



West Windsor - Plainsboro Regional School District

Advanced Topics in Computer Science

Unit 0: Computer Science

Content Area: Computer Science

Course & Grade Level: Computer Science - Grade 12

Summary and Rationale

The West Windsor-Plainsboro Regional School District recognizes the importance of the study 21st Century Life and Careers standards. Additionally, it is also believed this learning should not be taught in isolation and cross curricular and career ready practices are embedded in every unit of study. Unit 0 is incorporated into each unit of study of this curricular document.

Recommended Pacing:

ELA Companion Standards and Career Ready Practices will be integrated throughout all units of study.

Interdisciplinary Connections

Grades 9-10

Progress Indicators Reading Science and Technical Subjects

Key Ideas and Details

RST.9-10.1. Accurately cite strong and thorough evidence from the text to support analysis of science and technical texts, attending to precise details for explanations or descriptions.

RST.9-10.2. Determine the central ideas, themes, or conclusions of a text; trace the text's explanation or depiction of a complex process, phenomenon, or concept; provide an accurate summary of the text.

RST.9-10.3. Follow precisely a complex multistep procedure when carrying out experiments, taking measurements, or performing technical tasks, attending to special cases or exceptions defined in the text.

Craft and Structure

RST.9-10.4. Determine the meaning of symbols, key terms, and other domain-specific words and phrases as they are used in a specific scientific or technical context relevant to *grades 9-10 texts and topics*.

RST.9-10.5. Analyze the relationships among concepts in a text, including relationships among key terms (e.g., *force, friction, reaction force, energy*).

RST.9-10.6. Determine the author's purpose in providing an explanation, describing a procedure, or discussing an experiment in a text, defining the question the author seeks to address.

Integration of Knowledge and Ideas

RST.9-10.7. Translate quantitative or technical information expressed in words in a text into visual form (e.g., a table or chart) and translate information expressed visually or mathematically (e.g., in an equation) into words.

RST.9-10.8. Determine if the reasoning and evidence in a text support the author's claim or a recommendation for solving a scientific or technical problem.

RST.9-10.9. Compare and contrast findings presented in a text to those from other sources (including their own experiments), noting when the findings support or contradict previous explanations or accounts.

Range of Reading and Level of Text Complexity:

RST.9-10.10. By the end of grade 10, read and comprehend science/technical texts in the grades 9-10 text complexity band independently and proficiently.

Career Ready Practices

CRP1. Act as a responsible and contributing citizen and employee.

CRP2. Apply appropriate academic and technical skills.

CRP3. Attend to personal health and financial well-being.

CRP4. Communicate clearly and effectively and with reason.

CRP5. Consider the environmental, social and economic impacts of decisions.

CRP6. Demonstrate creativity and innovation.

CRP7. Employ valid and reliable research strategies.

CRP8. Utilize critical thinking to make sense of problems and persevere in solving them.

CRP9. Model integrity, ethical leadership and effective management.

CRP10. Plan education and career paths aligned to personal goals.

CRP11. Use technology to enhance productivity.

CRP12. Work productively in teams while using cultural global competence.

Competencies for 21st Century Learners

X	Collaborative Team Member	X	Effective Communicator
X	Globally Aware, Active, & Responsible Student/Citizen	X	Information Literate Researcher
X	Innovative & Practical Problem Solver	X	Self-Directed Learner

Advanced Topics in Computer Programming	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
<p>Advanced Topics in Computer Science is the second course of a two-year college level sequence in program design, implementation, and testing using the structured language Java. It is designed for students who have completed AP Computer Science and have a strong background in both mathematics and object oriented programming. The course investigates abstract data types, how those structures extend algorithmic design, and the primary sorting, searching, and hashing techniques. At the conclusion of the course, students will be able to compare the various structures and strategies presented and selectively choose appropriate data types for given applications, analyzing the pros and cons of each, in order to match types to each application.</p>	
Recommended Pacing	
135 days	
ISTE National Educational Technology Standards	
<p>CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.</p>	
<p>CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels--machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).</p>	

