Syllabus

CS121 Computer Science 2: Data Structures and Objects

Catalog Course Description

Continuation of Computer Science 1. Strong emphasis on object-oriented paradigm and related problem-solving skills. Topics include recursion, dynamic data structures and related algorithms, efficient sorting and searching, file processing, and reusing standard libraries and packages in chosen object-oriented environment. Hands-on lab exercises and programming assignments illustrate these topics and build problem-solving abilities.

Credit-hours: 4

Prerequisite

CS120 with a minimum grade of C-

Instructor and Session Details

Instructor and section information (https://bsu.instructure.com/courses/191261/pages/instructor-and-section-information)

Course Objectives

Students will:

- **Create, refine, and execute** Java applications involving linear data structures and user interaction (GUI or CHUI), based on provided requirements, using an integrated development environment
- **Analyze and write programs** that demonstrate understanding of the following Java language features:
  - loops
  - conditionals
  - fields (distinguishing static vs. instance)
  - methods (distinguishing static vs. instance)
  - classes and objects
  - access control modifiers

    - implementation inheritance via `extends`
    - interface inheritance via `implements`
    - the use of generics in the Java API
    - file I/O
    - reading XML or JSON as an example of an *n*-ary tree
- **Compare** alternative implementations of abstract data types (*e.g.* linked list vs. array-based list).
- **Describe** common applications where the following data structures would be used, and **choose** the appropriate data structure for a problem from: array, linked list, stack, queue, hash table, and binary search tree.
- **Explain** recursion and **implement, test, and debug** simple recursive functions and procedures.
- **Compare** iterative and recursive solutions to elementary problems such as factorial and **explain** which solution is appropriate.
- **Demonstrate** preorder, inorder, and postorder traversal of a binary search tree.
- Informally **analyze** the worst-case performance of search and sort algorithms, over linear data structures and binary search trees, using Big-Oh notation.

Additional Objectives of this section

- **Compare** command-line operating system to the more familar GUI system
- **Utilize** pointers and other low-level memory management tools to understand how computer memory works
- **Compare** heap and stack memory allocation to appreciate how these different allocation methods work
- **Explore** multiple programming languages and paradigms to get a wider appreciation of syntax variations

Course Rationale

This course helps students develop their programming skills while introducing concepts of data structures. The course uses the object-oriented paradigm in Java. This will be a significant shift for the students since CS120 tends to use a procedural approach. Hence, significant attention is paid during the semester to the fundamentals of Java programming, including programming syntax, the mechanisms of object-orientation, and how to interact with an integrated development environment.

This particular section of the course uses a historical overview of programming languages to explain how we got to the current state of object-oriented programming and data structures.

Early examples will be in the C and C++ languages, explaining some lower-level concepts that are often hidden from the programmer in higher-level languages. We will then switch to Java. This approach will assist the learner with a language-neutral approach to problem-solving, and will facilitate learning new languages and paradigms in the future. We will also work directly with the Unix/Linux command line, as this approach leads to great flexibility and understanding of some lower-level concepts that are obscured by some modern operating systems.

Course Content

Please look over the modules section of the canvas course for a list of the specific concepts, examples, and assignments we will be working on. Note that some material is adapted from a previous course, and will be updated as we go.

Bibliography

C programming: **Beej's guide to C programming (https://beej.us/guide/bgc/html/split/)**, available free online.

C++ programming: Wikibooks **C++ programming, (https://en.wikibooks.org/wiki/C%2B%2B_Programming)** available free online.

Java programming: Eck, David, **Introduction to Programming (https://math.hws.edu/javanotes-swing/)** in Java available free online.

Other sections of the course use this book, but there is not a free version online.

Gaddis and Muganda. Starting Out with Java: From Control Structures through Data Structures. Pearson.

To facilitate the transition from Python to Java, the committee recommends **Brad Miller's *Java for Python Programmers* (https://runestone.academy/ns/books/published/java4python/index.html)** which is available for free online. Miller's text provides a narrative overview that compares essential language differences. A shorter, side-by-side reference can be found on **Lambert's *From Python to Java* site. (http://kennethalambert.com/pythontojava/index.htm)**

Format

This course is intended to be a second course on computer programming. The student should already be comfortable with the basic ideas of programming in any language, including

- basic data types (int, float, string)
- operators (math operators, string operators, built-in functions)
- input and output (printing, input, basic file I/O)
- branching (if statement, conditions, if/else)
- looping (for loop, while loop)
- functions (defining a function, parameters, scope, return values)
- composite data structures (arrays, lists, and dictionaries)

Most students will have learned these ideas in Python, an excellent introductory language. However, the first language will not matter much, as long as the students have an understanding of these ideas. Throughout the course, we will reinforce these ideas by showing how they appear in different

languages, and how they changed through time. You will be re-introduced to these ideas as you see how they grew from very low-level languages like C into C++, Java, and back even into Python.

