Unit 3 Puzzles

Content Area: **Computer Science**
Course(s):
Time Period: **Marking Period 1**
Length: **10 days**
Status: **Published**

Summary

In Unit 3, students complete the course arc by becoming puzzle designers. Having practiced algorithmic thinking in Unit 1 and systems thinking through Dungeons & Dragons in Unit 2, students now apply both lenses to a form built specifically for another person to experience: the puzzle. The unit follows a solve–analyze–design arc across three phases. In Phase 1 (Days 1–4), students solve a wide range of puzzle types and build a shared vocabulary for what makes a puzzle satisfying, frustrating, or broken. In Phase 2 (Days 5–10), students go deep on five puzzle types in turn — logic grids, word puzzles, visual/spatial puzzles, ciphers, and math/pattern puzzles solving a model of each, deconstructing how it was built, and drafting an early design of their own, before spending Day 10 diagnosing intentionally broken versions of each type. In Phase 3 (Days 11–15), students design, build, self-test, and playtest an original physical puzzle pack of 4–5 puzzles for a real, known classmate audience, then revise based on structured peer feedback. Since students are designing for a specific partner who will actually attempt to solve their work, the unit raises the stakes on every computational thinking skill introduced earlier in the course: decomposition, pattern recognition, abstraction, algorithm design, and above all, debugging.

Revision Date: July 2026

Essential Questions

- What separates a good puzzle from a broken one?
- How does knowing exactly who will solve your puzzle change how you design it?
- Why doesn't testing your own puzzle tell you whether it actually works?
- Is a puzzle a system with one right answer, or is there room for creativity in how someone gets there?

Enduring Understandings

- Puzzle design is applied computational thinking: every puzzle encodes a solution path that a designer must build deliberately, verify rigorously, and debug when it fails.
- Designing well for a real audience requires anticipating how someone else's mind will approach a problem not just whether the designer personally finds it solvable.

Objectives

Students Will Know

- The defining elements of any puzzle: constraints, a goal, and a solution path.
- How five distinct puzzle types (logic, word, visual/spatial, cipher, and math/pattern) each encode and conceal their solution differently.
- Vocabulary connecting puzzle design to computational thinking: algorithm, conditional (if/then) logic, Boolean logic (AND, OR, NOT), decomposition, abstraction, pattern recognition, debugging, and iteration.
- The difference between a puzzle that is difficult and one that is broken a missing constraint, a contradictory clue, or more than one valid solution.
- The components of a complete puzzle submission: a solution key, a difficulty justification, and a Designer's Note explaining intent.

Students Will be Skilled At

- Solving and then deconstructing puzzles across five distinct types to identify each one's underlying mechanic.
- Designing an original puzzle with a single, verified, unique solution.
- Writing a solution key and a Designer's Note that clearly explains design intent.
- Collecting structured peer feedback using a Solver Feedback Form and revising a design in response to it.
- Diagnosing why a broken puzzle fails and identifying whether the fix requires a missing constraint, a contradictory clue, or a clearer rule.
- Constructing a finished, print-ready puzzle pack intended for a real, known audience.

Learning Plan

This unit is built around a single guiding principle: since students design puzzles for classmates who will actually attempt to solve them, the audience is real and known, which raises the stakes on every design decision in a way that designing for an anonymous or hypothetical solver never could. The unit unfolds across three phases over fifteen 40-minute class periods. Phase 1 (Days 1–4) immerses students as solvers and builds the shared critical vocabulary constraints, solution path, difficulty, fairness, uniqueness of solution — that the rest of the unit depends on. Phase 2 (Days 5–10) moves students from solver to analyst to early designer, one puzzle type at a time. Phase 3 (Days 11–15) is the capstone, in which students plan, build, self-test, playtest, and revise an original physical puzzle pack.

