

CMSC 132: Object-Oriented Programming II



Course Introduction

Department of Computer Science
University of Maryland, College Park

Things You Will Learn

- Object-oriented software development
 - Modern software development techniques
 - Object-oriented design
- Algorithms & data structures
 - Lists, trees, graphs
- Programming skills
 - Java API, IDE, testing, debugging

Course Is Not Just About Java

- May seem to focus on Java
 - All programming in Java
 - Many interesting Java language features
- Lessons intended to be general
 - Principles should apply to all languages
 - Ways of thinking about design
 - General ideas about software
 - Can translate skills to other languages

Course Catalog Description

- Introduction to use of computers to solve problems using software engineering principles
- Design, build, test, and debug medium-size software systems. Learn to use relevant tools
- Use object-oriented methods to create effective and efficient problem solutions
- Use and implement application programming interfaces (APIs)
- Programming done in Java

Why Object-Oriented Programming?

- Coding is small part of software development
- Estimated % of time
 - 35% Specification, design
 - 20% Coding, debugging
 - 30% Testing, reviewing, fixing
 - 15% Documentation, support
- Object-oriented approach makes other parts of software development easier

Assume You Already Know

- Coding
 - Variables, operators, loops, arrays
- Basic object-oriented programming
 - Classes, methods, inheritance
- Java
 - Class libraries, exceptions
- Tools
 - Eclipse IDE, debugger

Where does 132 fit in?

- **CMSC 131**
 - Basic programming skills
- **CMSC 132**
 - Software design & basic algorithms
- **CMSC 212**
 - Low-level programming
- **CMSC 250**
 - Discrete math & logic
- **CMSC 351**
 - Analysis of algorithms

Organization

- **Personnel**
 - **Instructors**
 - Nelson Padua-Perez
 - Chau-Wen Tseng
 - **Teaching assistants**
 - Fatih, Liping, Saket, Adam, Eric, Nick, Jonathan
- **Classes**
 - Lectures
 - Labs
 - Office hours

Textbook

- **Recommended**
 - “Objects, Abstractions, Data Structures and Design Using Java (version 5.0)”
 - By Elliot Koffman and Paul Wolfgang



Textbook (cont.)

- **Recommended**
 - “Java Precisely (2nd Edition)”
 - By Peter Sestoft



Projects

- **8 projects**
 - Evaluate design, coding, testing skills
 - Tries to involve interesting application areas
 - Networking, user interfaces, data compression
- **Late policy**
 - Projects due at 6 pm
 - 20% penalty, up to 9am the next morning
 - Plan to complete all projects on time
- **Good faith attempt**
 - Must attempt all projects to pass

Projects (cont.)

- **Environment**
 - **Eclipse IDE**
- **Automated submission & testing**
 - **Submit server**
 - <https://submit.cs.umd.edu>
 - **Maintains record of submissions**
 - CVS repository
 - May use for research
 - **Release testing**
 - Can evaluate project using real test cases

Grading

- **Based on**
 - Projects, homework exercises, quizzes, midterms, final
- **Point distribution (roughly)**
 - 40% Projects
 - 6% Homework Exercises
 - 14% Quizzes
 - 10% Midterm #1
 - 10% Midterm #2
 - 20% Final Exam
- **Available on-line**
 - <https://grades.cs.umd.edu>

Academic Honesty

- All individual assignments & exams must be done individually (except "open" assignments)
- Do not copy (or allow others to copy) your work in any way
- **Submissions will be compared to submissions from current and previous semesters**
- Cases of academic dishonesty will be referred to the University's Office of Judicial Programs
- Visit Student Honor Council website for more detailed explanation of academic dishonesty

Course Advice

- Start projects **early**
- Ask questions
- Read book
- Attend lectures
- Attend labs
- Attend office hours

Course Bulletin Board

- **Bulletin Board (Forum)**
 - <https://forum.cs.umd.edu/forumdisplay.php?f=67>
- **Policy on project postings**
 - Can ask about specification, setup, tools, etc.
 - Do **not** ask about design, implementation, etc.
 - Violators may face penalty for academic dishonesty

Excused Absences

- Students must apply in writing and furnish documentary support for excused absences
- Support should explicitly indicate the dates or times the student was incapacitated
- Excused absence does not typically translate into project extensions
- Students requesting reasonable academic accommodations due to a disability must provide a letter from the Office of Disability Support Services

Topics Preview

- **Algorithms & data structures**
 - Asymptotic efficiency
 - Lists, stacks, queues
 - Trees, tries, heaps
 - Sets, maps, graphs
 - Recursion

Topics Preview

- **Object-oriented software development**
 - Software life cycle
 - Requirements & specifications
 - Designing objects & classes
 - Testing & code coverage
 - Unified Modeling Language (UML)
 - Programming paradigms
 - Design patterns

Topics Preview

- **Programming skills**
 - Java collection framework
 - Exceptions
 - Threads, synchronization
 - Java APIs
 - Networking
 - Graphics User Interfaces (GUI)