Unit 1: Recursion	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
<p>Recursion is a technique that allows us to break down a problem into smaller pieces. The process consists of the same computation recurring repeatedly until the problem is solved. This approach can be applied to many types of problems. Often it is difficult to solve problems without using recursion. We will examine both simple and complex examples of recursion as we learn how to “think recursively”.</p>	
Recommended Pacing	
15 days	
ISTE National Educational Technology Standards	
<p>CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.</p>	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-I.A.1	Demonstrate knowledge of and skill regarding the syntax and semantics of a high level programming language, its control structures, and its basic data representations.
CS-I.A.3	Demonstrate knowledge of and skill regarding program correctness issues and practices (e.g., testing program results, test data design, loop invariants).
CS-I.A.4	Design, implement, and test programs of sufficient complexity to demonstrate knowledge and skills included in CS-I.A.1
<p>CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels--machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).</p>	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-III.1	Describe how data is represented at the machine level (e.g., character, boolean, integer, floating point)
CS-III.2	Identify and provide usage examples of the various data structures and files provided by a programming language (e.g., objects, various collections, files)
CS-III.3	Describe the elements (people, hardware, software, etc.) and their interactions within information systems (database systems, the Web, etc.)

Instructional Focus			
Unit Enduring Understandings			
<ul style="list-style-type: none"> • Recursion uses stacks to keep track of active values in previous recursive calls. • A recursive solution is normally slower and uses more memory than its iterative counterpart, and sometimes is very much slower. • Helper methods can make recursive solutions more efficient by changing or expanding the parameter list. • Backtracking examines partial solutions, abandoning unsuitable ones and returning to consider other potential solutions. 			
Unit Essential Questions			
<ul style="list-style-type: none"> • How can programmers keep track of the values in a method that is called recursively multiple times? • What is infinite recursion and what is its result? • If a problem can be solved iteratively, can it also be solved recursively? • How can recursion organize and solve problems that seemingly are solved only by trial and error methods? 			
Objectives			
Students will know: <ul style="list-style-type: none"> • Recursive computation solves a problem by using the solution to the same problem with successive inputs. • For a recursive call to terminate, a base case for the simplest values must exist. • It is sometimes easier to find a recursive solution if you make a slight change to the original problem. Students will be able to: <ul style="list-style-type: none"> • Compare the efficiency of a recursive solution to an iterative counterpart, and use the results to increase the efficiency of the recursive solution. • Use backtracking to escape from mazes, or find solutions to systems that are constrained by rules. 			
Evidence of Learning			
Assessment			
Common Assessment 1.1 - Recursion Backtracking Lab Recursion Lab			
Competencies for 21st Century Learners			
✓	Collaborative Team Member		Effective Communicator
	Globally Aware, Active, & Responsible Student/Citizen	✓	Information Literate Researcher
✓	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley)			

Unit 2: Sorting, Searching, and Algorithm Efficiency	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
Sorting and searching data are two of the most common tasks in computer science. Many different algorithms have been created to successfully and efficiently sort data and search through its contents. We will study these algorithms and compare their performance.	
Recommended Pacing	
10 days	
ISTE National Educational Technology Standards	
CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-I.A.1	Demonstrate knowledge of and skill regarding the syntax and semantics of a high level programming language, its control structures, and its basic data representations.
CS-I.A.3	Demonstrate knowledge of and skill regarding program correctness issues and practices (e.g., testing program results, test data design, loop invariants).
CS-I.A.4	Design, implement, and test programs of sufficient complexity to demonstrate knowledge and skills included in CS-I.A.1
CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels--machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-III.1	Describe how data is represented at the machine level (e.g., character, boolean, integer, floating point)
CS-III.2	Identify and provide usage examples of the various data structures and files provided by a programming language (e.g., objects, various collections, files)
Instructional Focus	
Unit Enduring Understandings	
<ul style="list-style-type: none"> ● Big-O notation measures the efficiency of an algorithm. ● There are advantages to using Linear Search vs Binary Search and vice versa. ● There are advantages to using Quadratic Sorts vs Recursive Sorts and vice versa. 	

