# AP Computer Science A

## U-46 Curriculum Scope and Sequence

| Reporting Strand | Instructional Focus | Standards | Pacing |
|---|---|---|---|
| **Intro to Computer Science & Java Programming** | Correctly use java syntax, style and comments | IIB3, IIB4b, IIB5c, IIIA-C, IVB | 2-3 weeks |
| | Know the vocabulary and software development issues related to Java programming | | |
| **Primitive Data Types and Variables** | Create and use primitive data types or String objects in a program. | IIB2a/b, IIB5a, IIIA-C, IIIF2, IIIF1, IVA/B | 2-3 weeks |
| | Correctly use java math operators, cast operators and the Math class library to read and write math algorithms | | |
| | Number Base Conversions | | |
| **Intro to Object Oriented Programming** | Understand how to create classes and objects | IIA1-3, IIA5, IIB2c/d, IIB4a, IIIA-C, IVC | 2-3 weeks |
| | Understand the major parts of a class: instance variables (fields), methods, and constructors | | |
| | Be able to define and invoke methods with or without parameters and with or without return types. | | |
| **Boolean Expressions and Conditional Statements** | Write and evaluate boolean expressions using logical and relational operators | IIB5c, IIB4c, IIIA-D, IVA | 3 weeks |
| | Determine output of code that utilizes conditional statements | | |
| | Write code that utilizes conditional statements | | |
| **Iterations and Arrays** | Determine the correct output of code that uses iterations | IIB4d, IIIA-D, IVE, VA1 | 4-5 weeks |
| | Write code that utilizes while, for and do-while iterations | | |
| | Create and traverse arrays with iterations and use arrays as arguments to methods | | |
| **Strings** | Understand the various unique features of Strings as objects. | IIB5b, IIIA-D, IVB | 3 weeks |
| | Understand how methods from the String, Integer, and Double classes can be used to manipulate strings | | |
| | Be able to write Java code which utilizes methods from the String, Integer, and Double classes | | |
| **Classes, Class Hierarchies, and Interfaces** | Understand the concepts of passing by reference vs. passing by value | IA1-6, IIA1-5, IIB1/2e, IIIA-D, IVC | 3-4 weeks |
| | Static fields and methods | | |
| | Understand encapsulation (private vs. public) | | |
| | Understand class inheritance and polymorphism | | |
| | Understand abstract classes and interfaces. | | |
| **Array Lists** | Understand how the java.util.ArrayList methods and constructors are used | IIIA-E, IVD, VA | 2-3 weeks |
| | Be able to write code using java.util.ArrayList methods and constructors | | |
| **Recursion** | Analyze the output from recursive code | IIB4e, IIIA-E | 2-3 weeks |
| **Searching & Sorting** | Understand the differences and running time of various searches and sorts | IIIA-E, VB/C VA2-3 | 2-3 weeks |
| **Computing in Context** | Understand system reliability, privacy, legal issues and intellectual property, social and ethical ramifications of computer use, | VI | 1 week |

# Every Unit Rubric: Program Analysis

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIIA-C, IA1 | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | Identify and correct logical errors in code.<br><br>Test code for all possible inputs. | Correct compilation and runtime exception errors in code | Interpret compilation and runtime exception errors in code | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 1: Introduction to Computer Science & Java Programming

| Standards | Meets | Proficiency | Improvement Needed |
|---|---|---|---|
| IIB3, IIB4b, IIB5c, IIIA-C, IIIF1, IVB | | ● Java Background<br>○ History of programming<br>○ Advantages of Java over other languages<br>● Java SDK<br>○ Downloading Java<br>● IDE<br>○ What is an IDE/What are its functions<br>○ Downloading/using an IDE<br>● Compiling Java<br>○ What happens when Java compiles<br>○ bytecode<br>○ class files<br>● Syntax vs Style<br>○ Why style is important<br>○ What is a syntax error<br>● Naming Classes and Variables<br>● Indenting<br>○ Properly indenting programs<br>○ Importance of indenting properly<br>● Comments<br>○ Why comments are used<br>○ Creating single and multi-line comments<br>○ Usefulness of Javadoc comments<br>● Use the System class for input/output | |

