

CSCI 152
Spring 2005

Programming Fundamentals II

Instructor: Derek Harter
Office: JOUR 208
Phone: 903-886-5402
Email: dharter@online.tamu-commerce.edu
Web Page: <http://faculty.tamu-commerce.edu/dharter/tamu/classes/2005spring/csci152>
Office Hrs: M-W 9:00am – 11:00am
Class: Section 002: T 4:30 – 7:10pm
Section 003: MW 12:30 – 1:45pm

Course Description: This course is a continuation of CSci 151. The skills in problem solving and programming acquired in the previous class will be reinforced and enhanced. The concepts of object-oriented programming and design will be introduced. Specific topics include multi-dimension array processing and applications, sort and search algorithms, file processing, string processing, structures, classes, inheritance, pointer variables.

Goals: After completion of this course, you should be able to code application programs using arrays, sorts and searches, strings, user-defined files, and structures. You should be able to create and use classes to represent abstract data types.

Prerequisite: CSCI 151 or other course covering basic concepts of the C or C++ languages.

Text: C++ Programming: From Problem Analysis To Program Design (Second Edition)
by D.S. Malik
ISBN: 0-619-16042-X

Evaluation:

Your grade for the course will be based on the following approximate numbers and percentages:
54% 6 quizzes (the 6 highest grades of 7 quizzes taken)
22% comprehensive final exam
24% 5-6 programs

Format for the quizzes will typically be approximately half coding (usually small segments like functions or parts of functions) and half analyzing the effects of executing code (describing output, completing diagrams, short essay, etc).

Format for the written final exam will be mostly multiple-choice, true/false, and short answer with minimal coding.

A **study guide** will be provided for the final exam listing topics to be covered (or omitted) and recommending selected problems from the text from which the majority of exam questions will be derived. Answers (which are not provided in the text) for these problems will be available on my web page.

Letter grades will be assigned according to the following scale:

- A at least 90% of the total points
- B at least 80% but less than 90% of the total points
- C at least 70% but less than 80% of the total points
- D at least 60% but less than 70% of the total points
- F less than 60% of the total points

You must earn an A on your own. Lower borderline grades may be affected by your class participation and behavior, the pattern of your grades, and the class grade distribution.

For details of program requirements, see the accompanying handout **General Policy for Programming Assignments**.

Educator: The web page for our class this semester is:

<http://faculty.tamu-commerce.edu/dharter/tamu/classes/2005spring/csci152>

The course syllabus, input data files, sample programs, lecture slides, programming assignments and other miscellaneous handouts and useful materials may be found here. In addition, I will be using the online Texas A&M Educator system this semester to disseminate course materials and collect assignments. You will need to sign up and register for an Educator account by going to:

<http://online.tamu-commerce.edu>

If you don't already have an account, follow the instructions on this page to get an account. All students will then need to register for the CSci 152 course in the Educator system. I will send all e-mail and announcements using your educator accounts, and all submissions of programming assignments will be via Educator. Please check you account e-mail and announcements frequently, and use my dharter@online.tamu-commerce.edu e-mail address to send correspondence to me regarding the class.

Makeups: If you miss a quiz, that will be the low grade (0 points) which will be dropped. If there is a possibility you might not be able to take the final exam at the scheduled time, please contact me as soon as you recognize that you have a problem.

Final: The dates and times of the final for our sections are listed in the course schedule below. University policy mandates that we give finals only during the finals week and only on the dates and times assigned for our class. Finals can not be given at special times to accomodate end-of-semester travel plans or other similar circumstances. Please keep the date and time of in mind when planning your travel and end-of-semester activities.

Attendance: You are responsible for everything covered in all class meetings. Class attendance will be taken at the beginning of each class.

Drops: If you cannot complete the course, please don't forget to drop. If you are making an obvious effort in the course at the time you drop (still attending class, attempting program assignments), you may drop passing no matter what your actual grade might be. If you just disappear, your grade will be whatever you have actually earned at the end of the semester.

Students requesting accommodations for disabilities must go through the Academic Support Committee. For more information, please contact the Director of Disability Resources & Services, Halladay Student Services Bldg., Room 303D, (903) 886-5835.

