Summit Public Schools
Summit, New Jersey
Grade Level / Content Area: Mathematics
Length of Course: 1 Academic Year
Curriculum: AP Computer Science A
Updated: Spring 2019

Developed By
Brian Weinfeld

Course Description: **AP Computer Science A is a full year course designed to prepare students for the AP Computer Science A examination. Throughout the course, students will review and extend previously learned programming skills as well as learn new advanced concepts in order to prepare them for this exam. Students will gain a variety of Java skills that serve not only to enhance their programming ability but also to assist them in their conceptual understanding of programming as a whole. Upon completion of this course students will be prepared to continue their computer science studies in an advanced level class or college setting.**

Anticipated Timetable for AP Computer Science

QUARTER 1

Unit 0 – Review of Object Oriented Programming Design

Topic	Time Frame
Review of basic statements syntax including int, double, if and while	2
Review of basic class creation	1
Review of the difference between, compile-time, run-time and logic errors	1
Review of terminology for key parts of a program (constructor, variables, etc.)	1
Total	5

Unit 1 – Utilizing Primitive Data Types and Conditional Statements

Topic	Time Frame
Review of int/double/boolean and lesson on memory allocation of each type	1
Lesson on how int and double interact with each other. (casting)	1
Lesson on assignment operators and relational operators, increment operators, etc.	1
Introduction to truth tables and compound Boolean statements	1
Quiz: Truth tables and basic math commands with int/double	1
Review of core control structures (while, if and for) and when to use each.	1
Lesson on nested control statements	1
Project #1: Control Statement and Math Project	3
Total	10

Unit 2 – Review of Classes

Topic	Time Frame
Review of class creation and terminology (constructor, etc.)	2

Lesson on constructors ("this" and defaults)	2
Lesson on method overloading	1
Lesson on "final" and utilizing "public" vs. "private", "static"	2
Quiz: Class creation and definitions	1
Introduction to passing by reference vs. passing by value	1
Exploration of memory allocation and classes	1
Exploration of "null" keyword and passing "null" parameters	1
Project #2: Creating and Utilizing Class Project	3
Test #2	1
Total	15

Unit 3 – Class Hierarchy

Topic	**Time Frame**
Introduction to super and sub classes	1
Exploration of super/sub inheritance attributes	1
Introduction to abstract classes and interfaces	2
Lesson on comparable interface	1
Exploration of memory allocation in super and sub classes	1
Exploration of dynamic vs. static binding	
Project #3: Creating and Utilizing Class Hierarchy	4
Test #3	1
Total	11

QUARTER 2

Unit 4 – Review of Common Java Packages

Topic	**Time Frame**
Review of how packages are imported and why this action is required	1
Review of basic String commands and lessons on new commands (toString, substring)	2
Review of Math commands	1
Lesson on Object class	2
Project #4: String/Math Project	3
Test Review	1
Test #1	1
Total	11

Unit 5 – 1 and 2-dimensional Arrays

Topic	**Time Frame**
Review of 1 dimension arrays and introduction to 2-dimensional arrays	1

Lesson on declaring arrays, assigning values and possible errors (array out of bounds)	1
Exploration of standard array algorithms	2
Project #5: Array Project	3
Test #2	1
Total	8

Unit 6 – ArrayList Class

Topic	**Time Frame**
Lesson on the restrictions of basic arrays and an introduction to the ArrayList Class and the Wrapper classes	1
Lesson on the necessity of Wrapper classes and basic creation and utilization of the ArrayList class.	2
Quiz: ArrayList vs. array	1
Lesson on effectively using ArrayList class (swapping values, combining arrays, etc.)	1
Lesson on using ArrayList with Objects	1
Project #6: ArrayList Object Project	3
Total	9

Unit 7 – Recursion

Topic	**Time Frame**
Introduction to recursion and its purpose	1
Lesson on creating a recursive program	1
Project #7: Recursion Project	3
Lesson on recursive techniques and common mistakes	2
Quiz: Recursion	1
Test Review	1
Test #7	1
Midterm Review	3
Midterm	1
Total	14

