

CSC 441

Object Oriented Programming

Section F1: Ford on-site BSCS

Syllabus

Fall 2002-03

Number and Title of Course:

CSC 441 Object Oriented Programming

Catalog Description of Course:

This course will present the fundamentals of object-oriented programming techniques, including encapsulation, type-extensibility, inheritance, and polymorphism. The implementation language will be C++. The course will begin with a description of that part of C++ that is simply part of C (called the kernel language) and then present objects and their implementation in C.

Course Prerequisite:

This is an advanced programming class. It is primarily intended for students who have at least one year of a college level programming course in a classical programming language such as Pascal, C, PL/1, Algol, etc . It is designed to introduce them to modern object-oriented concepts as well to the C programming language. (No previous knowledge of "C" will be formally assumed).

However students who have taken their introductory programming course in C++ may take this both for additional review, and to gain experience with more complex programming projects. Students who have no background in computer programming should instead take the 171/172 sequence.

This semester there are several students who have been unable to take 172 before, and have been given permission to take this course instead. These students will be asked to do a subset of the assignments.

Instructor: Dr. R. Michael Canjar

Office:	Engineering 323
Office Hours:	Mon/Tues: half hour before and after Ford classes. Wed: 3:00-5:00 PM (on campus) Thurs: 11:30 AM 12:30 PM (on campus) & by Appointment
Phone:	(313)-993-1209
E-Mail: *	canjarm@udmercy.edu
Home Page:	http://es.udmercy.edu/~canjarm/canjarm.htm
Course Page:	http://es.udmercy.edu/~canjarm/oop-ford/oop-www.htm

* E-mail is the most efficient way to reach me. When you send me an E-mail, please indicate "Question from OOP" in the subject line. I give priority to responding to E-mails that have questions from students in my classes.

Required Text:

Ira Pohl; *Object Oriented Programming Using C++*, 2nd Edition.
There will also be supplementary lecture notes and handouts.

Optional:

Any third-party book on using Visual C++

Software:

Microsoft [Visual C++ Version C++ 5.0/6.0](#) will be the official compiler for the course. Instruction will be oriented toward those compilers. It is available in the University computer labs.

General Objectives:

1. To introduce the basic concepts of Object Oriented Programming.
2. To introduce the C++ language.

Specific Objectives:

Upon completion of the course, students will

1. Understand the **C-language kernel** of C++.
 2. Understand and be able to employ **data encapsulation** and **abstraction** and be able to create their own **Data Types**, possessing the same robustness as Native Types.
 3. Understand the use of classes and objects in Software Development and be able to use them in a **team project**.
 4. Understand the application of OOP to **Containers and Iterators** and be able to write and employ container classes in their programs.
 5. Understand Polymorphism, both parametric (templates) and inherited and be able to use **class libraries**.
-

Major Topics:

1. Introduction to C/C++ Programming. The C Kernel of C++. Stream based Input-Output.
 2. Pointers and Arrays in C and the semantics of C-pointer operations.
 3. Introduction to Classes. Operator Overloading. Class Example: Fraction
 4. Introduction to Data Abstraction. Class Examples Vect. and Stack
 5. Constructors, Destructors, and Copy-Semantics.
 6. Using OOP in large software projects. Team Project: Fraction Calculator.
 7. Containers and Iterators: Enhancing the class Vect
 8. Inheritance and the Use of Virtual Functions.
 9. Templates.
-

Instruction Method and Techniques:

1. There will be lectures of 150 minutes each week.
2. Software and Example programs will be demonstrated in class. Examples will be made available on class handouts and may be downloaded from the Internet.
3. Students will have 4-5 individual lab assignments to be done outside of class.
4. There will be one Team Project, done outside of class over several weeks..

Assignments for the Course:

1. There will be readings from the Text supplemented by class handouts.
2. There will be 3 individual programming assignments, some of which may be in several parts. Students taking the class for graduate credit will have a 4th assignment. (See the schedule below.)
3. There will be a major Team Project.
4. There will be an in-class Midterm and Final Exam. The current plan is for this exam to be open-book open notes, closed computer, and closed friends.

Course Evaluation

Individual Assignments	30%
Team Project	20%
MidTerm Exam	20%
Final Exam	30%

Differential Assignments

Students who are taking this class in *lieu* of CSC 172 will be asked to do a reduced set of assignments.

Make Up Policy :

Make Up exams will only be given to students who miss an exam for legitimate reason (as defined above under "Attendance") and who notify the instructor in advance.

No Mid-Term Grades

Because most of the grade evaluation in the class is based upon more advanced projects will be due later in the semester, there will be no basis for giving out interim Mid-Term grades.

Attendance/Participation :

Students are expected to attend class on a regular basis and participate in the discussions. They are responsible for all the material presented therein. Formal attendance records will not be maintained; however attendance is highly correlated with performance on the projects and the exams.

The instructor will attempt to make reasonable accommodations for students who miss a class due to illness, death in the family, or other legitimate reasons. However students who are forced to miss several classes will have difficulty completing the course in a satisfactory manner.