Unit Essential Questions			
<ul style="list-style-type: none"> How can we quantify the efficiency of an algorithm? Which sorting or searching algorithms are most efficient? What are the best/worst case scenarios when sorting/searching data? 			
Objectives			
Students will know: <ul style="list-style-type: none"> There are various methods to search through large amounts of data. The capabilities and implementations of iterative and recursive data sorts. That algorithm complexity can be expressed using Big-O notation. Students will be able to: <ul style="list-style-type: none"> Compare/Contrast the efficiency of sorting and searching algorithms. Choose which is the best sort or search algorithm to use in specific situations. 			
Evidence of Learning			
Assessment			
Common Assessment 2.1 - Sorting and Searching Sort Efficiency Lab and Analysis Custom Sort Lab			
Competencies for 21st Century Learners			
✓	Collaborative Team Member		Effective Communicator
	Globally Aware, Active, & Responsible Student/Citizen	✓	Information Literate Researcher
✓	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley) Suggested Resources:			

Unit 3: Linked Lists	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
A linked list is a valuable data structure that allows you to add and remove elements efficiently without moving any of the other elements in the list. We will explore several implementations of linked lists and discuss the efficiency of its operations.	
Recommended Pacing	
22 days	
ISTE National Educational Technology Standards	
CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-I.A.1	Demonstrate knowledge of and skill regarding the syntax and semantics of a high level programming language, its control structures, and its basic data representations.
CS-I.A.2	Demonstrate knowledge of and skill regarding common data abstraction mechanisms (e.g., data types or classes such as stacks, trees, etc.)
CS-I.A.3	Demonstrate knowledge of and skill regarding program correctness issues and practices (e.g., testing program results, test data design, loop invariants).
CS-I.A.4	Design, implement, and test programs of sufficient complexity to demonstrate knowledge and skills included in CS-I.A.1
CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels--machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-III.2	Identify and provide usage examples of the various data structures and files provided by a programming language (e.g., objects, various collections, files)
CS-III.3	Describe the elements (people, hardware, software, etc.) and their interactions within information systems (database systems, the Web, etc.)

Instructional Focus			
Unit Enduring Understandings			
<ul style="list-style-type: none"> • A linked list has a number of nodes, each of which stores an object and a reference to the next node. • A linked list can be an efficient way to store and access data. • List iterators are used to access elements of a linked list. 			
Unit Essential Questions			
<ul style="list-style-type: none"> • What is a linked list and how can we implement it? • What is a doubly linked list? • What is a circular linked list? • What are the advantages/disadvantages of using doubly or circular linked lists? What are good examples for using either implementation? • What applications are best implemented with arrays? What applications are best implemented with linked lists? 			
Objectives			
Students will know: <ul style="list-style-type: none"> • Java's linked list collection implements the list interface. • The capabilities and implementations of linked lists. • The Big-O running time of linked list algorithms. Students will be able to: <ul style="list-style-type: none"> • Use a linked list to add, remove, and access elements. • Compare/Contrast the efficiency of array list and linked list algorithms. • Compare the efficiency of different implementations of linked lists. • Determine best test cases to troubleshoot list implementations. 			
Evidence of Learning			
Assessment			
Common Assessment 3.1 - Linked Lists			
Linked List Modeling			
Linked List Implementation Lab			
Competencies for 21st Century Learners			
✓	Collaborative Team Member		Effective Communicator
	Globally Aware, Active, & Responsible Student/Citizen	✓	Information Literate Researcher
✓	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley)			