QUARTER 3

Unit 8 – Searching and Sorting

Topic	**Time Frame**
Lesson on linear vs. binary searches	1
Programming linear and binary searches	1
Lesson on the importance of sorting and a discussion on different sorting techniques.	1
Lesson on the selection sort (first by hand and then programming)	1

Lesson on the insertion sort (first by hand and then programming)	1
Lesson on merge sort (first by hand and then programming)	1
Quiz: Performing different sorts and identifying best/worst case scenarios.	1
Project #8: Selection/Insertion Sort Project	4
Total	11

Unit 9 – Magpie Lab

Topic	**Time Frame**
Discussion on the new AP standards and introduction to the labs	1
Magpie Activity 1 (Introduction to chat bots)	2
Magpie Activity 2 (Modification of chat bot code)	2
Magpie Activity 3 (Custom chat bot responses)	2
Magpie Activity 4 (Utilizing Arraylists and chat bots)	2
Total	9

Unit 10 – pix Lab

Topic	**Time Frame**
Introduction to pix lab	1
Pix lab Activity 1 (2-dimensional arrays and digital pictures)	2
Pix lab Activity 2 (modifying digital pictures)	3
Pix lab Activity 3 (creating/altering digital pictures)	3
Total	9

Unit 11 – Elevens Lab

Topic	**Time Frame**
Introduction to GUI and Elevens Lab	1
Elevens Lab Activity 1 (creating elevens game)	2
Elevens Lab Activity 2 (integrating GUI environment)	3
Elevens Lab Activity 3 (extending game design)	3
Elevens Lab Activity 4 (developing game design)	3
Total	12

QUARTER 4

Unit 12 – AP Exam Review and Practice

Topic	**Time Frame**
Discussion and outline of AP exam and expectations	1
Review of Quarter 1 and 2	1
Practice AP Questions on Quarter 1 and 2 Topics	1
Review of Quarter 3 and 4	1

Practice AP Questions on Quarter 3 and 4 Topics	1
Lesson on expectations for free response section of AP exam	1
Practice of Free Response AP Questions	3
Review of Gridworld Code and Expectations	1
Practice of Multiple-Choice and Free Response Gridworld Questions	3
PRACTICE AP EXAM	3
Review of difficult topics/student sticking points	3
Total	19

Unit 0: Review of Object Oriented Programming Design

Standard	
Big Ideas: *Course Objectives / Content Statement(s)* ● **Students will review the core features of OOP** ● **Students will review basic java programming techniques and terminology.**	
Essential Questions *What provocative questions will foster inquiry, understanding, and transfer of learning?*	**Enduring Understandings** *What will students understand about the big ideas?*
● What is the proper syntax for creating a class? ● What is the difference between compile-time, run-time and logic errors? ● What are the names of the different parts of a java program (constructor, method, object, data, etc.)?	Students will understand that… ● Classes and data types must be custom designed for every unique task. ● Java is a language that has its own unique syntax. ● Code must be as simple, logical and intuitive as possible.
Areas of Focus: Proficiencies (Cumulative Progress Indicators)	**Examples, Outcomes, Assessments**
Students will: Be able to create a class. Name the key components of a Java program. Identify the common types of errors made.	Instructional Focus (1 week): ● No assessment Instructional Strategies: ● Individual work ● In-class programming time with assistance from teacher ● Worksheets and mini-5 minute assessments. Technology Integration ● Computer/BlueJ program Global Perspectives

	We will discuss the AP Computer Science A examination and investigate its effect on computer science instruction at the high school level. We will briefly discuss the evolution of computer science as it is taught in high schools.

