AP Computer Science Principles Pacing Calendar

AUGUST

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

SEPTEMBER

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

OCTOBER

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

NOVEMBER

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

DECEMBER

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

JANUARY

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

FEBRUARY

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28				

MARCH

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

APRIL

S	M	T	W	T	F	S
						1
12	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

MAY

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

First/Last Day of Semester
Progress Reports
Holidays/Non-Instructional

Intro to Programming	14 days
Control Structures	15 days
Functions and Parameters	13 days
The Internet, Digital Information	14 Days
Arrays & Data Structures	10 Days
Create Task	8 days
Cumulative Review	10 days
AP Exam	

22

Unit 1 Intro to Programming

Pacing:

August 24th - September 11

Subsection	Essential Knowledge	Lessons / Topics
CodeHS Unit 1: Introduction to Programming with Karel the Dog (3 weeks, 15 hours)		
Abstraction CodeHS Activities ***Abstraction***	AAP-3.B.1 AAP-3.B.7 AAP-3.B.2 CRD-2.G.1 AAP-3.B.3 DAT-1.A.2 AAP-3.B.4 DAT-1.A.5 AAP-3.B.6	Procedural Abstraction Modularity Program Reuse Digital Data (Bits) Reducing Complexity
Programming Style CodeHS Activities ***Intro to Programming*** ***Super Karel*** ***Ultra Karel*** ***Top-Down Design*** ***Commenting Your Code***	CRD-2.G.1 CRD-2.B.5 CRD-2.G.2 AAP-3.D.1 AAP-2.M.1 AAP-3.D.2 AAP-2.M.3 AAP-3.D.3 CRD-2.B.1 AAP-3.D.4 CRD-2.B.2 AAP-3.D.5	Program Documentation Using Existing Code and Libraries APIs Commenting Code
Control Structures CodeHS Activities ***If/Else Statements*** ***For Loops*** ***While Loops in Karel***	AAP-2.G.1 AAP-2.J.1 AAP-2.K.1	If/Else Statements (Selection) For Loops and While Loops (Iteration)
Debugging Strategies CodeHS Activities ***Functions in Karel*** ***Debugging Strategies***	CRD-2.I.1 CRD-2.I.2 CRD-2.I.3 CRD-2.I.5	Logic Errors Syntax Errors Run-Time Error Testing
Designing Algorithms CodeHS Activities ***Karel Algorithms***	AAP-2.A.4 AAP-2.M.2 AAP-2.B.1 AAP-4.A.2 AAP-2.B.2 AAP-4.A.4 AAP-2.B.6 AAP-4.A.5 AAP-2.B.7 AAP-4.A.6	Sequencing, Selection, Iteration Clarity and Readability Using Existing Algorithms Optimization and Efficiency
CodeHS Unit 2: Practice PT: Pair-Programming Paint (3 days, 3 hours)		
Collaboration and Communication	CRD-1.A.3 CRD-2.F.7 CRD-1.A.4 CRD-2.G.1	Collaboration Diverse Perspectives

Unit 1 Intro to Programming

Pacing:
August 24th - September 11

Subsection	**Essential Knowledge**	**Lessons / Topics**
	CRD-1.B.2 CRD-2.G.3 CRD-1.C.1 CRD-2.G.4 CRD-2.F.5 CRD-2.H.1 CRD-2.G.5 CRD-2.H.2 CRD-2.F.6	Bias Avoidance Pair-Programming Design and Planning Program Documentation Acknowledgement of Reused Code

CodeHS Unit 3: Programming with JavaScript (2 weeks, 10 hours)

Programming Languages CodeHS Activities ***What is Code?*** ***Uses of Programs***	AAP-2.A.2 AAP-2.A.3 CRD-1.A.1 CRD-1.A.2 CRD-2.B.1	What is Programming? Pseudocode Programming Languages Computing Innovations
Variables CodeHS Activities ***Variables***	AAP-1.A.1 AAP-1.B.2 AAP-1.A.2 AAP-1.B.3 AAP-1.A.3 DAT-1.A.1 AAP-1.A.4 AAP-1.B.1	Variable Names Assignment Operators Data Types Variables as Abstractions
Arithmetic Expressions CodeHS Activities ***Basic Math in JavaScript***	CRD-2.B.4 AAP-2.B.3 CRD-2.I.5 AAP-2.B.4 CRD-2.J.1 AAP-2.B.5 CRD-2.J.2 AAP-2.C.1 CRD-2.J.3 AAP-2.C.2 AAP-2.A.1 AAP-2.C.3 AAP-2.A.2 AAP-2.C.4 AAP-2.A.3 AAP-2.D.1 AAP-2.A.4 AAP-2.D.2	Program Behavior Testing using Inputs Arithmetic Expressions Order of Operations Modulus String Concatenation
User Input CodeHS Activities ***User Input*** ***Mouse Events: Mouse Clicked*** ***Key Events***	AAP-1.C.4 CRD-2.C.5 AAP-3.A.6 CRD-2.C.6 AAP-3.A.9 CRD-2.D.2 CRD-2.C.2 CRD-2.C.3	Strings User Input Program Output Events Mouse and Key Events