Phase 1 opens with students solving five or six deliberately different puzzle types as a class, a logic grid, a word puzzle, a visual puzzle, a cipher, and a physical or tactile challenge so that Day 1's discussion of what these had in common surfaces the unit's core definition: a puzzle is a designed experience with constraints, a goal, and a solution path. Day 2 turns to the psychology of puzzles, asking why solving something produces a satisfying “aha” moment and what tips a puzzle from rewarding into frustrating. Day 3 formally categorizes the puzzle types from Day 1 by their core mechanic and introduces the vocabulary of constraints, solution path, difficulty, fairness, and uniqueness of solution that recurs for the rest of the unit. Day 4 closes the phase by analyzing three puzzles side by side — one well designed, one too easy, and one broken so students begin to see every puzzle decision as a choice with a direct effect on the solver's experience.

Phase 2 devotes one day to each of five puzzle types, and every day follows the same solve-deconstruct-design rhythm established in Unit 1's capstone options. On Day 5 (Logic Puzzles), students solve a three-category logic grid as a class, identify which clues functioned as if/then eliminations, and then practice writing two original clues that would lead a solver to a given solution grid. On Day 6 (Word Puzzles), students solve a word search built with a second layer underneath it the leftover letters, marked with small numbered cells, spell a hidden message before sketching the requirements for a hidden-message puzzle of their own. On Day 7 (Visual & Spatial Puzzles), students solve a five-difference spot-the-difference pair and experiment with a simple diagonally-cut square puzzle, connecting geometric reasoning about area and congruent shapes to spatial puzzle design. On Day 8 (Cipher & Code Puzzles), students decode two Caesar-shift messages using a provided shift key, then choose their own shift value to encode a short message for a partner, making explicit that a cipher is simply an algorithm with a parameter. On Day 9 (Math & Pattern Puzzles), students find the rule behind two number sequences one governed by a conditional branching rule and one governed by a fixed non-linear pattern and attempt a small logic-and-arithmetic grid puzzle (cages that must sum or multiply to a target, with no repeated digit in any row or column), tying the day explicitly back to the Boolean logic and conditionals from Unit 1. Day 10 closes Phase 2 by handing students one intentionally broken version of each puzzle type an unsolvable logic grid, a cipher missing its key, an unfindable word search — so they practice the same diagnostic move from every prior unit: naming whether the failure is a logic error or a missing constraint, then rewriting the fix.

Phase 3 is the capstone. On Day 11, students choose four to five puzzle types for their pack, sketch a rough plan for each, and draft an opening Designer's Note, with the teacher circulating to approve plans before building begins; the plan must include at least three different puzzle types, at least one puzzle built on logic or Boolean reasoning, a genuine range of difficulty, and a solution key for every puzzle. Days 12–13 are build days: students construct their physical puzzle pack by hand, and every student is required to self-test each puzzle solving it start to finish themselves before Day 14, since students who skip this step reliably submit broken work. On Day 14, packs are exchanged, and solvers work through a partner's pack while the designer observes silently without offering hints, the same rule that has applied to partner testing since Unit 1's Debug Report and Unit 2's encounter playtesting; solvers then complete a Solver Feedback Form naming their favorite puzzle, any puzzle that felt broken or unfair, whether difficulty ratings were accurate, and one thing they would change. Day 15 is revision and submission: students revise based on that feedback and submit a complete pack Designer's Note, all puzzles, solution key, difficulty ratings with written justifications, and a written response to the Solver Feedback Form explaining what was changed and why.

Across all three phases, five computational thinking skills recur by design: decomposition (breaking a puzzle into its constraints, goal, and mechanic), pattern recognition (both while solving and while writing logic-grid

or math-pattern clues), abstraction (separating a puzzle's mechanic from its theme or "skin"), algorithm writing (every puzzle has a solution path the designer must construct and verify), and debugging (the Day 10 broken-puzzle diagnostics and the Day 15 revision cycle). The unit deliberately reuses vocabulary and structures from earlier in the course: the language of syntax versus logic errors carries forward from Unit 1's Debug Report, and the silent-observer rule during partner testing is identical to the dynamic used in Unit 1's robot test and Unit 2's encounter playtesting, so students recognize the pattern rather than learning a new protocol. Teachers should also make the Unit 2 connection explicit: designing an encounter that anticipates how different players might approach it and designing a puzzle that anticipates how a solver will get stuck are the same underlying skill applied to a different format.

