

Syllabus for CMPS 250 Fall 2007

Introduction to Computer Science, Data Structures & Software Design

Instructor: Frank Ducrest

Lecture: 9:00 – 9:50 AM, ACTR 106, MTWRF

Instructor: Office: ACTR 215

Email: fdd@louisiana.edu

Web Pages: access Moodle via ULink

Office Hours: on the Moodle web site, select Resources, then Instructor

Office Phone: 337-482-5001

Course Description:

Introduction to Unix and Linux. Introduction to C++. Integrated software engineering principles, fundamental data structures and algorithm design and development. Focus on requirements, specifications, design and testing. Fundamental data structures will include arrays, linked lists, stacks and queues. Prerequisites: MATH 110 with a grade of C or better. Co-requisite: Math 270.

Course Goals

- To gain experience programming in C++.
- To gain experience in using Linux and Unix operating systems.
- To gain experience with object oriented development and related design issues.
- To develop a working knowledge of multi-dimension arrays, abstract data types lists, stacks, and queues, searching, C++ templates, pointers and dynamic structures.

Textbook: C++ Programming: Program Design Including Data Structures, 3rd Edition; by D. S. Malik, Publisher – Thomson Course Technology; ISBN-10: 1-4188-3640-0, ISBN-13: 978-1-4188-3640-5

Grading Scale:

at least 90% or above	A
at least 80% but below 90%	B
at least 70% but below 80%	C
at least 60% but below 70%	D
below 60%	F

Point Breakdown:

Assignments and projects	25%
Exam #1	25%
Exam #2	25%
Final Exam	25%

Your overall average must be 70% or above and you must have at least a 65% exam average and a 65% average on other assignments (review, labs, assignments, projects) to qualify to pass this class with a C or greater.

Blue Tooth Devices, CD/DVD Players, iPods, Cellphones and Other Distractions:

Personal electronic entertainment and communications devices are certainly a lot of fun and even necessary at times. However, they are at best inappropriate for use during class. Use of such devices during class is not permitted.

Email:

When necessary, the instructor will communicate with students enrolled in the class via postings on the class Moodle site and through UCS email. It is the student's responsibility to check both regularly and to insure that his or her UCS email account is able to receive email.

Attendance:

- a) A sign-in sheet will be used to take roll at each class meeting. It is the student's responsibility to sign the sheet.
- b) You are responsible for all missed work, regardless for the reason for an absence. You are responsible for getting any notes or material you may have missed.
- c) Absences for University sponsored events will be treated like any other absence.
- d) Students missing an excessive number of classes may be dropped from the class role.

Collaboration:

Certain types of assistance are inappropriate for assignments and projects in this class. Read the collaboration policy given to you at the beginning of the semester. Each assignment and project will have specific information.

Assignments, Projects and Exams

- e) Your code must compile with the assigned compiler. Your projects will be compiled with the assigned compiler before being graded. Do NOT rely on other compilers than the one assigned.
- f) To submit your projects, the code must be in your class directory. Make sure that you follow the naming convention established for each assignment or project. Remember that Unix is case sensitive; that is, upper and lower case letters are different. You may submit a project as many times as you like before the deadline. Note that the latest submission overwrites previous submissions. If you have an emergency and cannot finish a project on time, contact the instructor prior to the deadline, not after. Otherwise, late assignments or projects will not be accepted.
- g) You have ONE WEEK in which to question your grade on an assignment, project or exam starting from the date it is returned. After that time there will be no further discussion or review of the grade.
- h) Assignments and projects are due at the beginning of class one week after they are assigned unless otherwise indicated in the assignment specifications. Neither projects or assignments will not be accepted late.
- i) Once you submit a project, do not modify files or change the time stamp of the files until you have resolved any questions about the grade you earned for the project. Do remove the binary files created by the compiler, as you will not have room in your class account to keep all examples, assignments and projects in binary form.
- j) For projects, you will submit a manila containing documented printouts of your source code. Do not staple or clip the sheets together.
- k) In addition to the standard documentation each source file for projects must include the following certification of authenticity comment at the very beginning of the file.

Certification of Authenticity: {Pick one of the following: }

I certify that this assignment is entirely my own work.