Unit 2 Algorithm

Pacing:
September 14-October 2

Subsection	Essential Knowledge	Lessons / Topics
CodeHS Unit 4: JavaScript Control Structures (2 weeks, 10 hours)		
Comparison Operators CodeHS Activities ***Booleans*** ***Comparison Operators***	AAP-2.E.1 AAP-2.F.5 AAP-2.F.4 AAP-2.E.2 AAP-2.F.1 AAP-2.F.2 AAP-2.F.3	Booleans Relational Operators Operands
Selection CodeHS Activities ***If Statements*** ***Random Numbers***	AAP-2.G.1 AAP-2.I.2 AAP-2.H.1 AAP-2.L.3 AAP-2.H.2 AAP-2.L.4 AAP-2.H.3 AAP-3.E.2 AAP-2.I.1	Selection Conditional Statements Nested Conditionals Equivalent Boolean Statements Random Numbers
Iteration CodeHS Activities ***While Loops***	AAP-2.K.2 AAP-2.L.1 AAP-2.K.3 AAP-2.L.2 AAP-2.K.4 AAP-2.L.5 AAP-2.K.5	Iteration Loops Different but Equivalent Algorithms

Unit 3 Abstraction

Pacing:
October 5 - October 23

Subsection	Essential Knowledge		Lessons / Topics
CodeHS Unit 5: Functions and Parameters (2 weeks, 10 hours)			
Functions and Parameters CodeHS Activities ***Functions and Parameters 1*** ***Functions and Parameters 2*** ***Functions and Return Values 1*** ***Functions and Return Values 2***	CRD-2.C.6 CRD-2.D.2 CRD-2.B.3 CRD-2.C.4 AAP-3.A.1 AAP-3.C.2 AAP-3.A.2 AAP-2.M.2	AAP-3.A.3 AAP-3.A.4 AAP-3.B.5 AAP-3.C.1	User and Application Input Program Output Procedures Parameters Return Values Using Existing Algorithms

Unit 4 Internet and Digital Information

Pacing:
October 26 - November 13

Subsection	Essential Knowledge	Lessons / Topics
CodeHS Unit 8: Digital Information (3 weeks, 15 hours)		
Number Systems CodeHS Activities ***Intro to Digital Information*** ***Number Systems***	CRD-2.C.1 DAT-1.A.7 CRD-2.D.1 DAT-1.B.1 CRD-2.J.2 DAT-1.B.2 CRD-2.J.3 DAT-1.B.3 CRD-2.I.4 DAT-1.C.1 DAT-1.A.2 DAT-1.C.2 DAT-1.A.3 DAT-1.C.3 DAT-1.A.4 DAT-1.C.4 DAT-1.A.5 DAT-1.C.5 DAT-1.A.6	Computing Devices Abstraction Program Input and Output Bits and Bytes Overflow Errors Range of Value Limits Binary and Decimal Systems
Data Compression CodeHS Activities ***Data Compression*** ***Lossy Compression***	DAT-1.A.8 DAT-1.D.4 DAT-1.A.9 DAT-1.D.5 DAT-1.A.10 DAT-1.D.6 DAT-1.D.1 DAT-1.D.7 DAT-1.D.2 DAT-1.D.8 DAT-1.D.3	Lossless Data Lossy Data Digital and Analog Data
Cryptography CodeHS Activities ***Cryptography***	AAP-4.B.1 AAP-4.B.2 AAP-4.B.3 IOC-2.B.8 IOC-2.B.5	Decidable Problems Computer Viruses Encryption
CodeHS Unit 11: The Internet (2 weeks, 10 hours)		
Internet Hardware and Addresses CodeHS Activities ***Welcome to the Internet*** ***Internet Hardware*** ***Internet Addresses***	CSN-1.A.1 CSN-1.A.8 CSN-1.A.2 CSN-1.B.3 CSN-1.A.3 CSN-1.B.4 CSN-1.A.7 CSN-1.A.4	Protocols Computing Devices Computer Networks Bandwidth
Routing CodeHS Activities ***Routing***	CSN-1.A.5 CSN-1.E.2 CSN-1.A.6 CSN-1.E.3 CSN-1.B.5 CSN-1.E.4 CSN-1.B.6 CSN-1.E.5 CSN-1.B.7 CSN-1.E.6 CSN-1.E.1 CSN-1.E.7	Routing Scalability Fault-Tolerance Redundancy

