COMPSCI 121: An Introduction to Problem Solving with Computers

Contents

What this course is about 2
Our hopes and vision for the course 2
What you’ll learn 2
How you’ll learn 3
How you’ll know you are learning 4
  What you Should Know about Grading Categories and Weights 4
What course materials you need 5
How learning is accessible to you 6
  Feel Included 6
  Get Disability Accommodation 6
  Communicate with Us 7
How you can be successful in this course 7
  Attend all Classes and Labs 7
  Visit Office Hours 7
  Approach Tutors for Help 7
  Get an Extension if Needed 8
  Follow the Academic Honesty policy 8
What you will learn along the way 8
What this course is about

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. The course teaches how problems can be solved computationally using the object-oriented approach that underlies Java. No previous programming experience is required.

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. See the Lab Protocol document in Moodle for more information.

3. **Textbook:** We will 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.

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 receive a course code to create a student account in Gradescope.

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 are dropped)
- iClicker questions: 5% (based on lecture attendance; the first 2 lecture sessions are not counted; the lowest 2 grades are dropped)
- Programming projects (5 projects): 35%
- Exams (four): 40%. In order to pass the course with 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.

<table>
<thead>
<tr>
<th>Learning Outcomes</th>
<th>zyBook questions and exercises 10%</th>
<th>Exams 40%</th>
<th>Projects 35%</th>
<th>Labs 10%</th>
<th>iClicker 5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recognize, trace, implement, translate, debug code</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analyze problem, apply, adapt, relate, debug</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model, design, test and refactor</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Teamwork values</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
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:

90 to 100 -> A
80 to 89 -> B
mid 60 to 79 -> C
50 and above -> 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 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 days beyond their release date.

We will retain all graded materials for this course until the end of next semester (Spring 2019). If you wish to review them, please make an appointment to see us.

What course materials you need

1. You will need a reliable laptop computer for this class. Chromebooks, Microsoft Surface, and tablets will not work for this course. Your laptop must be configured to access the Eduroam wireless network (https://www.umass.edu/it/wireless).
2. Please see or visit the campus OIT office for support. Do note that lack of working equipment or reliable internet access is not a valid excuse for failure to submit assigned work. See the Software Requirements document posted in Moodle for more details.
3. iClickers are required for our interactive lectures. You must register your iClicker in the Moodle course page to get points for answering iClicker questions. You should purchase the specific type of iClicker model 2 - see https://www.umass.edu/it/audience-response-system.
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, please let us know as soon as possible. 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.

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) 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 Policy 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. Should you need to miss a class, please let us know in advance via the Moodle private forum. If you miss more than two class/lab sessions, we need to meet to figure out how you can still pass the course as attendance counts towards your grade.

The lowest 2 grades for labs and the iClicker questions are dropped. Beyond these, medical conditions, religious or funerary events, university-related event (athletic event, field trip, or performance), extenuating non-academic reasons (military obligation, family illness, jury duty, automobile collision) will be accommodated with written documentation. 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 instructors and TAs for questions about the course material. No appointment is necessary for office hours. See the Moodle page for office hours for Instructors 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/

Get an Extension if Needed

We understand that sometimes you may need an extension for a submission. Contact us at least 24 hours before the assignment (project or lab) is due. Medical conditions, religious or funerary events, university-related event (athletic event, field trip, or performance), extenuating non-academic reasons (military obligation, family illness, jury duty, automobile collision) that need extension will be accommodated with written documentation. Please note that problems with computer or internet access, holiday, or family travel are not valid excuses.

To ensure that you submit your work on time we recommend that you begin early and not wait until the last minute to submit. You will be able to submit multiple times so submit early and often to ensure you have something in before the deadline.

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.

What you will learn along the way

CODE DEVELOPMENT
- Coding style
- Commenting and documentation
- Modular code design
- Diagramming techniques
- Skeleton code (stubbing)
- Test your code

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
ARROWS

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