Unit 1: Utilizing Primitive Data Types and Conditional Statements

Standard	
Big Ideas: *Course Objectives / Content Statement(s)* ● **Students will review the core primitive data types – int, double and boolean.** ● **Students will know how different data types interact with each other** ● **Students will know what values different data types may hold** ● **Students will know how to write compound conditional statements and be able to evaluate truth tables** ● **Students will review the core control structures (while, if and for).**	
Essential Questions *What provocative questions will foster inquiry, understanding, and transfer of learning?*	**Enduring Understandings** *What will students understand about the big ideas?*
● What is the difference between / and %? ● What are the core arithmetic operators in Java? ● How do I write logic tables in order to assist in writing code?	Students will understand that… ● All evaluations done by conditional statements must be true or false ● Conditional statements will only check as much information as required. ● Data types may interact with each other in certain circumstances, however precautions must be made to ensure that no data is lost.
Areas of Focus: Proficiencies (Cumulative Progress Indicators)	**Examples, Outcomes, Assessments**
Students will:	Instructional Focus (2 weeks):
Be familiar with the int, double and boolean types. They will be aware that other types	● Hands-on programming time mixed with instructional classes on new

exist, but we will not investigate them with any significant depth. All the core arithmetic operators (+, -, *, /, %) The increment and decrement operators (++, --) The assignment operator and relational operators (=, ==, !=, >, >=, <, <=) Logical operations (&&, ||, !) Control Structures (if, if/else, while, for) The "final" keyword.	topics or common problems/mistakes Sample Assessments: ● 5-minute quizzes to test concepts previously learned in class. ● Project: Students will create a program that allows the user to create and keep track of their bank accounts. Functions might include creating and closing the account, adding and taking out money, compounding interest, etc. ● Written Test: Students will answer AP style multiple-choice and free response questions on primitive data types and conditional statements. Instructional Strategies: ● Individual work with help from peers ● In-class programming time with assistance from teacher ● At home programming time Technology Integration ● Computer/BlueJ program Media Literacy Integration Students will investigate the usage of truth tables in Classes. We will investigate real world possible uses for truth tables and write them out by hand to examine different cases where sensitive information is handled. For example, we may examine when an ATM machine determines that the person using the current card is not the owner.

Unit 2: Review of Common Java Classes

<table>
<tr><th colspan="2">Standard</th></tr>
<tr><td colspan="2">Big Ideas: Course Objectives / Content Statement(s)
● Students will review common java packages including java.lang.Object, java.lang.Math, java.lang.String
● Students will review how to import java packages.
● Students will review proper notation and utilization of these packages' most common classes (Math, Random, PrintStream, String)</td></tr>
<tr><td>Essential Questions
What provocative questions will foster inquiry, understanding, and transfer of learning?</td><td>Enduring Understandings
What will students understand about the big ideas?</td></tr>
<tr><td>● How do I utilize java packages that have already been constructed?
● What is the proper syntax for each class?
● How does the String class function and which aspects of it is part of the core functionality of Java?
● How do I gain user input from the console?</td><td>Students will understand that…
● Each class has its own unique uses and syntax and that it is critical to know the difference between important classes.
● Classes should only be imported when required for core use in a program.
● Classes are externally written components to Java that have such widespread usefulness that they have become fully integrated into the language.</td></tr>
<tr><td>Areas of Focus: Proficiencies (Cumulative Progress Indicators)</td><td>Examples, Outcomes, Assessments</td></tr>
</table>

Students will:	Instructional Focus (3 weeks): ● Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes Sample Assessments: ● 5-minute quizzes to test concepts previously learned in class. ● Project: Students will create a class that utilizes user input to develop some core functionality. For example, students may create a class that allows the user to purchase different items from a store. ● Written Test: Students will answer AP style multiple-choice and free response questions on common Java classes. Instructional Strategies: ● Individual work with help from peers ● In-class programming time with assistance from teacher ● At home programming time Technology Integration ● Computer/BlueJ program Global Perspectives We will examine the development of Java as a programming language and discuss why certain core functionalities of Java are used through classes. This will require a discussion and investigation of the advancement of programming over time.
Know String concatenation utilizes toString	
Be able to utilize all the escape sequences inside print statements	
Be able to utilize all the appropriate String methods (substring, toString, etc.)	
How to write code that will use user input in their program as int, double and String.	