Unit 4: Stacks and Queues	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
Abstract data types provide the user with a basis of what should be included in specific classes. Stacks and queues are versatile abstract data structures. We will implement several stack and queue programs and compare their efficiency. Many useful applications of stacks and queues will be discussed.	
Recommended Pacing	
24 days	
ISTE National Educational Technology Standards	
CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-I.A.1	Demonstrate knowledge of and skill regarding the syntax and semantics of a high level programming language, its control structures, and its basic data representations.
CS-I.A.2	Demonstrate knowledge of and skill regarding common data abstraction mechanisms (e.g., data types or classes such as stacks, trees, etc.)
CS-I.A.3	Demonstrate knowledge of and skill regarding program correctness issues and practices (e.g., testing program results, test data design, loop invariants).
CS-I.A.4	Design, implement, and test programs of sufficient complexity to demonstrate knowledge and skills included in CS-I.A.1
CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels-- machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-III.1	Describe how data is represented at the machine level (e.g., character, boolean, integer, floating point)
CS-III.2	Identify and provide usage examples of the various data structures and files provided by a programming language (e.g., objects, various collections, files)
CS-III.3	Describe the elements (people, hardware, software, etc.) and their interactions within information systems (database systems, the Web, etc.)

Instructional Focus			
Unit Enduring Understandings			
<ul style="list-style-type: none"> • An abstract class defines what operations should be supported but leaves the implementation unspecified. • A stack is a collection of objects with last in first out property. • A queue is a collection of objects with first in first out property. • A priority queue is a queue where an element with high priority is served before an element with low priority. 			
Unit Essential Questions			
<ul style="list-style-type: none"> • What is an abstract type? • What are stacks and queues and how are they implemented? • What differentiates a priority queue and what structures can implement it? • What types of applications are better solved with stacks? Which are better with queues? 			
Objectives			
Students will know: <ul style="list-style-type: none"> • The capabilities and implementations of stacks and queues. • The organization of max and min heaps and their use in priority queues. • The capabilities and implementations of priority queues. • The Big-O running time of stack and queue algorithms. Students will be able to: <ul style="list-style-type: none"> • Use linked lists, stacks, and priority queues to add and remove elements from a collection. • Compare the efficiency of different implementations of stacks and queues. • Determine which collection is the most efficient to use in various situations. • Implement a priority queue and discuss its efficiency. 			
Evidence of Learning			
Assessment			
Common Assessment 4.1 - Stacks and Queues			
Stack Implementation Lab			
Queue Implementation Lab			
Competencies for 21st Century Learners			
✓	Collaborative Team Member		Effective Communicator
	Globally Aware, Active, & Responsible Student/Citizen	✓	Information Literate Researcher
✓	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley)			
Suggested Resources:			

Unit 5: Sets, Maps, and Hash Tables	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
The technique of hashing can be used to find elements in a data structure quickly. We will use this hash function to implement the abstract set and map data types. We will implement several set and map programs and compare their efficiency. Many useful applications of sets and maps will be discussed.	
Recommended Pacing	
25 days	
ISTE National Educational Technology Standards	
CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-I.A.1	Demonstrate knowledge of and skill regarding the syntax and semantics of a high level programming language, its control structures, and its basic data representations.
CS-I.A.2	Demonstrate knowledge of and skill regarding common data abstraction mechanisms (e.g., data types or classes such as stacks, trees, etc.)
CS-I.A.3	Demonstrate knowledge of and skill regarding program correctness issues and practices (e.g., testing program results, test data design, loop invariants).
CS-I.A.4	Design, implement, and test programs of sufficient complexity to demonstrate knowledge and skills included in CS-I.A.1
CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels--machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-III.2	Identify and provide usage examples of the various data structures and files provided by a programming language (e.g., objects, various collections, files)
Instructional Focus	
Unit Enduring Understandings	
<ul style="list-style-type: none"> Sets are collections of elements with no order and no duplicates. Maps are associations between keys and values. A good hash function minimizes collisions. Performance of hash tables using open addressing are affected by load factors. 	

