade will be based upon the seven items listed below. Plus/minus grades will be issued, when deemed appropriate.

Item	Percentage
Individual Homework Assignments	15
Final Project	10
Friday Problems	10
Study Group Notes	5
Exams #1, #2, #3	15% each, drop lowest
Final Exam (C and/or C++)	30

In order to pass the class you must earn at least a 40% on the final exam. (Thus, if you have a 75% in the course but earn a 30% on the final, you still get a C- in the course even though your percentage may qualify for a B.)

Letter Grades

I assign letter grades a bit differently than other professors. I do not use a straight 90-100, 80-90, etc. grading scale. Rather, at the end of the semester, I chose my lines for each grade. The drawback to this technique is that students do not know exactly what letter grade they are earning during the semester. The advantage to it is that if I make a difficult exam, I can adjust my grades accordingly so students don't get punished for my exam making skills. In the past, my A line has ranged from about 83-87%, my B line has ranged from 67-70%, and my C line typically ranges from 52-55%. I do not guarantee that these will be accurate for this semester, but I wanted to give you a rough idea of how the grades have gone in the past. After each exam, I will update WebCourses to show you what letter grade you have at that point in time. If you have further questions about my grading philosophy, please read the document I have posted at http://www.cs.ucf.edu/~dmarino/ucf. **Note: This grading breakdown is subject to change. Any changes will be discussed in class.**

Individual Programming Assignments

All programming assignments will be turned in over WebCourses. It's critically important to do these assignments in order to aid understanding of the course material. In order to grasp the material fully and feel comfortable with both Python and C, one needs to write **MORE** programs than are assigned. Several ungraded programs will be posted on the course web page, so students can get the necessary practice. Students are encouraged to work on these programs and come into see me for further help. **The due dates for each assignment will be posted on WebCourses ONLY.**

Community Service Opportunity

In lieu of the last individual assignment, you may perform 5 hours (or more) of community service. **You may NOT use any community service that you are doing for UCF already, such as the community service related to Symposium.** If you take this option, then you will automatically get a 100 for the last individual program. In order to get this credit, you must complete the community service and turn in the requisite form signed by the **October 28, 2016 in class.** ***NO LATE FORMS WILL BE ACCEPTED, PERIOD. Note: Your community service MUST BE with a registered 501(c)(3) organization to count for this assignment.***

Late Assignment Policy

NO LATE HOMEWORK ASSIGNMENTS WILL BE ACCEPTED. Due to possible server issues, it is **strongly suggested** that you attempt to submit programs **at least three hours** before the actual time it's due.

Friday Problems

On some Fridays, students will be asked to work in pairs on a programming problem. On each of those weeks a few pairs of students will be assigned to complete the Friday problem and turn it in for credit. Each student must submit two Friday problems, each of which will be worth 5% of the course grade. (Note: Each submission counts for both students in the group that submits the program. Groups may change from week to week.)

Exams

For each exam, you'll be allowed to use a limited amount of notes as an aid. The specifics for the allowable aids will be given in the class period before the exam for each exam.

Study Groups

I will assign study groups after the first couple weeks. Groups should meet once a week (physically or virtually) to review course material thereafter. I will collect notes detailing these meetings and grade those for the study group grade.

Final Project

Eight weeks into the class, students form groups in pairs for a final project. Students will propose a project, plan it and implement it, showing off their programs in the final week of class in a 10 minute presentation. The goal of the project is to have students work on a more intricate program (with many functions) of their own design, as opposed to solving a problem given to them.

Tentative Schedule

Week	Monday	Wednesday	Friday	Reading
Aug 22-26	Make Syllabus	Python/IDLE	Formula Probs	Cht 1
Aug 29-Sept 2	Math Lib	If Stmt	Friday Problem	Cht 2
Sept 6-9	**Labor Day**	Loops	Loops/Turtle	Cht 3
Sept 12-16	Strings	Lists, Sets	Friday Problem	Cht 4
Sept 19-23	Dictionaries	Review	**Exam #1**	Cht 4
Sept 26-30	C Basics	C - If	C - Loops	Cht 5 - 8.1
Oct 3-7	C quirks	Loop Control	Friday Problem	8.2-8.3
Oct 10-14	C functions	C functions	C functions	Cht 13 - 14
Oct 17-21	Arrays	Review	**Exam #2**	Cht 11
Oct 24-28	2-D Arrays	Strings	Friday Problem	Cht 12, 15
Oct 31-Nov 4	Structs	Structs	Friday Problem	Cht 16
Nov 7-10	TBA	TBA	**Veterans Day**	TBA
Nov 14-18	TBA	TBA	Final Proj Day	TBA
Nov 20-22	TBA	**Exam #3**	**Thanksgiving**	TBA
Nov 28-Dec 2	Presentations	Presentations	Presentations	None
Dec 5-9	**Optional FE Review**		**Final Exam (10 am – 1 pm)**	