Since Days 12–13 involve physical construction (cutting, drawing, assembling), teachers should have extra graph paper, rulers, colored pencils, index cards, and scissors on hand and build in buffer time, as construction reliably takes longer than students estimate. Early finishers in Phase 2 can be given an additional puzzle type to prototype or asked to test a classmate's early sketch. During Phase 3 build days, students who finish a puzzle early should be directed to begin their self-test rather than start a new puzzle, since a thoroughly tested four-puzzle pack is a stronger submission than an untested five-puzzle pack. Pair placement on Day 14 should mix students who chose different puzzle-type combinations so that everyone playtests a genuinely unfamiliar pack.

Assessment

Formative Assessment

Daily exit tickets check for understanding of that day's specific vocabulary and mechanic (for example, Day 5: "What makes a logic puzzle clue too easy? Too hard? Just right?"; Day 9: "What's the difference between a math puzzle and a math problem?"). Teacher circulation during Phase 2 design tasks and Phase 3 build days (Days 12–13) provides ongoing informal checks on whether a student's in-progress puzzle has a verifiable, unique solution before it reaches a partner.

Summative Assessment

The Final Puzzle Pack (Days 11–15) is the unit's summative performance assessment. Students submit a complete physical puzzle pack graded on a 20-point rubric across five equally weighted categories: Puzzle Variety & Range, Design Quality, Solution Key, Designer's Note, and Revision & Reflection.

Skill	4 — Exceeds	3 — Meets	2 — Approaching	1 — Beginning
Puzzle Variety	4–5 puzzles	4 puzzles with 3	3 puzzles with	Fewer than 3

& Range	spanning 3+ types with clear range of difficulty	types, mostly varied	limited variety	puzzles or all same type
Design Quality	Every puzzle is fair, solvable, and has exactly one solution	Most puzzles are fair and solvable with minor gaps	Some puzzles have structural issues	Puzzles are incomplete or broken
Solution Key	Complete, accurate, and clearly presented	Mostly complete with minor gaps	Present but incomplete	Missing or incorrect
Designer's Note	Clearly explains design choices, intended experience, and difficulty decisions	Explains most design choices with minor gaps	Vague or generic	Missing or off-base
Revision & Reflection	Response to feedback is specific, shows clear revision, and explains why changes were made	Response to feedback identifies changes clearly	Response to feedback is vague or minimal	No evidence of revision or reflection

Benchmark Assessment

Students' understanding will be assessed at various checkpoints during the cycle.

Alternative Assessment

The Day 13 self-test requirement and the Day 14 Solver Feedback Form both function as alternative, performance-based assessment: students receive real feedback from an authentic audience rather than only a teacher-assigned grade, and the Day 15 written response to that feedback is itself assessed as evidence of the debugging and iteration objectives.

Materials

Core Instructional Materials: Teacher-created puzzle activity sets for Days 5–9, each including a student activity sheet and teacher answer key: Day 5 Logic Puzzles, Day 6 Word Puzzles, Day 7 Visual & Spatial Puzzles, Day 8 Cipher & Code Puzzles, and Day 9 Math & Pattern Puzzles. Graph paper, rulers, scissors, colored pencils, and index cards for Phase 3 construction.

Supplemental Materials: Puzzle Baron (logic.puzzlebaron.com) for additional logic puzzle practice at varied difficulty levels; CS Unplugged for supplemental Boolean logic activities; Khan Academy for an optional geometry refresher (area, congruent figures) ahead of Day 7.

