

# CPSC 24500: Object-Oriented Programming

## Week 1 Programming Assignment

# Week 1 Collaborative Assignment: Reverse-Engineer Wordle → Prompt an AI → Build a Modular Java Clone

---

### Purpose

You'll experience the modern software loop from observation to design to implementation. First, we'll study Wordle's behavior together. Then, in teams of four or five, you'll reverse-engineer how it likely works, translate that understanding into an engineering design, and craft an AI prompt that instructs a generative model to produce a clean, modular Java implementation. Much of your work will require you to iterate your prompting to drive the desired results. At no time will you intervene with the code; you must accept the code as the AI generates it for you. Your sole mode of intervention is through successive prompting. You'll end the week by presenting your software and walking the class through a brief, professional-style code review.

### In-Class Flow

- We'll start by playing the original Wordle together ([link here](#)), noting behaviors and edge cases.
- Teams will form—everyone must contribute to the final developed product.
- On paper, you'll model game state, the game loop, and how feedback is computed.
- Your team will then draft a concise design that separates responsibilities into coherent parts (e.g., game engine, input validation, feedback representation, and a simple UI).
- With that design in hand, you'll write your own generative-AI prompt, run it on the AI model of your choice, and iteratively refine the resulting code until it aligns with your design and runs as expected.

### Allowed Tools (pick any; disclose which you used)

You may use any reputable generative AI. Popular options include ChatGPT (OpenAI), Claude (Anthropic), Gemini (Google), GitHub Copilot or Copilot Chat (Microsoft/GitHub), and Perplexity. You're welcome to try others as well. You must include a short "AI usage disclosure" listing the model(s) you used and the exact prompts (or the core prompt if you iterated many times). You may also experiment with Vibe Coding.

### Deliverables

- Reverse-Engineering Packet (PDF, one or two pages) with your observed behaviors, a simple sketch of your architecture, and brief method/interaction notes.

# CPSC 24500: Object-Oriented Programming

## *Week 1 Programming Assignment*

- Final AI Prompt (plain-text) showing exactly how you instructed the model.
- Java source files that compile and run on a typical setup.
- Presentation slides (3–5) for the in-class code review.
- Note: No unit tests, formal README, or build systems are required this week.
- Every person on the team must contribute to the final project in some meaningful way.

### Guidance on Modular OOP Design (high-level—names are up to you)

- Aim for clear separation of concerns rather than one giant class.
- Typical pieces include something that owns game state and attempts, something that knows valid words and can pick a secret, something that validates input and normalizes case, a lightweight representation of per-letter feedback, and a simple renderer for console output.
- Favor small, focused methods, hide mutable state, keep I/O at the edges, and choose names that make intent obvious.
- The class and interface names are yours to design—justify them in your packet and presentation.

### How to Write Your Own AI Prompt (guidelines, not a template)

- Start from your observed rules and translate them into precise engineering language.
- Open with a one-sentence goal, then specify constraints (Java only, console program, standard JDK).
- Describe the modules you want and how they interact (responsibilities and data flow).
- State any style expectations (small methods, clear names, comments where helpful).
- List a few tricky scenarios you discovered during reverse-engineering (e.g., repeated letters or invalid guesses) and require the model to handle them.
- Request a short explanation of how the generated code maps to your modules so you can verify alignment with your design.
- Iterate: adjust wording when the output drifts from your design; keep the prompt concise and specific.

### Critical Notes:

- You must instruct the AI (via your prompting) that **"the end result must demonstrate good object-oriented programming and design."**
- **You must not tell AI that you are creating the game of Wordle.** AI will of course figure that out, but I want you to define the rules as part of your engineering work.

# CPSC 24500: Object-Oriented Programming

## *Week 1 Programming Assignment*

### Presentation & Code Review

Each team gets 6–8 minutes: a quick demo, then a walkthrough of your architecture diagram and one or two core methods that embody your design choices. In the review, we'll look for correctness relative to your observed behaviors, clarity and modularity of your design, naming and readability, input handling, and how well the generated code actually matches the prompt you wrote. Every teammate speaks briefly. Be prepared to explain any method if asked.

### Collaboration & Responsible AI Use

Use generative AI as a tool to implement your design, not as a substitute for thinking. Include your exact prompts and a brief note of any human edits you made after generation. You are responsible for understanding the code you submit; I may ask teammates to explain specific parts during the review.

### Grading Rubric

Due to our contract grading guidelines this project will be graded simply as “meets criteria” or “needs revision.” I will be looking for the following elements:

- Reverse-Engineering Packet demonstrating careful observation and edge-case thinking.
- AI Prompt quality—clarity, specificity, and alignment to your design.
- Java implementation—modularity, separation of concerns, and readability.
- Working demo and basic user experience.
- Presentation and code-review discussion.
- Demonstration of code adherence to object-oriented concepts.
- A list of group participants with their contributions to the project.

### Submission Notes

As your code will contain multiple files which you will load to GitHub. All other files (Word, PDF, or PowerPoint) must be included on GitHub with your submission. It would be acceptable to either create a sub folder below the main folder or to just include these extra files in the main folder. You will submit the GitHub link to Blackboard for grading.

### Final Thoughts

- It is okay if the code generated by AI does not work. Do your best to achieve the desired results though an iterative process of prompting. I do not want you to intervene in the code in any way—hence, no human touch.

# **CPSC 24500: Object-Oriented Programming**

## *Week 1 Programming Assignment*

- Use any generative AI of your choosing. You may also, with consensus from your team, experiment with Vibe Coding. If you are unfamiliar with this term, don't worry—we will discuss this later in the semester.
- You are not allowed to touch the code manually. You get what you get from AI. If you are not satisfied with the results, continue prompting until you achieve the results you desire.
- Ensure that you do not prompt for “Wordle”; you must specify what you wish to engineer solely through prompting.
- All code and supporting files will be stored in a GitHub repository.