Unit 4 Internet and Digital Information

Pacing:
October 26 - November 13

Subsection	Essential Knowledge	Lessons / Topics
Packets and Protocols CodeHS Activities ***Packets and Protocols***	CSN-1.B.1 CSN-1.D.1 CSN-1.B.2 CSN-1.D.2 CSN-1.C.1 CSN-1.D.3 CSN-1.C.2 DAT-2.B.1 CSN-1.C.3 DAT-2.B.3 CSN-1.C.4 DAT-2.B.5	Datastreams Packets IP, TCP, UDP HTTP Metadata
Computing Systems CodeHS Activities ***Sequential, Parallel & Distributed***	DAT-2.C.7 CSN-2.A.6 DAT-2.C.8 CSN-2.A.7 CSN-2.A.1 CSN-2.B.1 CSN-2.A.2 CSN-2.B.2 CSN-2.A.3 CSN-2.B.3 CSN-2.A.4 CSN-2.B.4 CSN-2.A.5 CSN-2.B.5	Parallel Systems Scalability of Systems Sequential Computing Parallel Computing Distributed Computing Efficiency of Solutions Speedup
Impact of the Internet CodeHS Activities ***The Impact of the Internet*** ***Creative Credit and Copyright***	IOC-1.A.1 IOC-1.E.2 IOC-1.A.3 IOC-1.E.3 IOC-1.A.4 IOC-1.E.4 IOC-1.A.5 IOC-1.E.5 IOC-1.B.1 IOC-1.E.6 IOC-1.B.2 IOC-1.F.1 IOC-1.B.3 IOC-1.F.2 IOC-1.B.4 IOC-1.F.3 IOC-1.B.5 IOC-1.F.4 IOC-1.B.6 IOC-1.F.5 IOC-1.C.1 IOC-1.F.6 IOC-1.C.2 IOC-1.F.7 IOC-1.C.3 IOC-1.F.9 IOC-1.C.4 IOC-1.F.10 IOC-1.C.5 IOC-1.F.11 IOC-1.E.1	Computing Innovations Unintended Effects Impact on Society Rapid Sharing Digital Divide Citizen Science Crowdsourcing Creative Credit and Copyright
Cybersecurity CodeHS Activities ***Cybersecurity***	IOC-1.F.8 IOC-2.B.5 IOC-2.A.1 IOC-2.B.6 IOC-2.A.7 IOC-2.B.7 IOC-2.A.8 IOC-2.B.9 IOC-2.A.9 IOC-2.B.10 IOC-2.A.11 IOC-2.B.11 IOC-2.A.12 IOC-2.C.1 IOC-2.A.13 IOC-2.C.2	Legal and Ethical Concerns Personally Identifiable Info (PII) Digital Footprint Authentication Certificate Authorities (CAs) Computer Viruses Malware

Unit 4 Internet and Digital Information

Pacing:
October 26 - November 13

Subsection	Essential Knowledge	Lessons / Topics
	IOC-2.A.15 IOC-2.C.3 IOC-2.B.1 IOC-2.C.4 IOC-2.B.2 IOC-2.C.5 IOC-2.B.3 IOC-2.C.6 IOC-2.B.4 IOC-2.C.7	Phishing Keylogging Rogue Access Points Encryption

CodeHS Unit 13: Data (1 week, 5 hours)

Visualizing and Interpreting Data CodeHS Activities ***Getting Started with Data*** ***Visualizing and Interpreting Data***	DAT-2.A.1 DAT-2.D.5 DAT-2.A.2 DAT-2.D.6 DAT-2.C.1 DAT-2.E.1 DAT-2.D.1 DAT-2.E.2 DAT-2.D.2 DAT-2.E.3 DAT-2.D.3 DAT-2.E.5 DAT-2.D.4	Filtering and Cleaning Data Patterns and Trends Search Tools Tables, Diagrams and Displays Interactive Visualizations Combining Data Sources
Collecting Data and Data Limitations CodeHS Activities ***Data Collection and Limitations***	DAT-2.A.3 DAT-2.C.2 DAT-2.A.4 DAT-2.C.3 DAT-2.B.1 DAT-2.C.4 DAT-2.B.2 DAT-2.C.5 DAT-2.B.3 DAT-2.C.6 DAT-2.B.4 DAT-2.D.6 DAT-2.B.5 CRD-2.F.3	Metadata Correlation Using a Variety of Sources Incomplete or Invalid Data Bias Surveys, Testing, Interviews

