

CSCI/CMPE 1370-01

Engineering Computer Science I

Course Information

Instructor:	Tim Wylie
Contact:	Office: ENGR 3.287 Phone: 956-665-2577 Email: wylie@utpa.edu
Office Hours:	TR 9:00am - 11:00am
Schedule:	Lecture: MWF, 10:45am - 11:35am, ENG 1.272 Lab: F, 1:10pm - 3:50pm, ASB 2.110
Textbook: <i>(optional)</i>	Malik, D. S., C++ Programming: Program Design Including Data Structures. 7 th edition, 2014. ISBN-13: 978-1285852751
Course Website:	http://faculty.utpa.edu/wylie/CSCI1370

Course Description

CSCI/CMPE 1370 - Engineering Computer Science I: An introduction to computer science and computer engineering. The fundamentals of a high-level programming language will be introduced. Methods of problem solving, techniques of algorithmic development and concepts of procedural and object-oriented programming will be emphasized. Fulfills Computer Literacy Core Requirement. Corequisite: CSCI 1170.

CSCI/CMPE 1170 - Engineering Computer Science I Laboratory: The course includes hands-on instruction and laboratory exercises in developing programs written in a high-level object oriented programming language applying the principles taught in the CSCI 1370 lecture course. Co-requisite: CSCI 1370.

Course Topics

This course is an introduction to Computer Science and is taken as the first course for Computer Science majors and minors. This is also the first software course for Computer Engineering majors. It focuses on techniques of problem solving and algorithmic design, and includes lab experiences in design and implementation of these algorithms in C++. Topics in C++ include:

- Data types, variables and assignment

- Interactive input/output statements
- File input/output statements
- Selection and loop statements
- Functions
- Pointers
- One and two dimensional arrays
- Simple sorting and searching algorithms
- User-defined data types
- Structured data types
- Data abstraction and classes
- Characters, strings, and the string class

Course Objectives

Students will learn:

1. How to design and write computer programs to solve basic problems:
 - a) How to analyze a problem and develop an appropriate algorithm to solve it
 - b) How to implement algorithms by writing C++ code
 - c) How to compile and link code into a working program
 - d) How to use testing and debugging strategies to identify and fix program faults
2. How the programming language, libraries, and development environment each impact the way programs are written.
3. How a program can be written many different ways, and why some are better than others:
 - a) How different algorithms meet different requirements
 - b) How to evaluate, use, and modify existing algorithms
4. How to write and document your code so that it is useful to other programmers.

Learning Outcomes

Detailed Learning Outcomes

Upon completion of this course, students should be able to:

1. Be proficient with the programming environment and understand the basic aspects of program translation
2. Analyze a programming problem and develop a solution algorithm
3. Use the syntax and semantics of a higher-level language to implement their solutions to programming problems, including the correct use of the following:
 - a) Variables and assignment to variables
 - b) Primitive types such as integer, character, and floating-point variable types
 - c) Commonly-used built-in reference types, e.g., single-dimensional arrays, strings
 - d) Declaration of constants and variables
 - e) Assignment, logical, and arithmetic operators
 - f) Local and global variables

- g) Selective control structures (e.g., if, nested if, switch)
 - h) Iterative control structures (e.g., for, while)
 - i) Functions (user-defined and predefined)
 - j) Parameter passing involving both primitive types and reference types
 - k) Arrays and other structures (such as records)
4. Document their solutions
 5. Apply problem-solving strategies to design a solution to a problem similar to ones seen before
 6. Design simple ADTs to solve a problem similar to one seen before
 7. Apply testing and debugging strategies, including black-box and white-box testing, test drivers, stubs, and test suites, to identify software faults
 8. Formulate complex arithmetic expressions involving operators of differing precedence and associativity, and understand the order of evaluation of sub-expressions
 9. Formulate complex logical expressions involving multiple and/or/not combinations
 10. Implement nested if statements
 11. Demonstrate proficiency in the use of logic and arithmetic operators and their precedence
 12. Use simple I/O to read and write character and numeric data to and from files, keyboard, and display
 13. Use simple sorting and searching algorithms
 14. Use standard documentation to determine the use of an unfamiliar class or method
 15. Use teamwork roles and methods in the classroom

ABET Outcomes

1. An ability to apply knowledge of computing and mathematics appropriate to the discipline
2. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
3. An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
4. An ability to function effectively on teams to accomplish a common goal
5. An understanding of professional, ethical, legal, security and social issues, and responsibilities
6. An ability to use current techniques, skills, and tools necessary for computing practice
7. An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
8. An ability to apply design and development principles in the construction of software systems of varying complexity

Course Requirements

Attendance: Students are expected to attend every lecture and lab period for this course. If you know you are going to be unable to attend, contact the instructor beforehand to arrange for your absence. If you do not do so, you will not be given extensions or make-ups for any exercises, exams, or other activities.