Unit Essential Questions			
<ul style="list-style-type: none"> • What is a hash code and why is it important to create unique hash functions? • What is a collision and why does it affect performance? • Why do sets extend a Java Collection but maps do not? • Why is uniqueness important with sets? 			
Objectives			
Students will know: <ul style="list-style-type: none"> • The Big-O running time often depends upon the strategy implemented for hash collision resolution. • Hash maps and hash sets use arrays to arrange data. • Advantages of hash tables compared to other structures, including Big-O running time. Students will be able to: <ul style="list-style-type: none"> • Develop effective hash codes by writing their own hash functions. • Choose appropriate implementations from the Java Collections framework for solving programming problems. • Use load factors to dynamically resize hash tables. 			
Evidence of Learning			
Assessment			
Common Assessment 5.1- Sets and Maps Hash Table Implementation Lab Stack Implementation Lab Queue Implementation Lab			
Competencies for 21st Century Learners			
✓	Collaborative Team Member		Effective Communicator
	Globally Aware, Active, & Responsible Student/Citizen	✓	Information Literate Researcher
✓	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley)			

Unit 6: Trees	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
Binary trees organize data hierarchically, resembling a tree. This structure allows for excellent performance for adding, removing, and finding elements. The architecture of binary trees allows for recursive traversals of the data. We will implement several binary trees and compare their efficiency. Many useful applications of trees will be covered.	
Recommended Pacing	
15 days	
ISTE National Educational Technology Standards	
CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-I.A.1	Demonstrate knowledge of and skill regarding the syntax and semantics of a high level programming language, its control structures, and its basic data representations.
CS-I.A.2	Demonstrate knowledge of and skill regarding common data abstraction mechanisms (e.g., data types or classes such as stacks, trees, etc.)
CS-I.A.3	Demonstrate knowledge of and skill regarding program correctness issues and practices (e.g., testing program results, test data design, loop invariants).
CS-I.A.4	Design, implement, and test programs of sufficient complexity to demonstrate knowledge and skills included in CS-I.A.1
CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels--machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-III.2	Identify and provide usage examples of the various data structures and files provided by a programming language (e.g., objects, various collections, files)

Instructional Focus			
Unit Enduring Understandings			
<ul style="list-style-type: none"> Trees are composed of nodes, each of which can have up to two child nodes. The balance of a tree determines its Big-O running time. Many tree properties are computed with recursive methods. In a balanced tree, all paths from the root to the leaves have approximately the same length. 			
Unit Essential Questions			
<ul style="list-style-type: none"> How are tree structures organized and implemented? What is a node? How can nodes relate with other nodes? What are costs and benefits of balancing trees? What is the difference between a tree, a binary tree, and a balanced binary tree? 			
Objectives			
Students will know: <ul style="list-style-type: none"> Tree maps and tree sets use binary trees to arrange data. Removing nodes requires different implementations depending on how many children the node has. Visiting all elements in a tree starts with the root and then recursively visiting all subtrees. Roots of trees can be removed or replaced. Students will be able to: <ul style="list-style-type: none"> Create a binary search implementation in a binary tree. Traverse a binary tree recursively using a variety of different orderings. Implement a binary tree that includes removal, adding, and balancing. Describe which types of trees guarantee Big-O running time of $\log(n)$ and why. 			
Evidence of Learning			
Assessment			
Common Assessment 6.1- Binary Trees			
Binary Search Tree Implementation Lab			
Common Assessment 6.1			
Competencies for 21st Century Learners			
✓	Collaborative Team Member		Effective Communicator
	Globally Aware, Active, & Responsible Student/Citizen	✓	Information Literate Researcher
✓	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley)			