Unit 6 Create Task

Pacing:
December 7 - December 18

Subsection	Essential Knowledge		Lessons / Topics
CodeHS Unit 15 & 16: Explore MCQ Practice and Create Performance Task (3 weeks, 15 hours)			
AP CSP Explore Task **Practice**	IOC-2.A.3 IOC-2.A.4 IOC-2.A.5 IOC-2.A.6	IOC-2.A.14 IOC-1.F.11 CRD-1.A.1 CRD-1.A.2	Computing Innovations Data Input and Output Data Privacy and Security
Prepare for Create PT	ALL		Review Course Content Incremental Development Documentation Debugging Collaborative Development
Create PT			12 hours of class time to conduct Create PT
Design Thinking CodeHS Activities ***Intro to Design Thinking***	CRD-1.A.4 CRD-1.A.5 CRD-1.A.6 CRD-2.A.1 CRD-2.A.2 CRD-2.E.1 CRD-2.E.2	CRD-2.E.4 CRD-2.F.1 CRD-2.F.2 CRD-2.F.5 CRD-2.F.6 CRD-2.F.7 IOC-1.A.2	Computing Innovations Development Process Program Specifications Design Phase Communication Collaboration
Brainstorm, Prototype & Test CodeHS Activities ***Prototype*** ***Test***	CRD-2.E.2 CRD-2.F.7 CRD-1.A.5 CRD-1.A.6 CRD-1.A.4 CRD-2.E.3	CRD-2.F.4 CRD-2.F.3 IOC-1.D.1 IOC-1.D.2 IOC-1.D.3 IOC-1.F.11	Development Process User Testing User Research Diverse Perspectives Iterative Development Human Biases Legal and Ethical Concerns
Project Prep and Development CodeHS Activities ***Project Prep and Development***	CRD-1.B.1		Online Collaboration Tools

Unit 5 Arrays

Pacing:
November 16 - December 4

Subsection	Essential Knowledge	Lessons / Topics
CodeHS Unit 7: Basic Data Structures (2 weeks, 10 hours)		
Basic Data Structures CodeHS Activities ***Intro to Lists/Arrays*** ***Indexing Into an Array*** ***Removing an Element***	AAP-1.A.1 AAP-1.C.1 AAP-1.C.2 AAP-1.C.3 AAP-1.D.6 AAP-1.D.7 AAP-1.D.8 AAP-2.N.2 AAP-2.N.1	Data Values Lists and Elements Indices List Procedures
Data Abstractions CodeHS Activities ***Adding/Removing From Arrays*** ***Array Length and Looping***	AAP-1.D.1 AAP-1.D.5 DAT-2.E.4 AAP-1.D.2 AAP-1.D.3 AAP-1.D.4 DAT-2.E.2 DAT-2.D.4 DAT-2.E.5	Data Abstraction Translating and Transforming Data Filtering and Cleaning Patterns
Traversing a List CodeHS Activities ***Array Length and Looping*** ***Iterating Over an Array*** ***Removing an Element***	DAT-2.D.6 AAP-2.O.1 AAP-2.O.2 AAP-3.C.1 AAP-3.C.2 AAP-3.A.6 AAP-2.O.3 AAP-3.A.5 AAP-3.A.7 AAP-3.A.8 AAP-3.E.1	Extract and Modify Information Traversing a List Iteration Statements
Algorithm Efficiency CodeHS Activities ***Array Length and Looping*** ***Finding an Element in a List***	AAP-2.O.4 DAT-2.D.3 AAP-2.O.5 AAP-2.P.1 AAP-2.P.2 AAP-2.P.3	Using Existing Algorithms Search Tools Linear Search

Unit 5 Arrays

Pacing:
November 16 - December 4

Subsection	Essential Knowledge	Lessons / Topics
	AAP-4.A.1 AAP-4.A.3 AAP-4.A.7 AAP-4.A.8 AAP-4.A.9	Binary Search Algorithm Efficiency Heuristics
SImulation CodeHS Activities ***Simulation***	AAP-3.F.1 AAP-3.F.2 AAP-3.F.3 AAP-3.F.4 AAP-3.F.5 AAP-3.F.6 AAP-3.F.7 AAP-3.F.8	 Simulations as Abstractions Bias in Simulations Random Number Generators

Big Idea 1: Creative Development (CRD)

CRD-1.A.1 A computing innovation includes a program as an integral part of its function.

CRD-1.A.2 A computing innovation can be physical (e.g., self-driving car), nonphysical computing software (e.g., picture editing software), or a nonphysical computing concept (e.g., e-commerce).

CRD-1.A.3 Effective collaboration produces a computing innovation that reflects the diversity of talents and perspectives of those who designed it.

CRD-1.A.4 Collaboration that includes diverse perspectives helps avoid bias in the development of computing innovations.

CRD-1.A.5 Consultation and communication with users are important aspects of the development of computing innovations.

CRD-1.A.6 Information gathered from potential users can be used to understand the purpose of a program from diverse perspectives and to develop a program that fully incorporates these perspectives.

CRD-1.B.1 Online tools support collaboration by allowing programmers to share and provide feedback on ideas and documents.

