

# **CSCI 455: Programming Systems Design**

Course Information & Syllabus

**Fall 2009**

**WEBSITE:** <http://scf.usc.edu/~csci455/>

**Instructor:** Dr. K. Narayanaswamy

**Office:** Off Campus

**Email:** [swamy@cs3-inc.com](mailto:swamy@cs3-inc.com)

**Phone:** (310) 337-3013

**Office Hours:** Available before/after class or by appointment

## **Course Overview:**

CSCI 455 teaches you about programming in the C++ language. It is the first programming course taken by senior students who need to understand the principles of programming over the duration of a semester. Therefore, the course will be fast-paced and will include “soup to nuts” coverage of programming in C++. Specific topics include:

1. C++ programming basics including the mechanics of statements (assignment, conditional, iteration), data types, variables, input/output functions, user-defined functions, arrays, structures, and other critical basic programming concepts.
2. Principles of software development such as encapsulation and inheritance in C++ and how to program with C++ classes. These principles are important as you evolve into writing larger scale programs. These principles are now used in building component-based software systems, which is the only way to develop and manage large systems.
3. We will provide a brief overview of the Standard Template Library (STL), which will give you powerful constructs to program data structures and algorithms in a much more productive manner than possible when you write these programs manually.

At the end of this course, students should have familiarity with the concepts and techniques associated with programming in general, and how to use them in writing C++ programs appropriate for the classes to come in the CS curriculum.

## Course Schedule:

This class meets **Thursdays between 6:30 p.m. to 9:10 p.m.** Each meeting will be divided into two lectures of roughly 75 minutes. There will be a short break of around 10 minutes between the two lectures. Each of the two lectures will cover appropriate technical topics. Please follow the provided Course Syllabus to know what the lecture topic will be, relevant readings, and assignment schedule. You are completely responsible for making sure that you stay current on all course developments and on knowing how well the class is keeping up with the prescribed syllabus.

There are no Lab sessions in this course. For those who need a little extra help with assignments it is critical that you interact with the TA and formulate a good working relationship with him or her. Further, this course has numerous programming assignments and it is important for you to use the staff resources to get all the assistance possible to finish your assignments on time.

## Text Books:

There is a required textbook for this course, and two recommended or optional books. We provide guidance about the book chapters for reading purposes. No assignments will be set from the text book, however. *Before* a session, skim the portions of the book, if you have time. Certainly, *after* a class session, it is important to read the topic more carefully, and understand the discussions.

- **Required Text:** *C++ Programming: Program Design, Including Data Structures*, D.S. Malik; Course Technology, Thomson Learning (If you already have a good reference book for C++, that will be adequate).
- **Reference Book:** *C++ Primer* (3<sup>rd</sup> Edition), Stanley Lippman, Addison Wesley
- **Reference Book:** *Effective C++*, Meyers, Addison Wesley

## Computing Resources:

All program assignments should be submitted using your student account on University resources – specifically the “submit” program. **Programs will be tested only on University resources.** Wherever and however you develop your programs, **test them on the University’s UNIX infrastructure** before submission.

In particular, all C++ programs must compile and work using SUN workstations and the GNU C++ compiler ( *g++* ). You may use other platforms to help you develop and debug the program, but the eventual submission will have to be done with the “submit” program using the UNIX infrastructure. **NO EXCEPTIONS TO THIS RULE AND NO EXCUSES!!**

## Course Assignments:

There will be a combination of programming assignments (roughly 10 over the course of the semester), a mid-term examination, and a final examination.

1. **Examinations:** All examinations are **open-book and open-notes**. The mid-term examination will include all topics done up to that point, and final examination will be comprehensive. The examinations will be a mixture of objective questions and some questions requiring analysis or synthesis of small code segments. Unless there is a **documented dire emergency**, examinations cannot be made up. Please be mindful of this in planning your schedule.
2. **Programming/LAB Assignments:** You will have 10 programming assignments during this course. Each programming assignment will be posted by Friday at 12:00 p.m. Each assignment will be discussed in the lecture that Friday, and is generally due a week from the following Sunday night at midnight. Each programming assignment is worth 5 points on the final grade. Late policies: submitted the following day: 1 point off; submitted 2 days late: 3 points off; no late assignments accepted more than 2 days late without documented medical or other emergencies.