# Reporting Strand 2: Primitive Data Types and Variables

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIB2a/b, IB5a, IIIA-C, IIF2, IVA/B | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | Declare and initialize primitive data types using **multiple** syntaxes<br><br>**Write** and determine values that uses String concatenation (with + operator) and escape sequences.<br><br>**Write** and evaluate expressions (utilizing the order of operations in Java) that use %, type casting, and methods from the Math class<br><br>**Given a decimal, convert** to binary, octal and hexadecimal | **Declare and initialize** primitive data types using one type of syntax<br><br>**Determine values** that uses String concatenation (using + operator) and escape sequences.<br><br>**Evaluate expressions** (utilizing the order of operations in Java) **that use %, type casting, and methods from the Math class**<br><br>**Given a number in binary, octal and hexadecima**l convert to a decimal | **Identify** a correct syntax which declares and initializes primitive data types (int, double, and boolean)<br><br>**Identify** correct syntax which uses String concatenation (with + operator) and escape sequences.<br><br>**Identify methods** in the Math class (including: Math.pow(), Math.sqrt(), and Math.random())<br><br>**Count** in binary, octal and hexadecimal | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 3: Introduction to Object-Oriented Programming

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIA3,5, IIB2c/d, IIB4a, IIIA-C, IVC | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | Create and utilize **all** of the following:<br>● objects<br>● fields and local variables<br>● constructor<br><br>Explain the relationship between:<br>● methods and objects<br>● fields and class attributes<br>● "new" and the constructor<br>● variables and their scope<br><br>Create and utilize **all** of the following:<br>● methods with a return type<br>● methods without a return type<br>● methods with parameters<br>● methods without parameters | Create and utilize **2** of the following:<br>● objects<br>● fields and local variables<br>● constructor<br><br>Explain the relationship between **3 of the following:**<br>● methods and objects<br>● fields and class attributes<br>● "new" and the constructor<br>● variables and their scope<br><br>**Create and utilize at least 2** of the following:<br>● methods with a return type<br>● methods without a return type<br>● methods with parameters<br>● methods without parameters | Identify<br>● objects<br>● fields and local variables<br>● constructor<br><br>Explain the relationship between **2 of the following:**<br>● methods and objects<br>● fields and class attributes<br>● "new" and the constructor<br>● variables and their scope<br><br>**Identify** correct syntax that utilizes:<br>● methods with a return type<br>● methods without a return type<br>● methods with parameters<br>● methods without parameters | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 4: Boolean Expressions and Conditional Statements

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIB5c, IIB4c, IIIA-D, IVA | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | **Write** and evaluate boolean expressions (understanding the order of operations and including short-circuit evaluation and De Morgan's law) using logical and relational operators<br><br>Write and determine values of code utilizing the following conditional statements:<br>● if<br>● if/else<br>● **extended if**<br>● **nested if** | **Evaluate** boolean expressions (understanding the order of operations and including short-circuit evaluation and De Morgan's law) using logical **and** relational operators<br><br>**Write and** determine values of code utilizing the following conditional statements:<br>● if<br>● if/else | **Evaluate** boolean expressions (understanding the order of operations and including short-circuit evaluation and De Morgan's law) using logical **or** relational operators<br><br>Determine values of code utilizing the following conditional statements:<br>● if<br>● if/else | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 5: Iterations and Arrays

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIB4d, IIIA-D, IVE, VA1 | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br><br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | **Write** and determine values of code utilizing the following iterations:<br><br>● for<br>● while<br>● do-while<br><br>Create, modify, and traverse 1D arrays with iterations (including for-each) and use 1D arrays as parameters and return types.<br><br>Create, modify, and traverse 2D arrays with nested iterations (including for-each) and use 2D arrays as parameters and return types. | Determine values of code utilizing **all of** the following iterations:<br><br>● for<br>● while<br>● do-while<br><br>Create, modify, and traverse 1D arrays with iterations (including for-each)<br><br>Create, modify, and traverse 2D arrays with nested iterations (including for-each) | Determine values of code utilizing **1 of** the following iterations:<br><br>● for<br>● while<br>● do-while<br><br>Declare and initialize 1D arrays<br><br>Declare and initialize 2D arrays | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 6: Strings