CRD-1.B.2 Common models such as pair programming exist to facilitate collaboration.

CRD-1.C.1 Effective collaborative teams practice interpersonal skills, including but not limited to:

§§ communication
§§ consensus building
§§ conflict resolution
§§ negotiation

CRD-2.A.1 The purpose of computing innovations is to solve problems or to pursue interests through creative expression.

CRD-2.A.2 An understanding of the purpose of a computing innovation provides developers with an improved ability to develop that computing innovation.

CRD-2.B.1 A program is a collection of program statements that performs a specific task when run by a computer. A program is often referred to as software.

CRD-2.B.2 A code segment is a collection of program statements that is part of a program.

CRD-2.B.3 A program needs to work for a variety of inputs and situations.

CRD-2.B.4 The behavior of a program is how a program functions during execution and is often described by how a user interacts with it.

CRD-2.B.5 A program can be described broadly by what it does, or in more detail by both what the program does and how the program statements accomplish this function.

CRD-2.C.1 Program inputs are data sent to a computer for processing by a program. Input can come in a variety of forms, such as tactile, audio, visual, or text.

CRD-2.C.2 An event is associated with an action and supplies input data to a program.

CRD-2.C.3 Events can be generated when a key is pressed, a mouse is clicked, a program is started, or any other defined action occurs that affects the flow of execution.

CRD-2.C.4 Inputs usually affect the output produced by a program.

CRD-2.C.5 In event-driven programming, program statements are executed when triggered rather than through the sequential flow of control.

CRD-2.C.6 Input can come from a user or other programs.

CRD-2.D.1 Program outputs are any data sent from a program to a device. Program output can come in a variety of forms, such as tactile, audio, visual, or text.

CRD-2.D.2 Program output is usually based on a program's input or prior state (e.g., internal values).

CRD-2.E.1 A development process can be ordered and intentional, or exploratory in nature.

CRD-2.E.2 There are multiple development processes. The following phases are commonly used when developing a program:
§§ investigating and reflecting
§§ designing
§§ prototyping
§§ testing

CRD-2.E.3 A development process that is iterative requires refinement and revision based on feedback, testing, or reflection throughout the process. This may require revisiting earlier phases of the process.

CRD-2.E.4 A development process that is incremental is one that breaks the problem into smaller pieces and makes sure each piece

Big Idea 2: Data (DAT)

DAT-1.A.1 Data values can be stored in variables, lists of items, or standalone constants and can be passed as input to (or output from) procedures.

DAT-1.A.2 Computing devices represent data digitally, meaning that the lowest-level components of any value are bits.

DAT-1.A.3 Bit is shorthand for binary digit and is either 0 or 1.

DAT-1.A.4 A byte is 8 bits.

DAT-1.A.5 Abstraction is the process of reducing complexity by focusing on the main idea. By hiding details irrelevant to the question at hand and bringing together related and useful details, abstraction reduces complexity and allows one to focus on the idea.

DAT-1.A.6 Bits are grouped to represent abstractions. These abstractions include, but are not limited to, numbers, characters, and color.

DAT-1.A.7 The same sequence of bits may represent different types of data in different contexts.

DAT-1.A.8 Analog data have values that change smoothly, rather than in discrete intervals, over time. Some examples of analog data include pitch and volume of music, colors of a painting, or position of a sprinter during a race.

DAT-1.A.9 The use of digital data to approximate real-world analog data is an example of abstraction.

DAT-1.A.10 Analog data can be closely approximated digitally using a sampling technique, which means measuring values of the analog signal at regular intervals called samples. The samples are measured to figure out the exact bits required to store each sample.

DAT-1.B.1 In many programming languages, integers are represented by a fixed number of bits, which limits the range of integer values and mathematical operations on those values. This limitation can result in overflow or other errors.

DAT-1.B.2 Other programming languages provide an abstraction through which the size of representable integers is limited only by the size of the computer's memory; this is the case for the language defined in the exam reference sheet.

DAT-1.B.3 In programming languages, the fixed number of bits used to represent real numbers limits the range and mathematical operations on these values; this limitation can result in roundoff and other errors. Some real numbers are represented as approximations in computer storage.

DAT-1.C.1 Number bases, including binary and decimal, are used to represent data.

DAT-1.C.2 Binary (base 2) uses only combinations of the digits zero and one.

DAT-1.C.3 Decimal (base 10) uses only combinations of the digits 0 – 9.

DAT-1.C.4 As with decimal, a digit's position in the binary sequence determines its numeric value. The numeric value is equal to the bit's value (0 or 1) multiplied by the place value of its position.

DAT-1.C.5 The place value of each position is determined by the base raised to the power of the position. Positions are numbered starting at the rightmost position with 0 and increasing by 1 for continued on next page each subsequent position to the left.