Program assignment grades will be assigned using the following basic guidelines: correctness 75 % and style 25 % (including program design, program structure, and simple but effective documentation). Specific criteria will be spelled out in detail for each assignment.

## Course Grades:

Your overall course grade will be based on your performance in the different course assignments. The relative weight of each course assignment is as follows:

- **Programming/LAB Assignments** ( 5 points X 10) = 50 % of your grade
- **Mid Term Examination** = 20 %
- **Final Examination** = 30 % (to be held during finals week in the designated room)

## Academic Integrity:

Cooperation on programming assignments is healthy, unavoidable, and even encouraged to the degree that you learn from such interactions and that real-world programming indeed involves teams of people working together. But, **plagiarism is explicitly forbidden**, and serious action will be taken if students are caught cheating on programming assignments or examinations. Copying of any or all of another person's code is an example of cheating – regardless of size of code copied or the amount of functionality involved. Discussing design issues or getting help in understanding an algorithm from someone is certainly not a problem. Everyone should do his/her own thinking and programming.

If you have any question at all about any action of yours with respect to ethics, we suggest strongly that you check with the TA or instructor **before** taking the action, and/or attach information to your submission to identify the collaboration or help that you have had.

## CSCI 455 Syllabus

Week	Date	Course Topics	Readings	Activity
1	8/27	C++ Program Basics: statements, variables, expressions, operators, input, and output. Data types including numerics and string	Chapters 1, 2, and 8	
2	9/3	Simple file input and output If statement and logical expressions Iteration: “while/do” “do/while” statement	Chapters 2, 3, 4, and 5	P1 posted 9/3
3	9/10	Motivation for array data type 1-dimensional array and “for” iteration User-defined functions	Chapters 5 and 9	P1 due 9/13 P2 posted 9/10
4	9/17	User-defined functions and program design Passing Parameters by Value & Reference Functions that return values	Chapter 6 and 7	P2 due 9/20 P3 posted 9/17
5	9/24	Multi-dimensional arrays More on string data type Enumerated data type and typedef statement	Chapter 5, 8, and 9	P3 due 9/27 P4 posted 9/24
6	10/1	Motivation for Records; struct construct How to define and use struct in programs Programming collections with arrays	Chapter 11	P4 due 10/4 P5 posted 10/1
7	10/8	Algorithms and complexity issues Search algorithms and complexity Sort algorithms and complexity	Chapter 18	P5 due 10/11 P6 posted 10/8
8	10/15	<b>Midterm Examination (Location of midterm to be announced)</b> Recursive functions and algorithms that have recursive structure	Chapter 10, 18	Midterm P6 due 10/18
9	10/22	Overloading of functions Principle of Encapsulation/Class construct Public/Private members of a class	Chapter 15 Chapter 12	P7 posted 10/22
10	10/29	Constructors/destructors Programming with classes How to create new classes	Chapter 12	P7 due 11/1 P8 posted 10/29
11	11/5	Pointers and their uses Linked list representation of collections Algorithms that use linked lists	Chapter 14, 16 and 17	P8 due 11/8 P9 posted 11/5
12	11/12	Principle of Inheritance & evolution Public/Private/Protected members Dynamic binding and virtual functions	Chapter 13	
13	11/19	Standard Template Library and uses vector and list templates STL iterators and using them in algorithms	Chapter 21	P9 due 11/22 P10 posted 11/19
14	12/3	Additional STL algorithms How to program applications with aggregation/composition	Chapter 21	P10 is due 12/6 <b>NO LATE SUBMISSION!!</b>

**FINAL EXAM:** 12/10/2009 ; 7:00 p.m. to 9:00 p.m. **Location will be announced once it is determined...**