Academic Integrity

Students are expected to conform to a high standard of honesty and integrity in this course. Copying the work of someone else and other forms of cheating are strictly prohibited. Permitting or tolerating such behavior is also prohibited. The minimum penalty for any offense is a 0 on that assignment. The culprits may be subject to additional sanctions, up to and including expulsion from school for serious offenses, as prescribed by the University Catalog and the Engineering Science Student Handbook.

For the programming assignments, you may freely discuss the concepts and ideas of the projects. You may also get any help regarding use of the hardware/software. However any code written should be your own; you should not copy all or part of the code of someone else. Generally speaking, access to another's computer files in either computer or printed form can constitute a *prima facie* case of cheating. Do not "loan" your files to anyone in any form for any reason.

Tentative Course Outline

Project 1:		
Introducing C++ :GCD Program	Tues. 9-17-02	
Project 2:	The Kernel language Arrays, Structures, Pointers	
	Part A: Arrays	Tues. 10-1-02
	Part B: Pointer	Tues. 10-15-02
Project 3	Team Project, Fraction Calculator	Tues. 11-5-02

MidTerm Exam		Tues. 11-12-02
Project 4:	Containers, Iterators	Tues. 11-26-02
Project 5:	The Standard Template Library; Inheritance	Tues. 12-3-01
Final Exam		Tues. 12-10-01

Other Important Dates

Spirit Day	Thursday, September 12
Last Day to Withdraw for 100% Refund	Tuesday, September 10
Last Day to Withdraw for 75% Refund	Tuesday, September 17
Last Day to Withdraw for 50% Refund	Tuesday, September 24
Last Day to Withdraw for 25% Refund	Tuesday, October 1
Last Day to Delete a Course without a "W"	Friday, September 27
Undergraduate MidTerm Grades Due (N/A)	Tuesday, October 22
Last Day to Withdraw from a Course	Monday, November 18
Thanksgiving Break(University closed)	November 23 - 29

Ground Rules For Programming Assignments

1. **Source Language:** All projects are to be done in C++, **using aspects of the language that are discussed in class.** There are C-Language (CL) constructs which, while still legal in C++, have been superseded. For the purposes of this course, the C++ constructs are to be used. The following list includes some common examples of this. Students with prior CL experience should take care to use the C++ constructs; the old CL constructs are not to be used.

- Use I-O **Streams** in *iostream.h* and not *stdio.h*
- Use **enums** , **constants** and **inline functions** instead of *#define*
- Use the **operators new** and **delete** and not *malloc()* and *free()*.

As mentioned on the syllabus, Microsoft [Visual C++ Version C++ 5.0/6.0](#) is the recommended compilers for the course. These compilers require 32 bit Windows. Students working with other compilers must do so at their own risk.

2. **Hand-in:** For all projects, students should hand in a **copy of their source code** and output from a sample run of the program, **using any test data** that have been provided. In addition, the Output should **always be date-stamped** by the computer, indicating the date on which the output was produced. (Functions to assist in producing this output will be presented in class.)
3. **Grading:** Programs will be graded for scope, correctness, style, documentation, attractiveness of output, and timeliness. A program that "works" will not receive full credit unless it is well-written, properly documented, and uses the appropriate style. Please refer to the comments below regarding style.

Late projects will not receive full credit. Unless otherwise specified, projects turned in late **may lose as much as 20%** and projects may **not be accepted at all after that week**. Students who are unable to complete a project within the allotted time should discuss the situation with the instructor as soon as practical. The deadline for projects will usually be **the beginning of class** on the due date. **Under no circumstances should students skip all or part of class to work on the project.** Students who attend class on the day that an assignment is due will usually be granted a brief extension, if needed. **Projects turned in on time will usually be returned at the next class meeting.** No such guarantee is available for late projects.

4. **Style:** Style is an important component of programming. Programs should employ meaningful identifiers and informative comments. Capitalization may be used provided it is in a consistent, coherent manner. (You may also choose to follow the text and use exclusively lower case identifiers with underscore characters to enhance readability.)

Unlike programs written in more self-documenting language like Pascal, CL programs tend to be cryptic and difficult to understand. Students should make judicious use of informative comments in their projects. Note that comments that merely repeat the meaning of an instruction are not informative. Comments should explain the purpose and functioning of

code, not merely repeat it. (Some of the class handouts and text examples will have comments targeted at students learning the language for the first time. These comments would, of course, be inappropriate for other programs and for the homework assignments.)

Each program should certainly include a comment near the beginning giving the name of the project, the author, and his/her section number. Some explanation of each function is usually appropriate.

5. **Academic Integrity** Students are expected to conform to a high standard of honesty and integrity in this course. Copying the work of someone else and other forms of cheating are strictly prohibited. Permitting or tolerating such behavior is also prohibited. The minimum penalty for any offense is a 0 on that assignment. The culprits may be subject to additional sanctions, up to and including expulsion from school for serious offenses, as prescribed by the University Catalog and the Engineering Science Student Handbook.

For the programming assignments, you may freely discuss the concepts and ideas of the projects. You may also get any help regarding use of the hardware/software. However any code written should be your own; you should not copy all or part of the code of someone else. Generally speaking, access to another's computer files in either computer or printed form can constitute a prima facie case of cheating. Do not "loan" your files to anyone in any form for any reason.