Unit 3: Class Hierarchy

<table>
<tr><th colspan="2">Standard</th></tr>
<tr><td colspan="2">Big Ideas: Course Objectives / Content Statement(s)
Students will know how to implement super/sub classes
Students will be able to utilize inheritance in programming and create webbed class hierarchy
Students will be able to utilize static and dynamic binding
Students will know how to reference null parameters
Students will be able to create abstract classes and interfaces along with utilizing the comparable interface</td></tr>
<tr><th>Essential Questions
What provocative questions will foster inquiry, understanding, and transfer of learning?</th><th>Enduring Understandings
What will students understand about the big ideas?</th></tr>
<tr><td>When and why do we use super and sub classes?
How does the computer decide what information to reference and where to look for that information?
How do abstract classes and interfaces relate to super/sub classes?</td><td>Students will understand that…
Super and sub classes must follow the is-a relationship.
That abstract classes and interfaces have fundamental differences in structure and design.
Static and dynamic binding affects how a program interprets code.</td></tr>
</table>

<table>
<tr><th>Areas of Focus: Proficiencies (Cumulative Progress Indicators)</th><th>Examples, Outcomes, Assessments</th></tr>
<tr><td>Students will:</td><td rowspan="4">Instructional Focus (2 weeks):
Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes
Class work in order to craft a program as a group
In-Class single day programming assignments
Sample Assessments:
5-minute quizzes to test concepts previously learned in class.
Project: Students will create a program that organizes different types of athletes based on which sport they play.
Written Test: Students will answer AP style multiple-choice and free response questions on class creation and utilization.
Instructional Strategies:
Group work/individual work
In-class programming time with assistance from teacher
At home programming time
Technology Integration
Computer/BlueJ program
Global Perspectives

We will examine how Java extends class functionality to incorporate UML structures and class hierarchy. An exploration of how this enhances our programming capabilities will occur.</td></tr>
<tr><td>Know how to create super/sub classes from scratch</td></tr>
<tr><td>Know how to utilize the comparable interface</td></tr>
<tr><td>Know how to create interfaces/abstract classes from scratch.</td></tr>
<tr><td>Test code utilize static and dynamic binding.</td></tr>
</table>

Unit 4: Review of Classes

<table>
<tr><th colspan="2">Standard</th></tr>
<tr><td colspan="2">Big Ideas: Course Objectives / Content Statement(s)
● Students will know class terminology (constructor, instance variable, etc)
● Students will review how to create the basic components of a Class including constructors, methods, functions, instance variables, “get” functions, static and final information
● Students will learn advanced class techniques including – overloading methods, “this” call and general collection classes.
● Students will examine common runtime errors associated with the topics learned in previous 2 units (NullPointException, ArrayIndexOutOfBoundException, etc).</td></tr>
<tr><th>Essential Questions
What provocative questions will foster inquiry, understanding, and transfer of learning?</th><th>Enduring Understandings
What will students understand about the big ideas?</th></tr>
<tr><td>● Which features of Class design inherently go together (constructors</td><td>Students will understand that…
● Classes must be designed around related core functionality.</td></tr>
</table>

<table>
<tr><td>and this, “get” functions and instance variables, etc.)?
● What is the proper time to use each call?
● What types of coding mistakes create run-time errors?</td><td>● Classes should be designed to minimize or eliminate any possible errors from those that use it.
● Class design is an art. There is no single best way to implement any given system.</td></tr>
<tr><td>Areas of Focus: Proficiencies (Cumulative Progress Indicators)</td><td>Examples, Outcomes, Assessments</td></tr>
<tr><td>Students will:</td><td rowspan="6">Instructional Focus (3 weeks):
● Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes
● Class work in order to craft a program as a group
● In-Class single day programming assignments

Sample Assessments:
● 5-minute quizzes to test concepts previously learned in class.
● Project: Students will create a program that draws different polygons using multiple constructors. They will then write code that finds various information about the shapes (their area, perimeters and so on).
● Written Test: Students will answer AP style multiple-choice and free response questions on class creation and utilization.