As the semester progresses, we will not only move into more modern languages, but we will be discovering new ways of thinking those languages afford. You will be exploring object-oriented programming, data structures and algorithms, and much more as we progress through the semester.

It will not matter much which exact style of programming you are already familiar with or wish to explore in the future. In this class we will be looking at fundamental ideas which appear in every style of programming.

Hands-on programming assignments (lab, homework, and/or larger projects) illustrate these topics and help build bridges between theory and practice. There will be programming assignments due every week, and you will be given some time in class to work on these assignments. However, you will still probably want more time to work on the assignment at home to get the greatest benefit from the projects. Some assignments may span over multiple weeks, but there will still be checkpoints to ensure you are moving in the right direction.

Course attendance is the number one predictor of success. We will cover a lot of territory, and we will move very quickly. If you do not keep up with the course, you will likely get very lost. Please make sure that attendance to all lecture and lab sessions is a priority.

Hardware requirement

The primary operating system for this course will be Linux. Early in the course, we will explain how to get to a Linux environment we will provide. We will also explain how to get to appropriate Linux installations on any devices you own. All assignments will be submitted through github, which is also both a universal standard for software development, and is available on any platform. Pretty much any device will work for this class, but you really will need at least a full keyboard. A tablet or chrome book is theoretically usable in this course but will be more frustrating than necessary. However, any laptop or desktop machine will be plenty.

Grading Scheme

My grading scheme reflects my teaching philosophy:

- If you are continuously turning in work, even if you struggle, you will not fail
- If you do all I ask for each assignment, you can easily earn a high B or an A
- Earning an A+ requires extra effort *beyond* the basic requirements of the assignment
- I reserve A+ grades for extraordinary work
- Note that BSU doesn't even have A+ listed as a possible final grade.

Project Extensions

- This will be a difficult course, and you will probably have at least one assignment not completed by the time it is due.
- Please turn in your project as is on time
- I **do not** mind incomplete unworking projects turned in on time.
- I can see where you're stuck and provide feedback
- You will likely even get a passing grade for the assignment if you're showing a reasonable amount of effort.
- However, I generally do not grant extensions to projects.
- If you are lost, more time will not help
- An extension in one project will amount to less time available for the next project.
- Repeated extensions usually turn into a learning deficit that cannot be made up at the end of the semester

Freedom of Expression

In this course, we are committed to fostering a learning environment that values intellectual diversity, encourages free expression, and promotes open inquiry. As members of the Ball State Community, we treat each person in the Ball State community with civility, courtesy, compassion, and dignity and respect and learn from differences in people, ideas, and opinions. Please review Ball State University's Statement on Freedom of Expression , the resources on Ball State's Freedom of Expression webpage , and Ball State's Beneficence Pledge .

Title IX

Ball State University is committed to providing a safe and inclusive learning environment for all students. If you or someone you know has experienced sexual harassment—including sexual assault, dating violence, domestic violence, or stalking, please know that you are not alone. The University offers support services and resources. For more information or to report an incident, please visit www.bsu.edu/titleix or contact the Title IX Coordinator at 765-285-1545 or at titleix@bsu.edu. As your instructor, I am a mandatory reporter under the Title IX policy and required to report any information I receive about possible sexual harassment. This includes information shared in class discussions, assignments, or private conversations.

What happens after I report? The Title IX Coordinator will email the person who experienced sexual harassment (complainant) and invite them to schedule a meeting. If the complainant chooses to meet with the Title IX Coordinator,

· Title IX Coordinator will offer supportive measures (e.g., counseling, extensions on deadlines, course-related adjustments, changes to work or class schedules, and/or referrals to campus offices), review the policy, and discuss options to move forward.

· If a complaint is filed by the complainant or the University, an investigation will begin following University policy.

University Grade Appeal Policy

If you believe you received a final course grade that does not reflect your performance due to fairness or a procedural issue, you have the right to file an appeal within 5 school days after final grades are posted by the Office of the Registrar. Review the University Grade Appeal Policy and Process.

Student Academic Ethics Policy

Honesty, trust, and personal responsibility are fundamental attributes of the university community. Academic dishonesty and other forms of academic misconduct threaten the

foundation of an institution dedicated to the pursuit of knowledge and will not be tolerated. To maintain its credibility and reputation, and to equitably assign evaluations of scholastic and creative performance, Ball State University is committed to maintaining a climate that upholds and values the highest standards of academic integrity. Visit the VPAA's academic integrity website (Students tab) for resources on understanding academic integrity, citing sources properly, and avoiding inadvertent academic dishonesty. To learn about BSU's academic integrity expectations and students' rights, please read the University Student Academic Ethics Policy.