{or}

I certify that this solution to the assignment is entirely my own work, but I received some assistance from {name}.
Description of the type of assistance. (For example, if you consulted a book, and your solution incorporates ideas found in the book, give appropriate credit; that is, include a bibliographical reference.)

Etceteras:

- 1) Incomplete Grades are given in this course only under unusual / special circumstances.
- 2) You must have a pictured ID with you in order to take your exams and collect printouts in the main lab.
- 3) The UL at Lafayette Computer Science Department Policies regarding attendance, collaboration and auditing classes are in effect. (Sheet to be provided.)
- 4) Students are required to read the Academic Honesty section in the UL at Lafayette Undergraduate Bulletin or the UL at Lafayette Graduate Bulletin.
- 5) Exams at locations other than the lecture room: Students with special needs should report to Services for Students with Disabilities. If you take your exams through that office, your appointments must be made for the same time and day as the class takes their exams.

Tentative Course Coverage

I. CMPS 150 Course Material

A. Introduction

- i. syllabus, use of text, course coverage, grading, Unix / Linux, working remotely

B. Computers, Programming Languages and the Basic Elements of C++

- i. Reading: Chapters 1 and 2

C. Input and Output

- i. Reading: Chapter 3

D. Control Structures: Selection

- i. Reading: Chapter 4

E. Control Structures: Repetition

- i. Reading: Chapter 5

F. User Defined Functions

- i. Reading: Chapters 6 and 7

G. User Defined Simple Data Types, Namespaces, and the string Class

- i. Reading: Chapter 8

H. Arrays and Strings

- i. Reading: Chapter 9

I. Records

- i. Reading: Chapter 10

***** Exam # 1 *****

II. CMPS 260 Material

Reading Key:

- **** *“thorough understanding required”* *Your understanding of the material is very important for your success in the course. This material will be the basis of future material. Expect to use everything in this section in lab and projects.*
- *** *“understanding required”* *This is an example of the type of code that you will be expected to understand and create. If you cannot follow this type of code, the test questions and assigned projects will not be comprehensible.*
- ** *“basic understanding required”* *You will write or use code in lab based on these concepts, then write or use code in your projects based on these concepts.*
- * *“for expanded knowledge”* *Material designed to give an expanded view of a subject and provide insight and support to your understanding of C++ programming or object orientation. More than background material, but not a central component of the class. Will affect code that you will use, but will not be central to the code you will write.*

A. Classes, Object Oriented Programming and Data Abstraction

- i. Reading: Chapter 11
 - a) read all ****
 - b) review programming example Candy Machine ***
- ii. Reading: Chapter 12
 - a) read all for expanded knowledge of object orientation *

B. Pointers, Classes, Virtual Functions, Abstract Classes and Lists

- i. Reading: Chapter 13
 - a) pages 734 – 757, understanding required **
 - b) pages 758 – 786, read for expanded knowledge of object orientation and pointers *
 - c) pages 786 (Array-Based Lists) – 804 **

C. Overloading

- i. Reading: Chapter 14
 - a) pages 818 – 880 **

D. Templates

- i. Reading: Chapter 14
 - a) pages 881 – 892 **

E. Exceptions

- i. Reading: Chapter 15
 - a) pages 908 – 939 *

F. Recursion

- i. Reading: Chapter 16
 - a) pages 946 – 971 *

***** Exam # 2 *****

G. Linked Lists, Ordered Lists, Doubly Linked Lists

- i. Reading: Chapter 17
 - a) pages 982 – 1018
 - reading, concepts *****
 - code **
 - b) pages 1018 (Ordered Linked Lists) – 1045 **
 - c) programming example Video Store ***

H. Stacks

- i. Reading: Chapter 18
 - a) pages 1076 – 1112
 - reading, concepts *****
 - code **

I. Queues

- i. Reading: Chapter 18
 - a) pages 1131 - 1149
 - reading, concepts *****
 - code **

J. Design

- i. Reading: Programming Documentation Standard, parts II, II.1, II.2, II.3 ***

K. Sequential and Binary Search, Asymptotic Notation (Big O Notation)

- i. Reading: Chapter 19
 - a) pages 1184 - 1204 **

***** Final Exam *****