CS 1337 Computer Science I
Fall 2016 Syllabus UTD

INSTRUCTOR INFORMATION

Name	Mr. Jason Smith
E-mail address	jws130830@utdallas.edu
Office	ECSS 3.232
Office Phone	972-883-4835
Office Hours	
E-mail:	Emails will be answered 9 AM – 8 PM
On-Campus:	Tu/Th 9:00 – 10:00 AM 11:30 AM – 12:30 PM

COURSE INFORMATION

Course Number	CE/CS/TE 1337.001 and .002		
Credit Hours	3		
Meeting Time	001	M/W	2:30 – 3:45 PM
	002	Tu/Th	1:00 – 2:15 PM
Room	001	ECSS 2.311	
	002	GR 2.530	

DO YOU NEED ASSISTANCE?

E-mail: The easiest way to reach me is via e-mail. I make every effort to respond within a few hours. **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 help 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 CS1336, please visit me, the TA, or the Mentoring Center. The schedule for the Mentoring Center will be released within the first week of classes. Once the Mentoring Center schedule for this semester has been released, an announcement will be posted on eLearning. **If you need help, please make the effort to reach out. We can't help you if we don't know that you need help.**

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

WHAT DO I NEED FOR CLASS?

- **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. You are free to use your own blood, but that gets messy.

- **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 the 7th 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, although you are free to take pictures of pages from my book during office hours.

- **C++ Compiler (Required)**
  - All projects you submit will be compiled with MinGW 4.9.2. You may use any IDE that can utilize MinGW 4.9.2.

    In class, the IDE I will be using is Code::Blocks 16.01. This is a free download for Windows.
    http://sourceforge.net/projects/codeblocks/files/Binaries/16.01/Windows/codeblocks-16.01mingw-setup.exe. This download includes the IDE and MinGW 4.9.2.

  - For Mac users, I recommend using XCode or creating a Windows partition to install MinGW and an IDE. Be advised that there is a Mac version of Code::Blocks, but it has been others have reported it doesn't work with newer versions of the OS X operating system.

  - **If a student uses a compiler other than MinGW 4.9.2 for development, he/she is responsible for verifying prior to submission that the code compiles properly with the stated compiler.** 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 with the compiler that will be used for grading (MinGW 4.9.2).

  - 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.

REQUIRED COURSE INFORMATION SECTION

Course Prerequisite: Prerequisite: CS 1336 with a grade of C or better or equivalent.

Description of Course Content: 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.

Student Learning 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.

WHAT I 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 failing an assignment.

- **Ask questions any time!** During lecture, before/after class, during office hours, at 2 AM after a night clubbing the day before something is due, etc. I really mean any time. I will respond as soon as I can.

- **Take responsibility for your education.** I will treat this course as similar to a professional setting as I can. 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 you toward learning the intricacies of C++.

  Part of being a professional is learning how to teach yourself. I am going to guide you through the topics of the semester, but a significant portion of what you take with you to the next class will be things that you learned on your own.

- **Practice time management skills.** All assignments (homework and projects) are designed to be worked on over a period of days or weeks. I expect that you will work on the assignment a little at a time rather than waiting until a day or two before it is due. Those that procrastinate will find this class to be much harder than it should be and will face the risk of below average grades.

- **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. Programming takes lots of practice and mistakes will always happen. Study the mistakes you made so that you can learn the correct way to do it.

- **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.** If you bought the physical version, I know it is heavy and you would rather leave it at home to collect dust. However, we will refer to the book frequently in class. Your book wants to be a part of your academic experience. Don't prevent your book from having an adventure with you.
- **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.

- **Don't sleep in class.** Let's be honest; programming in C++ is not the most exciting topic. Combine that with fatigue from late night gaming and/or study sessions and it is super easy to doze off. Fight it off. Bring in a caffeinated beverage of your choice, such as Starbucks coffee or a Monster energy drink. Carry an emergency bottle of 5 Hour Energy in your backpack if need be. If students could learn C++ by sleeping, there would be no reason to get out of bed to attend class.

- **Don't pack up your things until class is over.** Most of the time we will go until the very last minute before ending class. Sometimes we might go over by a minute or two if I need to finish a discussion. If students start packing up before we are finished, it makes a lot of noise. That noise might prevent someone from hearing very crucial information such as what the next homework assignment is and when it is due. It also makes me think you are in a hurry to leave the awesome fun party we are having and hurts my feelings.

- **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.

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 10-12 hours a week.** Students should be prepared to tackle multiple course-related activities each week (e.g. reading the textbook, studying for quizzes/exams, practicing programming, etc.). There is a very high correlation between time committed to this class and grades.