Instructional Strategies:
● Group work/individual work
● In-class programming time with assistance from teacher
● At home programming time

Technology Integration
● Computer/BlueJ program

Global Perspectives

We will examine how Java as a language changes over time through the utilization and integration of previously designed classes. In this regard Java is much like a</td></tr>
<tr><td>Know how to utilize method overloading</td></tr>
<tr><td>Construct classes from scratch and use the “this” statement to make multiple constructors.</td></tr>
<tr><td>What visibility means in programming (i.e.: The appropriate usage of private and public)</td></tr>
<tr><td>What “static” means and how it is important to programming.</td></tr>
<tr><td>Know how to use “final” by itself and along with “public/private” and “static”</td></tr>
<tr><td>Know how to write their program to utilize a previously written core Java class from the early unit (Random, String).</td></tr>
<tr><td>Be able to name and identify the main run-time errors that cause a Java program to crash.</td></tr>
</table>

	spoken language – a living tapestry that changes as new ideas are developed. Java is not a language owned by any one group as users all over the world develop its functionality. We will examine the origins of core Java classes to illustrate this point.

Unit 5: 1 and 2-dimensional arrays

Standard	
Big Ideas: *Course Objectives / Content Statement(s)* ● **Students will review the core components of primitive type 1-dimensional arrays.** ● **Students will know how to use 1-dimensional arrays made of Objects.** ● **Students will extend this knowledge to solving problems utilizing 2-dimensional arrays with both primitive data types and Objects.**	
Essential Questions *What provocative questions will foster inquiry, understanding, and transfer of learning?*	**Enduring Understandings** *What will students understand about the big ideas?*
● How do arrays hold information differently than other data types? ● When should I utilize an array of objects as opposed to an array of data? ● How do 2-dimensional arrays function differently than 1-dimensional arrays?	Students will understand that… ● Arrays are indexed from 0 to length-1. ● Arrays of Objects create a chain of stored information similar to a user database and are more efficient than utilizing multiple arrays.

Areas of Focus: Proficiencies (Cumulative Progress Indicators)	**Examples, Outcomes, Assessments**
Students will:	Instructional Focus (1 week): ● Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes Sample Assessments: ● 5-minute quizzes to test concepts previously learned in class. ● Project: Students will create a program that stores a student's school attendance and uses this array to create a class that can find various information (how many absences, how many tardies, what days of the week the student is absent, possible punishments, etc.) ● Written Test: Students will answer AP style multiple-choice and free response questions on arrays and array implementation. Instructional Strategies: ● Individual work with help from peers ● In-class programming time with assistance from teacher ● At home programming time Technology Integration ● Computer/BlueJ program Media Literacy Integration We will examine real-world situations where Object arrays are required. Each group will select a different utilization of Object arrays and report to the class on what data is stored inside the array. For example, students may select and research doctor's medical records or their video game "gamer tags".
Know how to create and use arrays with primitive data types and Objects	
Know when and why it is appropriate to use Object arrays.	
Know how to index and find information in an array through traversing.	
Know how to use core functions of an array including addition and deletion.	
Know the shortcomings of arrays.	
Know how to create and/or utilize arrays that have been initialized and that haven't.	

Unit 6: ArrayList Class

Standard	
Big Ideas: *Course Objectives / Content Statement(s)* ● **Students will be able to appropriately utilize the List and ArrayList class** ● **Students will know how to utilize the Integer and Double "Wrapper" classes**	
Essential Questions *What provocative questions will foster inquiry, understanding, and transfer of learning?*	**Enduring Understandings** *What will students understand about the big ideas?*
● Why do we need wrapper classes if they appear to serve the same functionality as the core primitive data types? ● How is the ArrayList class more efficient than the primitive array class?	Students will understand that… ● The ArrayList Class was created to provide additional functionality to the array data type. ● The Wrapper class is required for use of the ArrayList class.

