Course Syllabus

Professor and TA Contact Information:

Name	Kamran Z. Khan
Phone	214-280-7124 (Cell)
Office Location:	ECSS 4.403
Office Hours:	Tue/Thurs 5:30-7:00 pm. Email me for an apt first
Email:	kkhan@utdallas.edu
TA:	TBD

COURSE INFORMATION

Course Number	CE 1337, CS 1337, TE 1337 Section 501 Mon/Wed 7:00 – 8:15pm, ECSS 2.203
Course Title	Computer Science I
Credit Hours	3
Term:	Fall 2015

Course Pre-Requisites, co-requisites, and/or other restrictions

Prerequisites	CS 1336 with a grade of C or better or equivalent. (Same as CE 1337 and TE 1337) (3-0)

Course Description: Review of control structures and data types with emphasis on structured data types. Applies the object-oriented programming paradigm, focusing on the definition and use of classes along with the fundamentals of object-oriented design. Includes basic analysis of algorithms, searching and sorting techniques, and an introduction to software engineering. Programming language of choice is C/C++

Student Learning Objectives/Outcomes: After successful completion of this course, the student should have an:

- Ability to use single and multi-dimension arrays.
- Ability to implement linear and binary searches.
- Ability to implement simple sorting algorithms.
- Ability to implement structured data types.
- Ability to define and implement a class.
- Ability to use fundamentals of object-oriented design.

Required Textbooks and Materials:

Starting Out with C++, From Control Structures through Objects (8th edition – orange slice)
Gaddis, Tony; Pearson Publishing ISBN 0-13-376939-9

Extra material may be posted on eLearning

Important Dates:

August 24	Classes start
September 7	Labor Day
September 9	Census Date
September 18	Project 1 due
September 23	Exam 1
October 5	Project 2 due
October 28	Project 3 due
October 29	Last Day to Withdraw
November 2	Exam 2
November 18	Project 4 due
December 10	Project 5 due
TBA (Finals Week)	Exam 3

These descriptions and timelines are subject to change at the discretion of the Professor.

University Policies

For all other University policies, please visit http://go.utdallas.edu/syllabus-policies

Date	Topic	Reading Assignment
8/24-26	Introduction to CS 1337 Introduction to Code::Blocks Searching and Sorting Arrays	Read Chapter 8
8/31-9/2	Searching and Sorting Arrays Advanced File I/O	Read Chapter 12 (omit 12.7, 12.8)
9/7-9	Labor Day Advanced File I/O	Read Chapter 19
9/14-16	Advanced File I/O Recursion	Project 1 Due
9/21-23	Recursion Exam 1 (Chapters 8, 12, 15)	Read Chapter 10
9/28-30	Characters, Strings and the String Class	
10/5-7	Characters, Strings and the String Class Pointers	Project 2 Due Read Chapter 9
10/12-14	Pointers Structured Data	Read Chapter 11
10/19-21	Structured Data Linked Lists	Read C. 17.1 & 17.2 Review Linked List PDF
10/26-28	Linked Lists	Project 3 Due
11/2-4	Exam 2 (Linked Lists, C. 10, 11, 13) Introduction to Classes	Read Chapter 13
11/9-11	Introduction to Classes	Read Chapter 14
11/16-18	More About Classes	Project 4 Due
11/23-25	Fall Break	
11/30-12/2	More About Classes Inheritance, Polymorphism & Virtual Functions	Read Chapter 15
12/7-9	Inheritance, Polymorphism & Virtual Functions Exam 3 (C. 13, 14, 15)	
12/10		Project 5 Due

DO YOU NEED ASSISTANCE?

E-mail:
Please include your course and section either in the subject or the body of your e-mail (preferably on the first line if not in the subject). This will allow me to address your e-mail as quickly as possible.

Help Desk:
For help with issues regarding your computer, UTD maintains a walk-in help desk. Visit their Web site for details: http://www.utdallas.edu/ir/helpdesk/

Tutoring:
For programming assistance in CS1337, a tutoring lab will be maintained. The schedule usually comes out a couple of weeks after the semester begins. Once the tutoring schedule for this semester has been released, an announcement will be posted on eLearning. In addition, it is part of the TA's job to help you, so please feel free to engage with him/her at any time. And, of course, I'll be happy to help as well.

Resources:

- C++ language tutorial: http://www.cplusplus.com/files/tutorials.pdf
- C++ reference: http://www.cppreference.com
- C++ tutorial: http://www.learncpp.com/

WHAT DO I NEED DURING LECTURE?

- **Notebook –** You are going to take a lot of notes and have quizzes. Bring paper to write on.

- **Writing Tool** – pen, pencil, crayon, etc. It's hard to take notes without one.