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIB5b, IIIA-D, IVB | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | **Write and** determine the value of methods from String, **Integer, and Double classes**<br><br>Apply the implications of **both of the** unique features of Strings as objects:<br>● Strings are immutable<br>● Strings can be declared with or without using keyword new. | Determine the value of **all** of the following String methods:<br>● length()<br>● substring(int from, int to)<br>● substring (int from)<br>● indexOf(String str)<br>● compareTo(String other)<br><br>Apply the implication of **1 of** the unique features of Strings as objects:<br>● Strings are immutable<br>● Strings can be declared with or without using keyword new. | Determine the value of **at least 2** of the following String methods:<br>● length()<br>● substring(int from, int to)<br>● substring (int from)<br>● indexOf(String str)<br>● compareTo(String other)<br><br>Identify the implication of the unique features of Strings as objects:<br>● Strings are immutable<br>● Strings can be declared with or without using keyword new. | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 7: Classes, Class Hierarchies, and Interfaces

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IA2-6, IIA1-5, IIB1/2e, IIIA-D, IVC | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | Apply the following concepts **when writing code:**<br>● static and non-static fields and methods<br>● encapsulation<br>● overloaded methods and constructors<br>● passing parameters by reference vs. by value<br>● class inheritance and polymorphism<br>● abstract classes and interfaces. | **Given code, apply** the following concepts **to analyze various programming situations:**<br>● static and non-static fields and methods<br>● public vs.private (encapsulation)<br>● overloaded methods and constructors<br>● passing parameters by reference vs. by value<br>● class inheritance and polymorphism<br>● abstract classes and interfaces. | **Identify** the following:<br>● differences between static and non-static fields and methods<br>● differences between public vs. private<br>● attributes of overloaded methods and constructors<br>● differences between passing parameters by reference vs. by value<br>● attributes of class inheritance and concept of polymorphism<br>● attributes of abstract classes and interfaces. | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 8: Array Lists

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIIA-E, IVD, VA | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | **Write code that utilizes the following:**<br>• size()<br>• add(E obj)<br>• add(int index, E obj)<br>• get (int index)<br>• set (int index, E obj)<br>• remove(int index)<br>• ArrayList()<br>• ArrayList(int initialCapacity) | Determine the results (including the implications of capacity/size) when utilizing **all** of the following methods:<br>• size()<br>• add(E obj)<br>• add(int index, E obj)<br>• get (int index)<br>• set (int index, E obj)<br>• remove(int index)<br>• ArrayList()<br>• ArrayList(int initialCapacity) | Determine the results (including the implications of capacity/size) when utilizing **at least 5** of the following:<br>• size()<br>• add(E obj)<br>• add(int index, E obj)<br>• get (int index)<br>• set (int index, E obj)<br>• remove(int index)<br>• ArrayList()<br>• ArrayList(int initialCapacity) | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 9: Recursion

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIB4e, IIIA-E | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | Analyze the output from recursive code that includes multiple/mutual **and** tail/non-tail recursion | Analyze the output from recursive code that includes multiple/mutual **or** tail/non-tail recursion | Analyze the output from recursive code that includes single recursion | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

# Reporting Strand 10: Searching and Sorting

| Standards | 4 – Mastery | 3 – Proficient | 2 - Basic | 1 – Below Basic | 0 – No Evidence |
|---|---|---|---|---|---|
| IIIA-E, VB/C, VA23 | Can extend thinking beyond the standard, including tasks that may involve one of the following:<br>• Designing<br>• Connecting<br>• Synthesizing<br>• Applying<br>• Justifying<br>• Critiquing<br>• Analyzing<br>• Creating<br>• Proving | Determine what sorts are most optimal for a given situation<br><br>Write code that implements parts of the various searches and sorts | Identify the differences and relative running time of various searches and sorts | Identify the various sorts (Selection, Insertion, Mergesort) and searches (Sequential, Binary) and explain how they work | Little evidence of reasoning or application to solve the problem<br><br>Does not meet the criteria in a level 1 |

## Reporting Strand 11: Computing in Context

| Standards | Meets | Proficient | Improvement Needed |
|---|---|---|---|
| VI | | Describe the impact of computing on all of the following:<br><br>● impact of applications using databases on an individual's right to privacy<br>● economic and legal impact of viruses and attacks on computer systems<br>● need for fault-tolerant and reliable systems for life-critical applications<br>● need for software engineering standards<br>● intellectual property and legal issues<br>● social and ethical ramifications of computer use | |