Exercises: There will be a variety of in-class exercises during this course, including problem solving and small coding problems. They will be completed during class time and turned in at the end of class. You are expected to attend every lecture and take part in these exercises.

Assignments: The assignments will involve writing, compiling, and running your own programs. These assignments will be announced in class and posted in Blackboard. All programs for this course will be built using the Microsoft Visual C++ environment.

Labs: You must be registered for the corresponding CSCI/CMPE 1170 lab section. The lab assignments will be done during the weekly CSCI/CMPE 1170 lab period. If you are enrolled in only one of CSCI/CMPE 1370 or 1170, please contact the instructor.

Exams: There will be three exams for this course: two mid-term exams and one final exam. The material in this course is naturally cumulative, with each weeks topics building on all the prior material. While each of the two mid-term exams will focus on the most recent material, you are responsible for all of the material that has been covered in class up to that point. The final exam is cumulative, including questions focusing on the most recent material, as well as questions that specifically ask about material covered earlier in the course.

Scoring and Grading

CSCI/CMPE 1370 and 1170 are co-requisite courses. The purpose of the lab section is to increase your hands-on experience with the material, and to provide you with another avenue to demonstrate what you have learned. You will receive the same grade for both courses. That grade is calculated based on the total work performed in both CSCI/CMPE 1370 and 1170.

Grade Breakdown:		Final Grade:	
1370 Exercises	10%	90%-100%	A
1370 Assignments	20%	80%-89%	B
1370 Exams	40%	70%-79%	C
1170 Labs	30%	60%-69%	D
	—	50%-59%	F
Total possible score (max):	100%		

Note: Grades may be curved to reflect the overall performance of the class.

Course Schedule

This is a rough course schedule to give you an idea of topics and pacing. The actual course schedule is likely to change and will be kept up to date on the course website.

Week 1-2: Basic program organization and execution, data and variables, user input

Week 3-5: Conditional and iterative execution, simple boolean logic

Week 6-8: User-defined functions and parameters

Week 9-10: Arrays and structured data

Week 11-12: Structured data

Week 13-14: Classes and object-oriented programming

Week 15: Pointers and dynamic memory

Course Policies

Drop Class Policy: It is the students responsibility to request a Drop of the class. No “Drops” will be granted after the official drop or withdraw date. The student must drop both CSCI/CMPE 1370 and 1170. The instructor can discuss a student’s progress and standing in the course at any time the student wants.

Computer Use Policy: Please read and be aware of University policies for computer use, which can be found at: <http://www.utpa.edu/policies/UTPAAcceptableUse.pdf>

Late Work Policy: Labs and exercises will not be accepted late. Assignments must be turned in at the specified time on the given due date. Afterwards, the penalties are as follows:

- Within 24 hours late will lose 10%
- Within 48 hours late will lose 20%
- More than 48 hours late will receive no credit

Make-up Policy: No make-up exams will be given except for university sanctioned excused absences. If you need to miss an exam, it is your responsibility to contact me before the exam, or as soon after the exam as possible. Missing an exam without an approved (by the university or me) excuse will result in a zero.

Academic Integrity Policy: The University expects a student to maintain a high standard of individual honor in his/her scholastic work. Unless otherwise required, each student is expected to complete his or her assignment individually and independently. Although study together is encouraged, the work handed in for grading by each student is expected to be his or her own. Collaborations or help with other students should be noted on the assignment.

Any form of academic dishonesty will be strictly forbidden and will be punished to the maximum extent. Copying an assignment from another student in the class or obtaining

a solution from some other source will lead to disciplinary action. Allowing another student to copy ones work will be treated as an act of academic dishonesty, leading to the same penalty as copying. For more information, refer to <http://utpa.edu/hop/> for the student code of conduct.

Course Evaluation: Mandatory Course Evaluations period (Apr 15 – May 6): Students are required to complete an ONLINE evaluation of this course, accessed through your UTPA account (<https://my.utpa.edu/>); you will be contacted through email with further instructions. The evaluation window closes at 11:59 pm on May 6th, the last day of Spring classes. Students who complete their evaluations by May 6th will have priority access to their grades.

Note to Students with Disabilities

Students with disabilities are encouraged to contact the Disability Services office for a confidential discussion of their individual needs for academic accommodation. It is the policy of the University of Texas-Pan American to provide flexible and individualized accommodation to students with documented disabilities that may affect their ability to fully participate in course activities or to meet course requirements. To receive accommodation services, students must be registered with the Disability Services office (DS), University Center #108, 665-7005 or disabilityservices@utpa.edu.

Honors Credit

For those enrolled in CSCI/CMPE 1378 and 1178 for Honors credit, there will be one additional assignment. The assignment will be based on a topic the student is interest in, will be given around mid-term, and will be due at the end of the semester. The assignment does not affect the grade given in the class, but must be completed or an incomplete will be given for the course.