Areas of Focus: Proficiencies (Cumulative Progress Indicators)	Examples, Outcomes, Assessments
Students will: Know why the ArrayList class is required to fix a problem with the primitive data type array. Know why we must use the Wrapper classes to address a limitation of Java and why this limitation was created in the first place. Know the core methods in the Wrapper, List and ArrayList classes 	Instructional Focus (2 weeks): ● Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes ● Class work in order to craft a program as a group ● In-Class single day programming assignments Sample Assessments: ● 5-minute written warm up quizzes ● Programming Project ● Project: Students will modify an old project to integrate the ArrayList class. For example, students can modify their ATM program so that it may utilize the more advanced functionality provided by the class. ● Written Test: Students will answer AP style multiple-choice and free response questions on the ArrayList class. Instructional Strategies: ● Group work/individual work ● In-class programming time with assistance from teacher ● At home programming time Technology Integration ● Computer/BlueJ program Media Literacy Integration We will examine the first build of Java. Students will research what functionality was included in the language from the very start and what functionality was added over time as programming became more advanced. Each group will select an aspect of the program from the initial build (i.e.: arrays) and see how they have been modified over time to provide enhanced functionality (i.e.: the ArrayList class).

Unit 7: Recursion

Standard	
Big Ideas: *Course Objectives / Content Statement(s)* ● **Students will know how to use recursion to solve a variety of problems.** ● **Students will examine common run-time and logic errors common in recursion.**	
Essential Questions *What provocative questions will foster inquiry, understanding, and transfer of learning?*	**Enduring Understandings** *What will students understand about the big ideas?*
● How is recursion more efficient and elegant in certain cases?	Students will understand that…

● How can I rewrite old code to utilize recursion?	● Recursion is a key skill used when programming that allows for enhanced functionality and elegance.
Areas of Focus: Proficiencies (Cumulative Progress Indicators)	**Examples, Outcomes, Assessments**
Students will: Know how to utilize recursion to solve simple tasks and how to rewrite code to integrate recursion.	Instructional Focus (3 weeks): ● Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes Sample Assessments: ● 5-minute quizzes to test concepts previously learned in class. ● Project: Students will write several small methods and that utilize recursion. One method will require students to write solutions to summation problems. Instructional Strategies: ● Individual work ● In-class programming time with assistance from teacher ● At home programming time Technology Integration ● Computer/BlueJ program

Unit 8: Searching and Sorting

Standard
Big Ideas: *Course Objectives / Content Statement(s)* ● **Students will know how to write and utilize the bubble, selection and insertion sorts.** ● **Students will be able to sort data by hand using any of the above sorts.** ● **Students will know the advantages and disadvantages of each type of sorting method.** ● **Students will know how sequential and binary searches function. They will be able to work these searches by hand and utilize them while programming.**

<table>
<tr><th>Essential Questions
What provocative questions will foster inquiry, understanding, and transfer of learning?</th><th>Enduring Understandings
What will students understand about the big ideas?</th></tr>
<tr><td>● How do bubble, selection and insertion sorts function?
● How do I decide which sort will be the most efficient?
● How does sorting relate to arrays?
● Why is sorting an important concept in computer science?
● How do each of the searches function and which are most efficient?</td><td>Students will understand that…
● Sorting data through predetermined methods is a crucial programming skill.
● Students should be comfortable sorting data by hand, coding it and looking at results used identifying which sorting method.</td></tr>
<tr><th>Areas of Focus: Proficiencies (Cumulative Progress Indicators)</th><th>Examples, Outcomes, Assessments</th></tr>
<tr><td>Students will:
Know how each of the three sorts work and how to sort lists by hand.
Know how to program these sorts.
Know how the different searches functions as well as the advantages and disadvantages of each one.</td><td rowspan="2">Instructional Focus (3 weeks):
● Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes
● Group work/class work in order to craft a program as a group
● In-Class single day programming assignments

Sample Assessments:
● 5-minute quizzes to test concepts previously learned in class.
● Written Test: Students will answer AP style multiple-choice and free response questions on sorting. They will have to sort data by hand and identify which sort was used given a set of data.