Standards

Computer Science (NJSLS-CSDT, 2020) — 8.1 Computer Science by the End of Grade 8

- 8.1.8.AP.1: Design and illustrate algorithms that solve complex problems using flowcharts and/or pseudocode.
- 8.1.8.AP.4: Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs.
- 8.1.8.AP.6: Refine a solution that meets users' needs by incorporating feedback from team members and users.
- 8.1.8.AP.8: Systematically test and refine programs using a range of test cases and users.
- 8.1.8.AP.9: Document programs in order to make them easier to follow, test, and debug.
- 8.1.8.CS.4: Systematically apply troubleshooting strategies to identify and resolve hardware and software problems in computing systems.

Design Thinking (NJSLS-CSDT, 2020) — 8.2 Design Thinking by the End of Grade 8

- 8.2.8.ED.2: Identify the steps in the design process that could be used to solve a problem.
- 8.2.8.ED.3: Develop a proposal for a solution to a real-world problem that includes a model (e.g., physical prototype, graphical/technical sketch).
- 8.2.8.ED.5: Explain the need for optimization in a design process.
- 8.2.8.ED.6: Analyze how trade-offs can impact the design of a product.
- 8.2.8.ED.7: Design a product to address a real-world problem and document the iterative design process, including decisions made as a result of specific constraints and trade-offs (e.g., annotated sketches).
- 8.2.8.NT.1: Examine a malfunctioning tool, product, or system and propose solutions to the problem.

Mathematics (NJSLS-Math, 2023) — Grade 7, sprinkled where geometric reasoning applies

- 7.G.A.2: Draw (with technology, with ruler and protractor, as well as freehand) geometric shapes with given conditions.

- 7.G.B.6: Solve real-world and mathematical problems involving area, volume, and surface area of two- and three-dimensional objects composed of triangles, quadrilaterals, polygons, cubes, and right prisms.

Career Readiness, Life Literacies & Key Skills (NJSLS, 2020)

- 9.4.8.CI.2: Repurpose an existing resource in an innovative way.
- 9.4.8.CI.3: Examine challenges that may exist in the adoption of new ideas.
- 9.4.8.CT.1: Evaluate diverse solutions proposed by a variety of individuals, organizations, and/or agencies to a problem and use critical thinking skills to predict which one(s) are likely to be effective.
- 9.4.8.CT.2: Develop multiple solutions to a problem and evaluate short- and long-term effects to determine the most plausible option.

Interdisciplinary Connections (English Language Arts, NJSLS-ELA)

- W.7.4: Produce clear and coherent writing in which the development, organization, and style are appropriate to task, purpose, and audience (Designer's Note, Solver Feedback responses).
- SL.7.1: Engage effectively in a range of collaborative discussions with diverse partners (partner playtesting debriefs, Day 4 and Day 10 class discussions).
- L.7.6: Acquire and accurately use grade-appropriate general academic and domain-specific words and phrases (algorithm, conditional, Boolean logic, decomposition, abstraction, debugging, iteration).

Additional Applicable Interdisciplinary Standards and Related Requirements

- Amistad Law (N.J.S.A. 18A:52-16.1 et seq.): Puzzle themes and flavor text (e.g., cipher messages, word search vocabulary) may highlight the contributions and lived experiences of diverse cultural groups, at the designer's choice.
- LGBTQ & Disabilities Law (P.L.2019, c.6 and c.170): When students select names, characters, or scenarios for their puzzle packs, teachers should encourage representation of diverse identities and abilities, consistent with district-wide inclusion mandates.
- Climate Change (NJSLS cross-content standard, 2020): Students may optionally theme a puzzle pack around an environmental or sustainability scenario (e.g., a logic grid about sorting recyclables correctly, or a cipher message that reveals a conservation fact), connecting design choices to real-world environmental content.

Integrated Accommodations & Modifications

Suggested Strategies for Modification for Computer Science