Academic integrity violations include giving or receiving an unfair academic advantage (cheating), presenting someone else's ideas or work as your own (plagiarism), and falsifying academic records. Unless otherwise indicated, you must work independently by yourself. Check with me if you are unsure whether something constitutes academic dishonesty. Examples of academic integrity violations include but are not limited to:

· Using resources not authorized by the faculty member (including devices, AI tools, hidden notes, and open books)

· Using commercial study websites to find answers to graded assignments (Chegg, Course Hero, StudyPool, OneClass, etc.).

· Without the instructor's written permission, giving away, buying, or selling graded assignments class notes, exams, study guides, or other course materials to other students or to third-party vendors (Course Hero, Chegg, 24HourAnswers, etc.).

· Working with another person on any assignment other than authorized group projects.

· Sharing or allowing others to access your files, whether done with permission or not.

· Reusing your own work from another semester, course, or section.

· Sharing answers with others during exams (passing notes, texting, whispering, gesturing)

· Discussing exam questions and answers with students who have not taken the exam.

· Soliciting others to complete work for you.

Generative AI Statement

Academic honesty and AI

It is important to remember that the purpose of this course is NOT to produce software,. You will create programs throughout the semester, but the programs themselves are not the goal of this exercise. The purpose of this course is to change your brain just a little bit so that you can call yourself a quality programmer. The programs we write will be windows into your knowledge (and sometimes your frustration.) We do not require code because we want more computer programs in the world. We require code because we want to see what you understand and what you do not. For that reason, it is vital that any code you submit be your own work. If you turn in a brilliant program that you found on stack overflow or got ChatGPT to write for you, we defeat the purpose of you learning to code. Ultimately, you need to be able to explain every single line of code that you turn in. If you truly understand how it works, that's enough for me. But if you use strategies like copying from the Internet, using tools like Chegg, or using AI to help you write code, I can tell you a number of things:

- We'll almost certainly know. Code has a unique signature
- I may ask you to explain to me any code that you submit. If you cannot explain it, I will not consider it your code.
- AI at the moment writes code that's pretty easy to spot. It has some well-known idiosyncrasies. It also isn't guaranteed to work correctly, and almost never follows our style guidelines.
- Using such tools in a beginner course is against university policy and ethical standards
- We may refuse to grade any code you cannot explain
- Multiple submissions of code you do not understand could lead to academic misconduct issues

If you are lost, DO NOT use ChatGPT or the like. It has its uses in professional programming, but is not helpful at this point in your career. If you are lost on an assignment, make an honest effort, and turn in your mess. I promise to respect your honest efforts, even if the code looks like an absolute dumpster fire.

- If you truly tried to write the code and it doesn't work, that's fine
- As long as you documented well, you'll still probably get a passing grade

- We'll often allow resubmission after you figure out what is wrong
- Bad code that's yours is fine. Good code that isn't yours is a waste of everyone's time.

University Statement

We are committed to ensuring that all members of the community are welcome, through valuing the various experiences and worldviews represented at Ball State and among those we serve. We promote a culture of respect and civil discourse. If you need course adaptations or accommodations because of a disability, please contact the instructor of record as soon as possible. Ball State's Disability Services Office coordinates services for students with disabilities; documentation of a disability needs to be on file in that office before any accommodations can be provided. Disability Services can be contacted at 765-285-5293 or **dsd@bsu.edu (mailto:dsd@bsu.edu)** .

Attendance

The best way to learn any difficult skill is through regular practice, supervised by a knowledgeable coach. That's what classroom attendance is. While I do not intend to take formal attendance during each session, I can tell you that the best predictor of student success in this course is regular attendance to lectures and labs. You will have plenty of opportunities for guidance in the lab, and there is really no better way to learn. For formal university queries, I will use project submission dates as proof of last attendance.

Code of Student Rights and Responsibilities

We will uphold the BSU **Code of Student Rights and Responsibilities (http://cms.bsu.edu/about/administrativeoffices/studentrights/policiesandprocedures/studentcode)** . Of particular interest for this course are the following sections:

- Section V: **Code of Conduct ((http://cms.bsu.edu/about/administrativeoffices/studentrights/policiesandprocedures/studentcode/vcodeconduct)**
- Section VII: **Student Academic Ethics Policy ((http://cms.bsu.edu/About/AdministrativeOffices/StudentRights/PoliciesandProcedures/StudentCode/VIIEthicsPolicy.aspx)**

Evaluation of Student Performance

Programming projects provide one source of data for assessment of learning outcomes. These are graded on syntactic correctness, functional correctness, appropriate structure (object-oriented decomposition), appropriate style (naming conventions, documentation), and documentation of correctness (*e.g.* test cases).

These data are triangulated against other data collected by the instructor such as exams and essays.

Evaluation of the Course

In accordance with BSU policies and common academic practice, students will provide anonymous feedback using the University-approved evaluation instrument.