COMPSCI 121: An Introduction to Problem Solving with Computers

Contents

Course Description

We welcome you and your fellow students on this journey of learning. COMPSCI 121 provides an introduction to problem solving and computer programming using the Java programming language. 121 is intended to teach you how to program in Java, one of the most popular modern computer languages. Java brings a discipline to programming, called the object-oriented paradigm.

No previous programming experience is required; however, this course is intended for Computer Science majors or those who plan on applying to the major. The course is the first required class for the UMass computer science major; it is also required of Informatics majors, electrical engineering majors, and mathematics majors at UMass-Amherst. In addition to basic programming constructs such as looping, conditions, arrays, file handling, and methods, much attention is given to the Java object model as well as to Java's event model and its relation to graphical user interfaces.

Non-majors or students who do not have previous programming experience should strongly consider taking an introductory programming course designed specifically for non-CS majors such as COMPSCI 119 or COMPSCI/STATS 190F.

Our hopes and vision for the course

We envision this course as a supportive and inclusive learning community. We hope that this course will be a starting point for you to develop and deepen your awareness of problem solving with computers. Our aim is to provide you with knowledge and skills and inspire and foster your commitment to work towards the goals of your major. Our specific objectives are to:

1. Enable you to develop problem solving and programming skills to design solutions to non trivial problems and implement those solutions in Java.
2. Provide you with knowledge of essential facts, concepts, principles and theories relating to object oriented programming and design.
3. Prepare you to develop programming skills that can serve as a foundation for further study in computer science.
4. Equip you with skills to work productively as part of a team and to develop your ability for organization, communication, and collaboration.

What you'll learn

This course is designed to offer you opportunities to expand your thinking and understanding about programming. At the end of the course you should be able to:

1. Demonstrate knowledge and understanding of fundamental programming constructs, variables, expressions, assignments, I/O, control constructs and recursion.
2. Deploy appropriate theory, practices and tools for problem definition, specification, design, implementation, and testing of programs that use basic computation, simple I/O, standard conditional and iterative structures, the definition of methods, and parameter passing.
3. Use object-oriented design (inheritance, interfaces, polymorphism, abstract classes) as a mechanism for problem solving as well as facilitating modularity and software reuse (refactoring).
4. Design and model recommended programming practices (good java style and documentation, UML diagramming, and testing).
5. Work productively as part of a team and demonstrate your ability for organization, communication, and collaboration in teams.

How you'll learn

The course follows a flipped classroom model of teaching and learning. We will be using lectures, labs, and the "Programming in Java (Early Objects)" online textbook from zyBooks.

1. **Lectures:** Classes are highly interactive and rely on your thoughtful contributions. Participation activities from the textbook are due before lecture and challenge activities are due after the lecture. Lectures parallel the text material, but may expand upon and enrich concepts from the text. During the lecture, you can download starter code and develop along with the instructor and use iclickers for group work.
2. **Labs:** Labs allow for more individualized support, the opportunity to build deeper personal connections with peers, and engaged, active learning. You work in a group at developing code and answering questions that reinforce the topics covered in the text and in lecture. Although the lab document is submitted individually, the results may be the same for each group member.

3. **Textbook**: We use zyBooks' Programming in Java (early objects) state-of-the-art learning material, proven effective, and designed to maximize learning while respecting student time. The online textbook has embedded exercises and assignments. Participation activities are due before lecture and challenge activities are due after the lecture.
4. **Programming Projects:** We assign a number of programming projects during the semester. These projects provide you with an opportunity to apply the skills and concepts you learn in the course to more involved coding scenarios. You will write code that implements one or more major functions to a Java application.

In this course, we use Moodle and Gradescope systems to enrich your learning experiences.

1. **Moodle** is a Learning Management System (LMS) used for posting lecture materials and for online exams. The Moodle webpage, when expanded, shows the weekly topics and material covered in lecture and labs. You are enrolled in the Moodle course through Spire.
2. **Gradescope** is a grading management system that allows us to give you timely feedback for your lab assignments and programming projects. You will automatically be enrolled in Gradescope at the beginning of the semester.

How you'll know you are learning

You have a number of opportunities in this course to demonstrate your learning and earn credits towards your final course grade. You earn credits from the zyBook and lab assignments, projects, exams, and attendance at labs and lectures. The exams test the ability to recognize, trace, implement, and translate code. The assignments and projects deal with higher order programming skills (analyse, adapt, debug, test, apply, design, model, and refactor code).

What you Should Know about Grading Categories and Weights

The breakdown for your final course grade is given below and helpfully mapped for you in the table with the specific learning outcomes you achieve.