DAT-1.D.1 Data compression can reduce the size (number of bits) of transmitted or stored data.

DAT-1.D.2 Fewer bits does not necessarily mean less information.

Big Idea 3: Algorithms and Programming (AAP)

AAP-1.A.1
A variable is an abstraction inside a program that can hold a value. Each variable has associated data storage that represents one value at a time, but that value can be a list or other collection that in turn contains multiple values.

AAP-1.A.2
Using meaningful variable names helps with the readability of program code and understanding of what values are represented by the variables.

AAP-1.A.3
Some programming languages provide types to represent data, which are referenced using variables. These types include numbers, Booleans, lists, and strings.

AAP-1.A.4
Some values are better suited to representation using one type of datum rather than another.

AAP-1.B.1
The assignment operator allows a program to change the value represented by a variable.

AAP-1.B.2
The exam reference sheet provides the " <-" operator to use for assignment.

\

evaluates expression and then assigns a copy of the result to the variable a.

AAP-1.B.3
The value stored in a variable will be the most recent value assigned. For example:

a

b a

1

a 2
display(b)
still displays 1.

CRD-1.A.1	A computing innovation includes a program as an integral part of its funct	3.2 Uses of Programs 15.1 The Impacts of Computing
CRD-1.A.2	A computing innovation can be physical (i.e., self-driving car), non-physica	3.2 Uses of Programs 15.1 The Impacts of Computing
CRD-1.A.3	Effective collaboration produces a computing innovation that reflects the	2.1 Practice PT: Pair-Programming Paint!
CRD-1.A.4	Collaboration that includes diverse perspectives helps to avoid bias in the	2.1 Practice PT: Pair-Programming Paint! 18.1 Intro to Design Thinking 18.3 Test
CRD-1.A.5	Consultation and communication with users is an important aspect of the	18.1 Intro to Design Thinking 18.3 Test
CRD-1.A.6	Research gathered from users can be used to understand the purpose of	18.1 Intro to Design Thinking 18.3 Test
CRD-1.B.1	Online tools support collaboration by allowing programmers to virtually sl	18.4 Project Prep and Development
CRD-1.B.2	Common models such as pair programming exist to facilitate collaboratio	2.1 Practice PT: Pair-Programming Paint!
CRD-1.C.1	Effective collaborative teams practice interpersonal skills including but nc	2.1 Practice PT: Pair-Programming Paint!
CRD-2.A.1	The purpose of computing innovations is to solve problems or pursue int	18.1 Intro to Design Thinking
CRD-2.A.2	An understanding of the purpose of a computing innovation provides dev	18.1 Intro to Design Thinking
CRD-2.B.1	A program is a collection of program statements that performs a specific	1.2 Introduction to Programming With Karel 3.1 What is Code
CRD-2.B.2	A code segment refers to a collection of program statements that are par	1.8 Commenting Your Code
CRD-2.B.3	A program needs to work for a variety of inputs and situations.	5.1 Functions and Parameters 1
CRD-2.B.4	The behavior of a program is how a program functions during execution a	3.6 Basic Math in JavaScript
CRD-2.B.5	A program can be described broadly by what it does or in more detail by	1.8 Commenting Your Code
CRD-2.C.1	Program input is data that are sent to a computer for processing by a pro	8.1 Intro to Digital Information
CRD-2.C.2	An event is associated with an action and supplies input data to a prograi	3.8 Mouse Events: Mouse Clicked 3.9 Key Events 27.5 Mouse Events: Mouse Clicked
CRD-2.C.3	Events can be generated when a key is pressed, a mouse is clicked, a pr	3.8 Mouse Events: Mouse Clicked 3.9 Key Events 27.5 Mouse Events: Mouse Clicked
CRD-2.C.4	Inputs usually affect the output produced by a program.	5.1 Functions and Parameters 1
CRD-2.C.5	In event-driven programming, program statements are executed when tri	3.8 Mouse Events: Mouse Clicked 3.9 Key Events 27.5 Mouse Events: Mouse Clicked
CRD-2.C.6	Input can come from a user or other applications.	3.5 User Input 5.1 Functions and Parameters 1
CRD-2.D.1	Program output is any data that are sent from a program to a device. Proc	8.1 Intro to Digital Information
CRD-2.D.2	Program output is usually based on a program's input or prior state (such	3.5 User Input 5.1 Functions and Parameters 1
CRD-2.E.1	A development process could be ordered and intentional, or exploratory	18.1 Intro to Design Thinking
CRD-2.E.2	There are multiple development processes. The following phases are cor	18.1 Intro to Design Thinking 18.2 Prototype 18.3 Test
CRD-2.E.3	A development process that is iterative requires refinement and revision	18.2 Prototype
CRD-2.E.4	A development process that is incremental is one that breaks the probler	18.1 Intro to Design Thinking
CRD-2.F.1	The design of a program incorporates investigation to determine the requ	18.1 Intro to Design Thinking
CRD-2.F.2	Investigation in a development process is useful in understanding, identif	18.1 Intro to Design Thinking
CRD-2.F.3	Some ways investigation can be performed are as follows: collecting data	13.3 Data Collection & Limitations 18.3 Test