Instructional Strategies:
● Group work/individual work
● In-class programming time with assistance from teacher
● At home programming time

Technology Integration
● Computer/BlueJ program

Global Perspectives</td></tr>
</table>

	We will discuss and examine the importance in sorting in computer programming. A focus will be placed on real-world situations where large amounts of data must be organized and the importance of having different methods to sort data. We will conclude this section with students attempting to solve the famous Google sorting question and an examination of its final solution.

Unit 9: Magpie Lab

Standard
Big Ideas: *Course Objectives / Content Statement(s)* ● **Students will examine prewritten lab code and be able to interpret and add to this code** ● **Students will develop techniques to improve chatbots.**

<table>
<tr><td colspan="2">

- **Students will enhance their ability to work with the String class.**

</td></tr>
<tr><td>

Essential Questions
What provocative questions will foster inquiry, understanding, and transfer of learning?

</td><td>

Enduring Understandings
What will students understand about the big ideas?

</td></tr>
<tr><td>

- How do you expand on code that has been written by someone else?
- How do you adhere to standards to ease programming?

</td><td>

Students will understand that…

- Labs require examination, interpretation and expansion of core ideas.

</td></tr>
<tr><td>

Areas of Focus: Proficiencies (Cumulative Progress Indicators)

</td><td>

Examples, Outcomes, Assessments

</td></tr>
<tr><td>

Students will:

Utilize String methods.

Work with prewritten lab code.

</td><td>

Instructional Focus (2 weeks):

- Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes

Instructional Strategies:

- Individual work
- In-class programming time with assistance from teacher
- At home programming time

Technology Integration

- Computer/BlueJ program

</td></tr>
</table>

Unit 10: pix Lab

Standard
Big Ideas: *Course Objectives / Content Statement(s)*

<table>
<tr><td colspan="2">

- **Students will examine prewritten lab code and be able to interpret and add to this code**
- **Students will enhance their work with 2D arrays.**
- **Students will be able to modify and edit digital images to create manipulated photos and collages.**

</td></tr>
<tr><td>Essential Questions
What provocative questions will foster inquiry, understanding, and transfer of learning?</td><td>Enduring Understandings
What will students understand about the big ideas?</td></tr>
<tr><td>

- How is a digital picture created?
- How do we access and modify information in a 2D array and how is this related the digital picture we wish to manipulate?
- How do we standardize complex operations?

</td><td>Students will understand that…

- It is possible to work on, modify and add to projects even if they are not familiar with how parts of the program function.
- How to utilize classes in a GUI.

</td></tr>
<tr><td>Areas of Focus: Proficiencies (Cumulative Progress Indicators)</td><td>Examples, Outcomes, Assessments</td></tr>
<tr><td>Students will:
Utilize 2D array methods.
Modify and edit digital pictures.</td><td>Instructional Focus (2 weeks):

- Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes

Instructional Strategies:

- Individual work
- In-class programming time with assistance from teacher
- At home programming time