- **Textbook**: Starting Out with C++, From Control Structures through Objects (8th edition – orange slice) ; Gaddis, Tony; Pearson Publishing ISBN 0-13-376939-9
    - As you read the text, watch the corresponding VideoNotes. The VideoNotes are available at http://www.pearsonhighered.com/gaddis/.
        - NOTE: VideoNotes are only available if your book comes with an access code. If your book does not have an access code, you can buy one online at the above address. **The access code is not required for class**, but some of you may find the material accessible with this code to be a good resource.

  - Students may use a different edition of the book. In doing so, students accept the responsibility of verifying page numbers for assignments as well as learning the C++ 11 topics not present in the 7th edition. I will not copy any information from the 8th edition for students.

- **C++ Compiler (Required)**
  - All projects you submit will be compiled with Code::Blocks. This is a free download for either Windows or Mac. http://www.codeblocks.org/downloads/26 - download **codeblocks-13.12mingw-setup.exe**

  - Be advised that Code::Blocks may not work with newer versions of the OS X operating system. If you have a Mac, it is highly recommended that you have a Windows partition on your computer for code development.

  - **If a student uses a compiler other than Code::Blocks for development, he/she is responsible for verifying prior to submission that the code works properly in Code::Blocks.** No compiler is perfect and each one has its own quirks. It is the student's responsibility to make sure that the program functions as expected within the IDE that will be used for grading (Code::Blocks).

  - If you intend to use your own computers to write the class assignments, it is important that you get a compiler downloaded, installed, and running on your computer as soon as possible. If you don't have a computer, or if you're having problems getting a compiler installed, you should write your programs in the labs until the problems are resolved. In any case, please note that you are responsible for getting the programming assignments written and turned in on time. Since there are many computers available on campus, problems with your local machines will not be accepted as an excuse for not doing the assignments or late submissions.

WHAT DO WE EXPECT OF EACH STUDENT

- **Ask for help.** Email me or stop by during office hours. I want you to succeed. I would rather point you in the right direction so that you can complete an assignment instead of you remaining quiet and getting a failing grade.

- **Ask questions any time!** During lecture, before/after class, during office hours.

- **Take responsibility for your education.** I am not the type of teacher that lectures with slides and expects students to memorize. I will teach by creating program examples in class. I will give challenging assignments to push students toward learning the intricacies of C++. Part of being a professional is learning how to teach yourself. I am going to guide each student through the topics of the semester, but a significant portion of what students take with them to the next class will be things that they learned on their own.

- **Attend every class.** Not only might you miss essential words of wisdom, you might miss a quiz as well.

- **Make mistakes!** This is how you learn how to do something. Don't be discouraged when something goes wrong; it will enlighten you on what not to do in a similar situation.

- **Read the chapter before the corresponding lecture** (see class schedule below)**.** I use class time to write programs that help illustrate the topics mentioned in each chapter. If a student doesn't have minimal knowledge of the concepts that will be covered for that chapter (which are gained by reading the chapter), it will be harder to get a deeper connection to what we are accomplishing in class.

- **Bring your textbook to class.** I know it is heavy and you would rather leave it at home to collect dust, but your book wants to be a part of your academic experience.

- **Arrive to class on time and remain in class until dismissed.** Arriving late and leaving early cause disruptions to the other students in the class and to me. Should you need to leave early for a valid reason, please notify me in advance and sit near the door to limit the disruption. Repeat offenders will be penalized by replacing a previous quiz grade with a zero.

- **No computers in class.** I know, this sounds like crazy talk to say no computers in a computer science classroom, but hear me out. I have seen grades improve by about 10% in classes where I do not allow computers. Without a computer in front of them, students are more engaged during class time. Many students like coding along with me in class, however, this becomes a detriment to the student. While students are copying the code I am writing in class, they are not concentrating on the logic or details behind the code. It is the logic and ideas behind the statements that are more important. All code that is written in class will be posted in eLearning after it is completed.

- **Don't sleep in class.** Let's be honest; programming in C++ is not the most exciting class. Combine that with fatigue from late night gaming and/or study sessions and it is super easy to doze off. Fight it off.

- **Silence all cell phones.** As much as we all might like to sing along to a One Direction ringtone, it causes a distraction in class and distractions eat away time that you paid for to get an education.

WHAT EACH STUDENT SHOULD EXPECT

- **An open environment dedicated to learning.** I want students to feel free to voice their opinions. Oftentimes as we code in class, I will ask students what they would do in a certain situation. I want each student to feel as if he/she can speak freely and also be open for other students to discuss that idea, even if that means that some students will disagree.