- zyBook questions including end of chapter exercises: 10%
- Lab Participation(attendance and document) 10% (the first 2 labs are not counted; the lowest 2 grades thereafter are dropped)
- iClicker questions: 5% (based on lecture attendance; the first 2 lecture sessions are not counted; the lowest 2 grades thereafter are dropped)
- Programming projects: 35% (all projects are counted- no project grades are dropped)

- Exams (four): 40%. ***In order to achieve a Pass grade or a letter grade of D or better, all exams must be attempted and the average of all exam scores must be at least 50% of the total exam points.***

Learning Outcomes	zyBook questions and exercises 10%	Exams 40%	Projects 35%	Labs 10%	iClicker 5%
Recognize, trace, implement, translate, debug code	✓	✓		✓	✓
Analyze problem, apply, adapt, relate, debug	✓	✓	✓	✓	
Model, design, test and refactor			✓		
Teamwork values			✓	✓	

Your final letter grade is determined by calculating a *course final grade* based on a weighted sum of your scores for individual grading categories. This number is mapped to a letter grade. You may expect the following ***approximate*** grading range, however; *note that the final letter grade determination is made at the end of the semester*:
90 to 100 -> A- and A;
80 to 89 -> B-, B and B+;
70 to 79 -> C-, C, and C+;
60 to 69 -> D and D+.

You can take advantage of the wide range of assessments to be successful in this course. We regret that there are no opportunities for extra credit. To ensure fair grading we may make adjustments to the grading ranges listed in this document and we may also assign a grade to you based on extenuating circumstances and/or our judgment. See the *Attendance and Grading Policy* document posted in Moodle for more details of how assignments and projects are graded.

You have an opportunity to review the grades you receive on programming projects and exams within 3 working days of the release of the grade. It is important that you ask any questions as soon as possible to clarify any points about your grade or the content you submitted. To ensure timely action, we will not review any grades after 3 working days beyond their release date.

We retain all graded materials for this course until the end of next semester. If you wish to review them, please make an appointment to see us.

What course materials you need

Do I need a computer to do the class?
You will need a reliable laptop computer that can compile and run Java for this class. Your laptop must be configured to access the Eduroam wireless network. Please see or visit the campus OIT office for support. See: https://www.umass.edu/it/wireless.

Do I need a clicker for the class?
iClickers are required for lectures. You must register your iClicker in the Moodle 121 course page to get points for answering iClicker questions. You must purchase a specific type of iClicker model 2. See https://www.umass.edu/it/audience-response-system.

The Academic Oversight Committee (room 319 of Bartlett Hall) of the Student Government Association has created an iClicker Lending Library. This library is free, and it allows students to sign out an iClicker for a semester. The iClicker is registered to the student who is taking it out, and this student is able to use the iClicker in any class that asks for iClickers to be used. These iClickers must be registered on the Moodle 121 course page.

Are there special software tools that I need for this class?
You'll need to install Java and the jGRASP development environment for this course. We will provide you with specific instructions about which software to install at the beginning of the course. All software used in the course is freely available. While we strongly recommend jGRASP, you may use an alternate development environment other than jGRASP if you prefer; however, we will provide support for jGRASP but no other IDE (for e.g. Eclipse, Dr Java etc.). Any alternate IDE must include a debugger, the ability to run JUnit tests, and project management.

How learning is accessible to you

Feel Included

In this course, each voice in the classroom is valued. We honor UMass's commitment to embrace diverse people, ideas, and perspectives to create a vibrant learning and working environment. You are welcome regardless of age, background, citizenship,

disability, education, ethnicity, family status, gender identity, geographical origin, language, military experience, political views, race, religion, sexual orientation, socioeconomic status, and work experience.

This course is geared towards you working in groups. As such, we expect that you will observe social decorum at all times when interacting with peers. Please consult the UMass Guidelines for Classroom Civility and Respect: http://www.umass.edu/dean_students/campus-policies/classroom

Get Disability Accommodation

If you have a disability and require accommodations, you will need to register with Disability Services (161 Whitmore Administration building; phone 413-545- 0892). Information on services and materials for registering are also available on their website: www.umass.edu/disability. It is our goal to provide every student with a high quality learning experience. We invite you to contact us if you have any questions or concerns about disabilities or any issue that may impact the quality of your learning.

Students who are registered with Disability Services (DS) for time or other exam-related accommodations must schedule exams with DS to take the exam with them. Log into this site to schedule your exam.

