Unit 4: Standard Collection Algorithms

Content Area: **Computer Science**
Course(s):
Time Period: **Marking Period 1**
Length: **2-3 Weeks**
Status: **Published**

Summary

In programming, data is often stored in collections such as arrays, where the ability to organize and locate information quickly is essential. This section introduces standard algorithms used to process these collections, with a focus on searching and basic operations. Students will learn how to apply systematic approaches to find specific elements, analyze algorithm efficiency, and understand the trade-offs between different methods. These skills form the foundation for solving real-world problems where structured data must be accessed and manipulated effectively.

MA.9-12.1.2.12prof.Cr	Creating
MATH.K-12.1	Make sense of problems and persevere in solving them
ELA.L.SS.11–12.1	Demonstrate command of the system and structure of the English language when writing or speaking.
MATH.K-12.2	Reason abstractly and quantitatively
ELA.L.KL.11–12.2	Apply knowledge of language to understand how language functions in different contexts, to make effective choices for meaning or style, and to comprehend more fully when reading or listening.
MATH.K-12.5	Use appropriate tools strategically
MA.9-12.1.2.12prof.Cr2	Organizing and developing ideas.
MATH.K-12.6	Attend to precision
MATH.K-12.7	Look for and make use of structure
MATH.K-12.8	Look for and express regularity in repeated reasoning
CS.9-12.8.1.12.AP.1	Design algorithms to solve computational problems using a combination of original and existing algorithms.
CS.9-12.8.1.12.AP.3	Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice.
CS.9-12.8.1.12.AP.4	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
WRK.K-12.P.5	Utilize critical thinking to make sense of problems and persevere in solving them.
WRK.K-12.P.8	Use technology to enhance productivity increase collaboration and communicate effectively.
TECH.9.4.12.CT.1	Identify problem-solving strategies used in the development of an innovative product or practice (e.g., 1.1.12acc.C1b, 2.2.12.PF.3).
TECH.9.4.12.CT.2	Explain the potential benefits of collaborating to enhance critical thinking and problem solving (e.g., 1.3E.12profCR3.a).

Essential Questions / Enduring Understandings

Essential Questions:

- What algorithms would you choose to solve a given problem? Why?
- How do you traverse an array?
- How are elements inserted into an already established array?
- How are elements deleted from an already established array?
- What is an abstract data type?
- How do you search an array?
- Which search method is more efficient?
- How do you sort an array?
- Which sort method is best?
- Which search method is more efficient?

Enduring Understandings:

- Arrays can be manipulated.
- Different search algorithms exist and when best to use each.
- Different sort algorithms exist and when best to use each.

Objectives

Students will know:

- operations on arrays.
- how to traverse an array.
- how to insert elements into an array.
- how to delete elements from an array.
- how to sort an array using either bubble, selection, insertion, or merge sort algorithms.
- how to perform a sequential search.
- how to perform a binary search.

Students will be skilled at:

- recognizing sorting algorithms.
- iterating through a collection of data.

Learning Plan

- Preview the essential questions and connect to learning throughout the unit.
- Algorithm for accessing arrays.
- Traversing arrays.
- Insertion and deletion from an array.
- Selection and insertion of sorts.
- Merge sort (recursion - optional)
- Discussion of efficiency of the three different sorts.
- Searches (sequential and binary).
- Identify boundary cases for programs and how best to test those boundary cases.
- Discuss the use of pre- and post-condtions when writing programs. Using already written programs, have students identify the pre- and post-conditions.

Assessment

- Assessments
    - Formative: Daily assessments using examples from class notes and CodeHS.com, AP Classroom/Albert Checks for Understanding
    - Summative: Teacher-created assessments/projects and CodeHS Computer Science Projects, AP Classroom/Albert Unit Assessments
    - Benchmark: Check for understanding benchmark assessments on CodeHS, AP Classroom/Albert/Khan Academy Diagnostics
    - Alternative Assessments: Student-centered activities such as a doorbell coding project, game design projects, and other activities involving real world applications
        - Complete quizzes/test: Algorithms, Structure of Programs, Design of Programs
        - Be observed by the teacher during individual work on the performance tasks.
        - Conduct self-assessments and reflections
        - Conduct Peer Evaluations.

Materials

- Core instructional materials: Core Book List

- Supplemental materials: CodeHS, computers, and reference books.

Integrated Accommodation and Modification

See Linked Document.