Unit 7: Heaps and Heap Sort	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
A heap is a binary tree structure that is well suited to sort objects and implement priority queues. We will examine the heap sort and compare its efficiency to the other sorts that we studied earlier in the course. We will also use heaps to implement several types of priority queues and compare their efficiency.	
Recommended Pacing	
10 days	
ISTE National Educational Technology Standards	
CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-I.A.1	Demonstrate knowledge of and skill regarding the syntax and semantics of a high level programming language, its control structures, and its basic data representations.
CS-I.A.2	Demonstrate knowledge of and skill regarding common data abstraction mechanisms (e.g., data types or classes such as stacks, trees, etc.)
CS-I.A.3	Demonstrate knowledge of and skill regarding program correctness issues and practices (e.g., testing program results, test data design, loop invariants).
CS-I.A.4	Design, implement, and test programs of sufficient complexity to demonstrate knowledge and skills included in CS-I.A.1
CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels-- machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-III.1	Describe how data is represented at the machine level (e.g., character, boolean, integer, floating point)
CS-III.2	Identify and provide usage examples of the various data structures and files provided by a programming language (e.g., objects, various collections, files)
Instructional Focus	
Unit Enduring Understandings	
<ul style="list-style-type: none"> Removing a max or min from a heap requires replacement of the root. 	

<ul style="list-style-type: none"> • A min-heap is binary tree that is almost completely filled and every node value is no larger than the values of all its descendants. 			
Unit Essential Questions			
<ul style="list-style-type: none"> • Why are heaps used to implement priority queues? • What are minheaps and maxheaps? • Why can heaps and their nodes be stored efficiently in arrays? 			
Objectives			
Students will know: <ul style="list-style-type: none"> • The shape of a heap is very regular. • In a min-heap, both the left and right subtrees are larger than the root. • Storing heap values in arrays is very efficient. Students will be able to: <ul style="list-style-type: none"> • Create an implementation of a heap. • Compare the efficiency of heap sort to other sorting algorithms. • Implement a priority queue using a heap. • Compare/Contrast heaps to other binary trees. 			
Evidence of Learning			
Assessment			
Common Assessment 7.1- Heaps			
Heap Sort Implementation Lab			
Heap Priority Queue Implementation Lab			
Competencies for 21st Century Learners			
✓	Collaborative Team Member		Effective Communicator
	Globally Aware, Active, & Responsible Student/Citizen	✓	Information Literate Researcher
✓	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley)			
Suggested Resources:			

Unit 8: Graphs and Graph Theory	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
<p>Graphs are a powerful and versatile data structures that demonstrate relationships between different types of data. The data is stored in vertices and these vertices are connected with edges. We will discuss several implementations of graphs and their applications.</p>	
Recommended Pacing	
8 days	
ISTE National Educational Technology Standards	
<p>CS Standard I. Programming and Algorithm Design CS. Students will demonstrate proficiency in programming that requires the use of data abstraction to solve non-trivial programming problems in multiple programming paradigms.</p>	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-I.A.2	Demonstrate knowledge of and skill regarding common data abstraction mechanisms (e.g., data types or classes such as stacks, trees, etc.)
CS-I.A.3	Demonstrate knowledge of and skill regarding program correctness issues and practices (e.g., testing program results, test data design, loop invariants).
<p>CS Standard III. Data Representation and Information Organization. Students will demonstrate an understanding of data and information representation and organization at a variety of levels--machine level representation (for program correctness); data structures (for program implementation); problem representation (for solution design); files and databases (for general applications); and interactions among systems and people (for overall system design and effectiveness).</p>	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-III.2	Identify and provide usage examples of the various data structures and files provided by a programming language (e.g., objects, various collections, files)
CS-III.3	Describe the elements (people, hardware, software, etc.) and their interactions within information systems (database systems, the Web, etc.)
Instructional Focus	
Unit Enduring Understandings	
<ul style="list-style-type: none"> • Graphs are commonly implemented using tree maps. • Shortest paths, including mazes, can be solved using graphs and recursive methods. • Weighted graphs allow edges to be valued, permitting calculation and comparison of distance and time. • Connections between nodes can be direct or indirect. 	