CRD-2.F.4	Program requirements describe how a program functions, and may incluc	18.2 Prototype
CRD-2.F.5	A program specification defines the requirements for the program.	2.1 Practice PT: Pair-Programming Paint! 18.1 Intro to Design Thinking
CRD-2.F.6	In a development process, the design phase outlines how to accomplish	2.1 Practice PT: Pair-Programming Paint! 18.1 Intro to Design Thinking
CRD-2.F.7	The design phase of a program includes: brainstorming; planning and sto	2.1 Practice PT: Pair-Programming Paint! 18.1 Intro to Design Thinking 18.2 Prototype 18.3 Test
CRD-2.G.1	Program documentation is a written description of the function of a code	1.9 Abstraction 1.10 Super Karel 2.1 Practice PT: Pair-Programming Paint!
CRD-2.G.2	Comments are a form of program documentation written into the progran	1.8 Commenting Your Code
CRD-2.G.3	Programmers should document a program throughout the process of dev	2.1 Practice PT: Pair-Programming Paint!
CRD-2.G.4	Program documentation helps in developing and maintaining correct pro	2.1 Practice PT: Pair-Programming Paint!
CRD-2.G.5	Not all programming environments support comments, so other methods	2.1 Practice PT: Pair-Programming Paint!
CRD-2.H.1	It is important to acknowledge any code segments that were developed (	2.1 Practice PT: Pair-Programming Paint!
CRD-2.H.2	Acknowledgement of code segment(s) written by someone else and usec	2.1 Practice PT: Pair-Programming Paint!
CRD-2.I.1	A logic error is a mistake in the algorithm or program that causes it to bel	1.5 Functions in Karel 1.16 Debugging Strategies
CRD-2.I.2	A syntax error is a mistake in the program where the rules of the program	1.5 Functions in Karel 1.16 Debugging Strategies
CRD-2.I.3	A run-time error is a mistake in the program that occurs during the execut	1.16 Debugging Strategies
CRD-2.I.4	An overflow error is an error that occurs when the computer attempts to l	8.2 Number Systems
CRD-2.I.5	The following are effective ways to find and correct errors: test-cases; hai	1.5 Functions in Karel 1.16 Debugging Strategies 3.6 Basic Math in JavaScript
CRD-2.J.1	In the development process, testing uses defined inputs to ensure that ar	3.6 Basic Math in JavaScript
CRD-2.J.2	Defined inputs used to test a program should demonstrate the different e	3.6 Basic Math in JavaScript 8.2 Number Systems
CRD-2.J.3	Program requirements are needed to identify appropriate defined inputs	3.6 Basic Math in JavaScript 8.2 Number Systems
DAT-1.A.1	Data are values that can be stored in variables, passed as input to procec	3.4 Variables 3.7 Using Graphics in JavaScript 7.1 Intro to Lists/Arrays
DAT-1.A.2	Computing devices represent data digitally, which means that the lowest	1.9 Abstraction 8.1 Intro to Digital Information
DAT-1.A.3	A "bit" is shorthand for "binary digit," and is either 0 or 1.	8.1 Intro to Digital Information 8.2 Number Systems
DAT-1.A.4	A "byte" is 8 bits.	8.2 Number Systems
DAT-1.A.5	Abstraction is the process of reducing complexity by focusing on the mai	1.9 Abstraction 8.1 Intro to Digital Information
DAT-1.A.6	Bits are grouped to represent abstractions. These abstractions include, b	8.1 Intro to Digital Information
DAT-1.A.7	The same sequence of bits may represent different types of data in differ	8.1 Intro to Digital Information
DAT-1.A.8	Analog data are data with values that change continuously, or smoothly, c	8.9 Lossy Compression
DAT-1.A.9	The use of digital data to approximate real-world analog data is an examp	8.9 Lossy Compression
DAT-1.A.10	Analog data can be closely approximated digitally using a sampling techr	8.9 Lossy Compression
DAT-1.B.1	In many programming languages, the fixed number of bits used to repres	8.2 Number Systems
DAT-1.B.2	Other programming languages provide an abstraction through which the	8.2 Number Systems
DAT-1.B.3	In programming languages, the fixed number of bits used to represent re	8.2 Number Systems
DAT-1.C.1	Number bases, including binary and decimal, are used to represent data.	8.2 Number Systems
DAT-1.C.2	Binary (base 2) uses only combinations of the digits zero and one.	8.2 Number Systems
DAT-1.C.3	Decimal (base 10) uses only combinations of the digits 0 – 9.	8.2 Number Systems
DAT-1.C.4	As with decimal, a digit's position within the binary sequence determines	8.2 Number Systems
DAT-1.C.5	The place value of each position is determined by the base raised to the	8.2 Number Systems
DAT-1.D.1	Data compression can reduce the size (number of bits) of the data transm	8.8 Data Compression
DAT-1.D.2	Fewer bits does not necessarily mean less information.	8.8 Data Compression
DAT-1.D.3	The amount of reduction in bits by compression depends on both the am	8.8 Data Compression
DAT-1.D.4	Lossless data compression algorithms can usually reduce the number of l	8.8 Data Compression
DAT-1.D.5	Lossy data compression algorithms can significantly reduce the number c	8.9 Lossy Compression
DAT-1.D.6	Lossy data compression algorithms can usually reduce the number of bits	8.9 Lossy Compression
DAT-1.D.7	In situations where quality or ability to reconstruct the original is maximall	8.9 Lossy Compression