Technology Integration

- Computer/BlueJ program

</td></tr>
</table>

Unit 11: Elevens Lab

<table>
<tr><th colspan="2">Standard</th></tr>
<tr><td colspan="2">Big Ideas: Course Objectives / Content Statement(s)
● Students will examine prewritten lab code and be able to interpret and add to this code
● Students will integrate with a GUI to create their first Java game.
● Students will extend and develop their own games based on the provided framework.</td></tr>
<tr><th>Essential Questions
What provocative questions will foster inquiry, understanding, and transfer of learning?</th><th>Enduring Understandings
What will students understand about the big ideas?</th></tr>
<tr><td>● How is a Java package created?
● How is a GUI created?
● How do you modify a GUI?</td><td>Students will understand that…
● The core programming skills they have developed are utilized as all levels of programming.
● How to identify key pieces of code to modify and replace.</td></tr>
<tr><th>Areas of Focus: Proficiencies (Cumulative Progress Indicators)</th><th>Examples, Outcomes, Assessments</th></tr>
<tr><td>Students will:
Utilize Objects and ArrayLists in a GUI</td><td rowspan="2">Instructional Focus (3 weeks):
● Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes

Instructional Strategies:
● Individual work
● In-class programming time with assistance from teacher
● At home programming time

Technology Integration
● Computer/BlueJ program</td></tr>
<tr><td></td></tr>
</table>

Unit 12: AP Exam Review and Practice

Standard	
Big Ideas: *Course Objectives / Content Statement(s)* ● **Students will be prepared to take the AP Computer Science Exam A**	
Essential Questions *What provocative questions will foster inquiry, understanding, and transfer of learning?*	**Enduring Understandings** *What will students understand about the big ideas?*
● What is left for me to learn or review to prepare for the exam?	Students will understand that… ●
Areas of Focus: Proficiencies (Cumulative Progress Indicators)	**Examples, Outcomes, Assessments**
Students will:	Instructional Focus (~4 weeks): ● Hands-on programming time mixed with instructional classes on new topics or common problems/mistakes Sample Assessments: ● 5-minute quizzes to test concepts previously learned in class. ● Written Test: Students will be given an old, full AP exam for practice. Instructional Strategies: ● Individual work ● In-class programming time with assistance from teacher ● At home programming time Technology Integration ● Computer/BlueJ program

Curricular Addendum

Career-Ready Practices **CRP1**: Act as a responsible and contributing citizen and employee. **CRP2**: Apply appropriate academic and technical skills.	**Interdisciplinary Connections** ● Close Reading of works of art, music lyrics, videos, and advertisements

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.

- Use Standards for Mathematical Practice and Cross-Cutting Concepts in science to support debate/inquiry across thinking processes

Technology Integration

Ongoing:

- Listen to books on CDs, Playaways, videos, or podcasts if available.
- Use document camera or overhead projector for shared reading of texts.

Other:

- Use Microsoft Word, Inspiration, or SmartBoard Notebook software to write the words from their word sorts.
- Use available technology to create concept maps of unit learning.

Instructional Strategies:
Supports for English Language Learners:

Sensory Supports	Graphic Supports	Interactive Supports
Real-life objects (realia)	Charts	In pairs or partners
Manipulatives	Graphic organizers	In triads or small groups
Pictures & photographs	Tables	In a whole group
Illustrations, diagrams, & drawings	Graphs	Using cooperative group
Magazines & newspapers	Timelines	structures
Physical activities	Number lines	With the Internet (websites) or software programs
Videos & films		In the home language
Broadcasts		With mentors
Models & figures		

from https://wida.wisc.edu

Media Literacy Integration

- Use multiple forms of print media (including books, illustrations/photographs/artwork, video clips, commercials, podcasts, audiobooks, Playaways, newspapers, magazines) to practice reading and comprehension skills.

Global Perspectives

- The Global Learning Resource Library

Differentiation Strategies:

Accommodations	Interventions	Modifications
Allow for verbal responses	Multi-sensory techniques	Modified tasks/ expectations
Repeat/confirm directions	Increase task structure (e.g., directions, checks for understanding, feedback)	Differentiated materials
Permit response provided via computer or electronic device	Increase opportunities to engage in active academic responding (e.g., writing, reading aloud, answering questions in class)	Individualized assessment tools based on student need
Audio Books	Utilize prereading strategies and activities: previews, anticipatory guides, and semantic mapping	Modified assessment grading