Unit Essential Questions			
<ul style="list-style-type: none"> • What are edges and nodes and how are they related? • What data type structures can be used to implement graphs? • What types of values can be added to edges to increase functionality? • How can graphs be used to find the quickest path between two locations? 			
Objectives			
Students will know: <ul style="list-style-type: none"> • Graphs are connected groups of vertices and edges. • Adjacent nodes imply connectedness, but connectedness does not imply adjacency. • Only undirected graphs allow bidirectional traversal of edges between nodes. • Graphs be expressed using an adjacency matrix or adjacency list. Students will be able to: <ul style="list-style-type: none"> • Implement algorithms with graphs to search and find shortest routes. • Use recursive methods to compute lists of vertices and edges. • Enumerate the total number of paths allowable from a given node to another node. • Determine if a graph is cyclic. 			
Evidence of Learning			
Assessment Common Assessment 8.1- Graphs Graph Implementation Lab			
Competencies for 21st Century Learners			
✓	Collaborative Team Member		Effective Communicator
	Globally Aware, Active, & Responsible Student/Citizen		Information Literate Researcher
✓	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley)			

Unit 9: Computing and Society	
Content Area: Technology	
Course & Grade Level: Advanced Topics in Computer Science, 10-12	
Summary and Rationale	
In this unit, we will examine how the concepts of this course are utilized throughout society. We will also examine some of the ethical issues that arise from increased use of computing.	
Recommended Pacing	
6 days	
ISTE National Educational Technology Standards	
<p>CS Standard IV. Social Aspects of Computing. We live within a cultural environment and interact daily with other people. Computing specialists need to communicate and work with each other and with non-specialists. Specialists and non-specialists need to be cognizant of issues and risks related to computing in our society and to learn independently as new developments in technology arise. Students will demonstrate skills and understanding relative to social aspects of computing that are appropriate for specialists and non-specialists. In order to make informed decisions regarding computing in their personal lives and with respect to societal laws and norms, students will demonstrate an understanding of computing and potential issues and skill at recognizing, researching, and analyzing issues to reach defensible conclusions.</p>	
CSPI #	CS-I. Performance Indicators (CSPI)
CS-IV.A.1	Demonstrate awareness of social issues related to the use of computers in society and principles for making informed decisions regarding them (e.g., security, privacy, intellectual property, equitable access to technology resources, gender issues, cultural diversity, differences in learner needs, limits of computing, rapid change)
CS-IV.A.2	Analyze various social issues involving computing, producing defensible conclusions
CS-IV.A.3	Demonstrate an understanding of significant historical events relative to computing
Instructional Focus	
Unit Enduring Understandings	
<ul style="list-style-type: none"> There are many varied careers that deal with computing. Developers in the real world must consider many questions beyond “Will it work?” when designing products. Complexity of human endeavors proves enormously complex to code by hand, so other methods of coding must be developed to allow machines to learn quickly. 	
Unit Essential Questions	
<ul style="list-style-type: none"> Who controls the internet? What is meant by privacy in the computer age? How is artificial intelligence used in today’s world? 	

Objectives			
Students will know: <ul style="list-style-type: none"> • Application of new technology often involves unforeseen legal aspects and consequences. • Systems and products undergo positive and negative influences as a result of industry standardization. • Benefits and pitfalls of open source and commercial software systems affect both business and consumers. Students will be able to: <ul style="list-style-type: none"> • Explore practical applications of data structures. • Discuss ethical implications computing and technology. • Research careers in computing and technology. 			
Evidence of Learning			
Assessment			
Final Project			
Competencies for 21st Century Learners			
✓	Collaborative Team Member	✓	Effective Communicator
✓	Globally Aware, Active, & Responsible Student/Citizen	✓	Information Literate Researcher
	Innovative & Practical Problem Solver	✓	Self-Directed Learner
Resources			
Core Text: Big Java, 2nd Edition, Horstman (Wiley)			