Plan for Success

Your success in this class is important to us. We all learn differently and bring different strengths and needs to the class. If there are aspects of the course that prevent you from learning or make you feel excluded, please let us know as soon as possible. Together we'll develop strategies to meet both your needs and the requirements of the course. There are also a range of resources on campus, including:

- Writing Center - http://www.umass.edu/writingcenter
- Center for Counseling and Psychological Health (CCPH) - http://www.umass.edu/counseling
- English as a Second Language (ESL) Program - http://www.umass.edu/esl

Communicate with Us

We have 2 channels of communication for you to use with the course staff.

1. Use the Moodle "***Private Student Forum***" for your private messages (about grades, absences, or extensions) or to make an appointment with the professors or Teaching Assistants (TAS), and we will get back to you as soon as possible.
2. For questions about the course or the content of your studies use Piazza, which is designed to connect students, TAs, and professors. You can also answer questions from other students in Piazza.

See the *Communications Polic*y document posted in Moodle for more details.

How you can be successful in this course

Attend all Classes and Labs

This is a highly interactive course. This means it is important that you attend every class and lab, are on time, and ready to engage actively with your peers.

Please ensure that your laptop computer is reliable and that you are always able to access the internet through a reliable connection. If you experience technical issues you are responsible for obtaining a working machine. Laptops may be rented in the main library in such cases. We do not extend the due dates for labs, homework, and projects due to technical difficulties such as network connectivity or computer problems.

There can be situations when you do not attend a lecture or lab. You are allowed to miss two labs and two lectures during the semester with no penalty. If you miss more than two labs or lectures, please contact the professors.

Note that there are other conditions about absences that apply to this course. For example, we do not give extensions or makeups for family travel for vacations and other recreational events. Please see the *Attendance and Grading Policy* document posted in Moodle for more details.

Visit Office Hours

Office hours are an important part in supporting you throughout this course. During office hours you can visit the professors and TAs for questions about the course material. No appointment is necessary for office hours. See the Moodle page for office hours for professors and TAs. Drop by to meet us for specific questions, needs, and concerns.

Approach Tutors for Help

The Learning Resource Center (LRC) of the University of Massachusetts Amherst offers you a peer-supported environment to meet academic challenges. Tutors, Supplemental Instruction Leaders, and ExSEL Leaders are model students trained to assist you in achieving academic success. The support staff are available at the LRC, 10th floor in the Main Library (Du Bois). See http://www.umass.edu/lrc/

Follow the Academic Honesty Policy

We want our learning environment to be honest and fair. UMass Amherst has an Academic Honesty Policy that includes cheating and plagiarism as forms of dishonesty. Read the *Academic Honesty policy* document posted in Moodle for tips on how to keep from violating the policy.

Topic Outline for the Course

CODE DEVELOPMENT
- Coding style
- Commenting and documentation
- Modular code design
- Diagramming techniques
- Skeleton code (stubbing)
- Testing your code: Unit tests
- Using a debugger

PRIMITIVE DATA TYPES
- Numeric: int, double
- Logical, character: boolean, char

ABSTRACT DATA TYPES (Classes)
- Java library classes: String, Scanner, Random, etc.
- User-defined classes

OPERATORS
- Assignment and Arithmetic Operators
- Relational Operators, Increment Operator, Logical Operations, Concatenation

VARIABLES
- Initialization of Instance Variables
- Type Conversion, Numeric Cast
- The final Keyword
- Static final Variables

EXPRESSIONS
- Assignment
- Flow of Control: if, else
- Null References
- Packages
- Import Statements

RUNNING A PROGRAM
- The main() Method
- Using System Resources

Compiling Java source files
Using a debugger

OBJECT-ORIENTED PROGRAMMING
Defining a Class
Public Class Files
Objects and Encapsulation
Constructors
Instance Members, Class Members and Finalization
Static Members
Setter and Getter Methods
Member Classes
Local Classes
Anonymous Classes
Nested Top-Level Classes
Casting

METHODS
Method Signature
Access level, return type, name, arguments
Method Overloading
Static Methods

INTERFACES
Interfaces and polymorphic behavior

IMPORTING JAVA LIBRARIES

GRAPHICAL USER INTERFACE (GUI)
The event model
Java libraries awt and swing

STRINGS
The toString() Method
StringBuffer

EXCEPTIONS
Handling Errors Using Exceptions
Common Exceptions

Checked and Unchecked Exceptions
Chained Exceptions in Java

I/O

Standard Input and Output Stream Classes
File I/O using Scanner

INHERITANCE

Extending a superclass (generalization)
Abstract classes

POLYMORPHISM

Polymorphism Based on Overloaded Methods
Polymorphism, Type Conversion, Casting, Etc.
Runtime Polymorphism through Inheritance
Polymorphism and the Object Class

ARRAYS

Arrays of Primitive Types
Array indexing
Length of an array
Arrays of Objects
2D arrays