- **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 repeating 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++.** My goal is for you to know all of the topics of CS1336 as well (if not better) than me, and I'm going to push you toward that goal. 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:

Maintenance Projects (3)	30% (average of all 3 projects)
Standalone Projects (2)	20% (average of both projects)
Exams (3)	30% (average of all 3 exams)
Quizzes/homework	10%
In-class Projects	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.

Everything else will be graded by a TA. Please address any grading concerns you have regarding these grades with the TA. **When you email the TA with questions about your 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 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. I generally collect the homework within the first 15 minutes of class. Homework is not accepted via e-mail unless I have approved the submission prior to the due date.

Late Projects:

Projects will be accepted late at the penalty of 5% per hour late (rounded up) for up to 6 hours past the due date/time.

Projects: Projects will be major programming assignments that supplement recently discussed topics and should be completed in two to three weeks. Projects are intended to take approximately 15-20 hours to complete; this includes the design, coding and testing process. Waiting until a couple of days before the due date to start the project is a bad idea. Not only does this introduce unnecessary stress into your life, it hardly ever ends well for the student. Most students score poorly on projects that are built in less than three days.

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 Judicial Affairs and I will accept their decision in regards to the grade if they believe that academic dishonesty has occurred.

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.

You are responsible for testing your project thoroughly before submission. I will not give you the test cases that will be used for grading before the project is due.

Homework: Homework assignments are generally short coding assignments that can be done in 1-2 hours. These assignments will typically be due 1 week from the date given.

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 my discretion.

Exams: Exams will cover chapters as listed below in the tentative course schedule. Exams will include a variety of question types including multiple choice, multiple answer and essay questions. Students are expected to be able to apply knowledge from all previous chapters in conjunction with the tested chapters. Exams are not created to make you feel smart; they are designed for you to demonstrate your understanding of the concepts. A high score on an exam exhibits a deep understanding of the topics.

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.

ARE WE THERE YET?

All dates are subject to change at the discretion of the instructor

Date (001)	Date (002)	Topic	Reading Assignments
8/22	8/23	Introduction to CS 1337 Introduction to Code::Blocks	Read Chapter 8
8/24	8/25	Intro to C Searching and Sorting Arrays	
8/29	8/30	Searching and Sorting Arrays	Read Chapter 12 (omit 12.7, 12.8)

8/31	9/1	Searching and Sorting Arrays Advanced File I/O	
9/5	---	**LABOR DAY (NO CLASS 001)**	
9/7	9/6	Advanced File I/O	
9/12	9/8	Advanced File I/O Recursion	Read Chapter 19
9/14	9/13	Recursion	
9/19	9/15	Recursion	**Project 1 Due (9/15)**
9/21	9/20	Exam 1 (Chapters 8, 12, 19)	Read Chapter 9
9/26	9/22	Pointers	
9/28	9/27	Pointers	
10/3	9/29	Pointers Structured Data	Read Chapter 11
10/5	10/4	Structured Data	**Project 2 Due** Read C. 17.1 & 17.2 Review Linked List PDF
10/10	10/6	Linked Lists	
10/12	10/11	Linked Lists	
10/17	10/13	Enumerated Data Types Unions	Read Chapter 10
10/19	10/18	Characters, Strings and the String Class	
10/24	10/20	Characters, Strings and the String Class	
10/26	10/25	Characters, Strings and the String Class	**Project 3 Due**
10/31	10/27	Exam 2 (Linked Lists, C. 9, 10, 11)	
11/2	11/1	Introduction to Classes	
11/7	11/3	Introduction to Classes	Read Chapter 13
11/9	11/8	Introduction to Classes	
11/14	11/10	More About Classes	
11/16	11/15	More About Classes	**Project 4 Due**
	11/17	More About Classes	Read Chapter 14
11/21 – 11/25		**FALL BREAK**	
11/28		More About Classes	
11/30	11/29	Inheritance, Polymorphism & Virtual Functions	Read Chapter 15
12/5	12/1	Inheritance, Polymorphism & Virtual Functions	
12/7	12/6	Inheritance, Polymorphism & Virtual Functions	
12/10			**Project 5 Due**
Finals Week		Exam 3 (C. 13, 14, 15)	

Important Dates:

August 22	Classes start
September 5	Labor Day
September 7	**Census Day**
September 15	Project 1 due
September 20-21	Exam 1
October 4-5	Project 2 due
October 25-26	Project 3 due
October 27	**Last Day to Withdraw**
October 27	Exam 2 (002)
October 31	Exam 2 (001)
November 15-16	Project 4 due
December 10	Project 5 due
TBA (Finals Week)	Exam 3

The above schedule is subject to change at the discretion of the Professor.

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