

INSS 505, Introduction to Object-Oriented Programming
Maryland in Europe Graduate Programs
Bowie State University, Term 4, 2003
Panzer Computer Lab, Stuttgart, Germany
Instructor: Andrew Seely

Meeting Times: 0900-1700, 29 March, 5, 12, 26 April, 3, 10, 17 May 2003

Official Course Description for INSS 505

This course teaches the principles and techniques of object-oriented programming and design. It is designed to teach students an object-oriented programming language such as C++ or Java. This course satisfies the programming language prerequisite for the M.S. program. It is open only to students who have not previously taken an object-oriented programming course.

Student deliverables for this class:

1. Programming Projects: There will be five programming projects for this course. Projects must be turned in on a floppy disk and as a printout. Source code should be in ASCII text files in either DOS or UFS format. Projects should produce error-free bytecode with a single Java compiler command. Specific topics for the projects will be announced on the first day of class. The projects will tentatively be as follows:

1. Project 1: Designing Interfaces
2. Project 2: Protecting Classes with Encapsulation
3. Project 3: Working with Inheritance
4. Project 4: Practicing with Polymorphism
5. Project 5: Code Reuse

2. Writing Assignment: Due the last day of class; write an 800-1200 word essay describing a non-computer system in object oriented terms. The goals of this assignment are to demonstrate understanding of the object oriented paradigm as a model for complex systems and to recognize the practical application of this model in diverse environments. The topic of your paper may be a social system or a physical system but should be a system that lends itself to an object oriented discussion. References are encouraged but not required, but your discussion should be firmly grounded in the principles covered in class and in the text. Papers should be prepared according to the manuscript style described by the American Psychological Association's publication manual.

3. Exams: This class will have a single, comprehensive exam on the last day. This exam may consist of code writing, short-essay, system modeling, and other challenges.

Introduction to the Instructor: Andrew Seely

Background

- Doctoral Student-Computer Science, Nova Southeastern University
- MS-Computer Science, Nova Southeastern University
- BS-Computer and Information Science, University of Maryland University College-Europe
- Solaris Certified System Administrator, Linux Certified Professional

Contact

- UMUC Homepage: <http://faculty.ed.umuc.edu/~aseely/>
- Personal Homepage: <http://www.sonador.com/arseeley/>
- Email: aseely@faculty.ed.umuc.edu
- Mobile Phone: +49 (0) 175 244 8070
- Post: CMR 480 Box 772, APO AE 09128

The official web site for this course

<http://faculty.ed.umuc.edu/~aseely/inss505/>

Required Text

Wu, C. T. (2002). An Introduction to object-oriented programming with Java 2rd Edition, New York: McGraw-Hill.

System Requirements

All programming will be done using the Java Runtime Environment version 1.4, available for free download from <http://java.sun.com/> or on the CDROM available in class. Students should have sufficient computing resources to allow projects and homework to be fully completed outside of class. While we will hold class in the computer lab, we will have limited class time for working on assignments.

Grading Policies

Each project is worth 10%, the essay is worth 20%, and the final exam is worth 30% of the final grade. Final letter grades will be assigned as follows:
100-90% == A, 89.9-80% == B, 79.9-70% == C, 69.9 on down == F

Attendance and Late Policies

Students missing more than 25% of scheduled class time must drop or take an F(n). Late assignments will be accepted. Late assignments will not be accepted without prior coordination.

Programming Styles

This course will require a moderate amount of programming. Attention to code formatting and comments is essential. Format will be graded along with function. Minimum guidance:

1. Every source code file should begin with a comment block identifying the project, title, student, course, school, term, and due date for the file. Multiple-file assignments should also note which file is the primary code and which files are supporting code.
2. Every file should have a comment block to briefly describe the general purpose or function of the code in the file.
3. All variables should have a comment defining their purpose when they are first initialized unless their purpose is clear by the variable name. Loop increment variables do not need to be defined.
4. All code lines within the same block should be indented the same amount when possible. Code blocks defined by brackets should be commented to show which block is being started or ended.
5. When possible, file names should reflect the assignment and student name.

Plagiarism

The UMUC catalog suggests that any help you receive on a written work may be considered cheating unless otherwise stated. For this class, you are highly encouraged to make use of any printed or electronic programming reference available to you, but it is not acceptable to copy and paste code for your projects or homework. Use of standard code libraries distributed with the Java Development Kit is acceptable but use of third-party libraries may not be. If in doubt, check with the instructor. Code that is found to have been copied from another source will receive a grade of zero for the assignment.

Office Hours

The instructor will be available after class, by appointment, and any time via email to discuss class-related issues.

Tentative Schedule

Day 1:

The object oriented paradigm in detail; key terms and concepts; classes, objects, and interfaces; imperative programming syntax in Java; Chapters 1, 2, & 3.

Day 2:

Project 1 due.

Object oriented system design; continued discussion of classes and interfaces; concept and practice of encapsulation; primitive and complex data types in Java; Chapters 3 & 4.

Day 3:

Project 2 due.

Inheritance concept and practice; designing classes for reuse; introducing polymorphism; working with the Java Foundation Classes; Chapter 3, 4, 12, & 14.

Day 4:

Project 3 due.

Inheritance continued; application of polymorphism; treating input and output in an object oriented way; Chapters 11 & 14.

Day 5:

Project 4 due.

The graphical user interface and the object-oriented paradigm; Chapter 5 & 13.

Day 6:

Project 5 due.

Graphical interfaces continued; Practical applications of object oriented system design.

Day 7:

Essay due.

Total review of OOP.

Final Exam.