"All students enrolled at the University shall follow the tenets of common decency and acceptable behavior conducive to a positive learning environment." (See Student's Guide Handbook, Policies and Procedures, Conduct)

All students should be aware that plagiarism is a serious offense. This is true not only of written essays but also of work written in artificial computer languages such as C++. Copying code for assignments from other students or the internet is not allowed. You may discuss general aspects and strategies of the coding assignments with one another, but you must code the programming assignments on your own.
--

TENTATIVE SCHEDULE

Week	Date	#	Topic / Activity	Reading
1	17 Jan	1	Introduction, Preliminary Quiz	
		2	Review of selected 151 topics	Ch 1-8
2	24 Jan	1	Quiz 1 (over review material)	
		2	One-Dimensional Arrays and C-strings Two-Dimensional Arrays	Ch 9:423-450 Ch 9:450-468
3	31 Jan	1	Applications of Arrays: List Processing, Searching and Sorting	Ch 10:497-515
		2	Sequential search and binary search Programming Assignment #1 Due	Ch 10:515-523
4	7 Feb	1	Quiz 2	
		2	Programming Assignment #2 Due	
5	14 Feb	1	Programming Fundamentals and Computer Security	Handout
		2	Programming Fundamentals and Computer Security	Handout
6	21 Feb	1	Records (structs): Accessing, assignment, comparison, I/O Programming Assignment #3 Due	Ch 11:553-571
		2		
7	28 Feb	1	Quiz 3 Classes	Ch 12:595-605
		2	Member functions, accessors and mutators	Ch 12:605-617
8	7 Mar	1	Constructors and Destructors	Ch 12:617-630
		2	Data abstraction and Information Hiding Programming Assignment #4 Due	Ch 12:630-647
9	14 Mar	1	Spring Break Mar 14-19	
		2		
10	21 Mar	1	Programming Fundamentals and Computer Security	Handout
		2	Programming Fundamentals and Computer Security	Handout
11	28 Mar	1	Quiz 4	
		2	Inheritance and Composition: Inheritance Programming Assignment #5 Due	Ch13:673-705
12	4 Apr	1	OOD & OOP	Ch13:705-708
		2		
13	11 Apr	1	Quiz 5 Pointers, Classes, Virtual Functions and Dynamic Variables	Ch 14:741-766
		2	Shallow vs. Deep Copy, Classes and Pointers	Ch 14:766-781
14	18 Apr	1	Inheritance, pointers and virtual functions	Ch 14:781-791
		2		
15	25 Apr	1	Quiz 6 Overloading and Templates: Why operator overloading	Ch 15:801-803
		2	Operator overloading: syntax, this pointer Programming Assignment #6 Due	Ch 15:803-810
16	2 May	1	Overloading operators	Ch 15:810-844
		2	Quiz 7	
17	9-13 May		Final Exam Sec 002: Tuesday, May 10 4:30 – 6:30 pm Sec 003: Friday, May 13 8:00 – 10:00 am.	

Course Objectives

1. Be able to use one-dimensional arrays.
 - a. Be able to declare, initialize, and manipulate individual elements of a one-dimensional array.
 - b. Be able to pass one-dimensional arrays to functions.
2. Be able to use at least one (preferably at least two) sorting technique(s) to rearrange data in an array.
3. Be able to search an array using both linear and binary searching techniques.
4. Be able to use multiple-dimensional arrays.
 - a. Be able to declare, initialize, and manipulate individual elements of a two-dimensional array.
 - b. Be able to pass two-dimensional arrays to functions.
 - c. Be able to pass one row of a two-dimensional array to a function.
5. Be able to use traditional C-style strings.
 - a. Be able to declare and initialize C-strings.
 - b. Be able to manipulate C-strings one character at a time.
 - c. Be able to assign and compare C-strings and to determine their length.
 - d. Be able to pass C-strings to functions.
6. Be able to use the ANSI C++ string class.
 - a. Be able to declare and initialize strings.
 - b. Be able to manipulate strings one character at a time.
 - c. Be able to assign and compare strings and to determine their length.
 - d. Be able to pass strings to functions and to have a function return a string.
 - e. Understand the use of logical operators with string class instances
7. Be able to use address variables.
 - a. Be able to declare a pointer variable, initialize it, and dereference it.
 - b. Be able to pass a variable by address using a pointer parameter.
 - c. Understand the relationship between pointers and arrays.
 - d. Be able to use pointer arithmetic to access elements of an array.
 - e. Be able to declare, initialize and use a reference variable.
 - f. Be able to use a reference variable as a function parameter.
 - g. Be able to use a reference variable as the return data type of a function.
 - h. Be able to pass an argument to a function that uses a reference parameter to accept the argument's address.
 - i. Understand the differences between reference variables and pointer variables.
8. Be able to use structs.
 - a. Be able to declare a struct and manipulate its fields.
 - b. Be able to declare an array of structs and manipulate the elements.
 - c. Understand the differences between structs and classes.
9. Be able to use classes.
 - a. Be able to create and use class objects.
 - b. Be able to declare a class and control access to class data members.
 - c. Be able to create and use constructor and destructor functions.
 - d. Be able to create and use friend functions.
 - e. Be able to overload functions.
 - f. Be able to overload operator functions.
10. Be able to design and code a program which includes a user-created class.