Programming by Design: Computing, Representation, and Reasoning

Grade Level: 9th Grade — Full Year — Modular, Inquiry-Driven, Project-Based

Philosophy

Teach computer science as a lens for reasoning, systems thinking, and intellectual agency. Begin with functional reasoning, move through data modeling and control systems, and end with communication, networks, and real-world computational narratives.

Unit Overview

Unit 1: The Story of Data

Framing Concept: Programming is communication between humans and machines.

- Cultural/historical anchors: Ada Lovelace, Grace Hopper, underrepresentation
- Early routines: Computing in the News, What's Going On in This Graph?
- Mental models: abstraction, systems, structure
- No coding yet; set classroom norms and ways of thinking

Unit 2: Programming by Design (Racket)

Framing Concept: A program is a structured solution to a problem.

- Core practices: design recipe, decomposition, contracts, examples, tests
- Key concepts: pure functions, conditionals (`cond`), recursion
- Emphasizes algebraic reasoning and function modeling
- Visuals and math-aligned patterns (e.g., flag project)

Unit 3: Data Science and Representation (Pyret)

Framing Concept: Data is a constructed lens on the world.

- Functional transformations over tables: `filter`, `map`, `build-column`
- Boolean expressions, categories, visualizations
- Real datasets from museums and public archives
- Ends with synthesis and reflection on representation and omission
- Syntax shift from Racket to Pyret reinforces semantic continuity

Unit 4: Systems and Control (Python + EarSketch)

Framing Concept: Programs model dynamic systems through control flow and state.

- Intro to loops, mutable variables, accumulators

- Use EarSketch to introduce structure via audio: repetition, patterns
- Transition to standard Python: build simulations, rule engines
- Students experience stateful systems and algorithmic behavior

Unit 5: Interface and Communication (HTML + CSS)

Framing Concept: Code is interpretation—computational ideas need an audience.

- Learn structure and styling of the web
- Use static sites to present prior work (data, logic, projects)
- Emphasis on interpretability and design as argument

Unit 6: Code in the Wild (APIs + Jupyter Notebooks)

Framing Concept: Code lives in systems, documents itself, and pulls from the web.

- API calls: retrieve and process data (e.g., Met Museum API)
- Jupyter for literate computing: markdown + code + output
- Mini research project: question, investigate, explain with code
- Emphasizes reproducibility, analysis, and interdisciplinary reach

Unit 7: Networks, Protocols, and Power

Framing Concept: The internet is a system of abstractions that shapes global power.

- Explore IP, DNS, HTTP, and packets via simulation
- Command line basics: file navigation, permissions (if environment allows)
- Reflect on who owns infrastructure, metadata, and protocol standards
- Optional: version control basics (manual or Git)

Unit 8: Capstone Projects

Framing Concept: What can you build, explain, and defend?

- Design and build projects: data narratives, simulations, interactive sites
- Deliverables: artifact, technical documentation, self-reflection
- Optional showcase or community presentation

Threaded Themes & Practices

- Ethics and power in computing (bias, surveillance, automation)
- Design thinking, subgoal labeling, and structural clarity
- Weekly routines: journal writing, news analysis, graph interpretation

- Language comparisons to reinforce conceptual fluency

Pathway Alignment

- **AP CS A:** Object-oriented thinking, algorithm writing, Java readiness
- **Data Science + Capstone:** Investigative computing, research workflows
- **Algebra 2 + Computing:** Function modeling, logic, and systems thinking