- **Class commitment of 12-15 hours a week.** Some students may put in considerably more time every week, especially if students wait until the last minute to work on a project. Students should be prepared to tackle multiple course-related activities each week (e.g. reading the textbook, studying for quizzes/exams, practicing programming, etc.)

- **A quiz could be given at any time.** Quizzes will be given to measure how well you understand the information from each chapter. It is each student's responsibility to be prepared. Quizzes will be based on chapter readings, examples from lecture and/or exercises from the book and will primarily involve coding.

- **Exams focused on application.** I do not create run-of-the-mill multiple choice exams that ask students to regurgitate things from memory. The exams are completely different than anything you have had in any other class (unless you've had me for a previous class). I expect you to apply the knowledge you have learned to the situations on the test. Questions on the test are designed to make sure that you understand what you are doing rather than regurgitating an example from your notes or the textbook.

- **A simulated professional experience.** The projects in this class require you to exercise strategies found in “the real world”. Your logic for a project may force you to learn new techniques that haven’t been discussed in class. You will have to perform code maintenance and improve the efficiency of previously written code. These things offer a small taste of how life might be once you graduate and are given large sums of money by a company seeking your skills.

- **A deep understanding of C++.** Upon completion of this course you will have more knowledge of C++ than you ever expected. You should have peace of mind moving on in your program because you will be fully prepared to tackle what the next course in the sequence will throw at you.

THE INFORMATION YOU REALLY CARE ABOUT

Grading Scale:

98-100 A+	88-89 B+	78-79 C+	68-69 D+	Below 60 F
92-97 A	82-87 B	72-77 C	62-67 D	
90-91 A-	80-81 B-	70-71 C-	60-61 D	

Grade Components: Programming Projects (8-9): 30%
Exams (3): 60%
Quizzes/homework: 10%

General Grade Information:

All grades will be available in eLearning. The Weighted Total column will give you the most accurate information concerning your grade. The weighted total is an approximation of your grade in the class based on the grades currently in eLearning.

I do not curve grades. Assignments are combined into categories so that a low grade for one item will not destroy your grade. There are also opportunities provided to help students who may have done poorly on an assignment or exam.

Grade Disputes:

All grade disputes must be reported within 1 week and resolved within 2 weeks of the grade in question being posted in eLearning.

I am responsible for grading your exams. If you have questions regarding your exam, please contact me. Please note that due to FERPA, I cannot discuss grades via e-mail.

Quizzes, homework and projects will be graded by a TA. Please address any grading concerns you have regarding these grades with the TA. **If you email the TA to dispute a grade, please copy me on the email so that I am aware of the situation and can make sure it is resolved.**

Late Assignments:

Homework is due at the beginning of class and **will not be accepted late**. If your assignment is not submitted at the time of collection in class, it is late and will not be accepted. Please arrive to class on time in order to submit your homework. Homework is not accepted via e-mail unless I have approved the submission.

Projects will be accepted late at the penalty of 5% per hour late (rounded up) for up to 12 hours past the due date/time.

Projects:

Projects will be programming assignments that supplement recently discussed topics and should be completed in two to three weeks.

Projects are individual endeavors and students are not to work in groups on any project. Students are permitted (and I openly encourage students) to discuss the project. Feel free to share ideas on the logic, but **DO NOT SHARE ANY CODE.** All projects will be submitted in eLearning and will be compared for originality. Any projects that are approximate or identical copies will be reported to the Judicial Affairs Committee and I will accept their decision in regards to the grade.

Programming assignments will be graded on a 100 point basis. Not only will your project be graded on proper execution, but also things like efficiency, implementation and documentation. Keep in mind that you always want to write code that is easy to understand and is also easy to maintain. Fewer lines do not necessarily mean a better program. Please use comments liberally.

Homework:

Mini-projects are short coding assignments that can be done in the course of a few hours. These assignments will generally be due the next class meeting.

Quizzes:

Quizzes may be given in class and are generally unannounced. **No make-up quizzes will be given.** Quizzes missed for an excusable reason (with valid documentation) will be exempted. The exemption of a quiz is at the sole discretion of the instructor.

Exams:

Exams will cover chapters as listed below in the tentative course schedule below. Exams will include a variety of question types including multiple choice, short answer and programming exercises. Students are expected to be able to apply knowledge from earlier chapters in conjunction with the tested chapters. Exams are not created to make you feel smart; they are designed for students to demonstrate their understanding of the concepts.

An exam should not be missed except for the most extreme circumstances (such as hospitalization or death of an immediate family member). A make-up exam may be given to students with a valid reason (and documentation) for missing the exam. Otherwise, the missed exam grade will be zero. The allowance of a make-up exam is at the sole discretion of the instructor. Make-up exams must be completed within 48 hours of the date and time of the exam.