works before adding it to the whole.

CRD-2.F.1 The design of a program incorporates investigation to determine its requirements.

CRD-2.F.2 Investigation in a development process is useful for understanding and identifying the program constraints, as well as the concerns and interests of the people who will use the program.

CRD-2.F.3 Some ways investigation can be performed are as follows:
§§ collecting data through surveys
§§ user testing
§§ interviews
§§ direct observations

CRD-2.F.4 Program requirements describe how a program functions and may include a description of user interactions that a program must provide.

CRD-2.F.5 A program's specification defines the requirements for the program.

CRD-2.F.6 In a development process, the design phase outlines how to accomplish a given program specification.

CRD-2.F.7 The design phase of a program may include:
§§ brainstorming
§§ planning and storyboarding
§§ organizing the program into modules and functional components
§§ creation of diagrams that represent the layouts of the user interface
§§ development of a testing strategy for the program

CRD-2.G.1 Program documentation is a written description of the function of a code segment, event, procedure, or program and how it was developed.

CRD-2.G.2 Comments are a form of program documentation written into the program to be read by people and do not affect how a program runs.

CRD-2.G.3 Programmers should document a program throughout its development.

CRD-2.G.4 Program documentation helps in developing and maintaining correct programs when working individually or in collaborative programming environments.

CRD-2.G.5 Not all programming environments support comments, so other methods of documentation may be required.

CRD-2.H.1 It is important to acknowledge any code segments that were developed collaboratively or by another source.

CRD-2.H.2 Acknowledgement of a code segment(s) written by someone else and used in a program can be in the program documentation. The acknowledgement should include the origin or original author's name.

CRD-2.I.1 A logic error is a mistake in the algorithm or program that causes it to behave incorrectly or unexpectedly.

CRD-2.I.2 A syntax error is a mistake in the program where the rules of the programming language are not followed.

CRD-2.I.3 A run-time error is a mistake in the program that occurs during the execution of a program. Programming languages define their own run-time errors.

CRD-2.I.4 An overflow error is an error that occurs when a computer attempts to handle a number that is outside of the defined range of values.

CRD-2.I.5 The following are effective ways to find and correct errors:
§§ test cases
§§ hand tracing
§§ visualizations
§§ debuggers
§§ adding extra output statement

CRD-2.J.1 In the development process, testing uses defined inputs to ensure that an algorithm or program is producing the expected outcomes. Programmers use the results from testing to revise their algorithms or programs.

CRD-2.J.2 Defined inputs used to test a program should demonstrate the different expected outcomes that are at or just beyond the extremes (minimum and maximum) of input data.

CRD-2.J.3 Program requirements are needed to identify appropriate defined inputs for testing.

DAT-1.D.3 The amount of size reduction from compression depends on both the amount of redundancy in the original data representation and the compression algorithm applied.

DAT-1.D.4 Lossless data compression algorithms can usually reduce the number of bits stored or transmitted while guaranteeing complete reconstruction of the original data.

DAT-1.D.5 Lossy data compression algorithms can significantly reduce the number of bits stored or transmitted but only allow reconstruction of an approximation of the original data.

DAT-1.D.6 Lossy data compression algorithms can usually reduce the number of bits stored or transmitted more than lossless compression algorithms.

DAT-1.D.7 In situations where quality or ability to reconstruct the original is maximally important, lossless compression algorithms are typically chosen.

DAT-1.D.8 In situations where minimizing data size or transmission time is maximally important, lossy compression algorithms are typically chosen.

DAT-2.A.1 Information is the collection of facts and patterns extracted from data.

DAT-2.A.2 Data provide opportunities for identifying trends, making connections, and addressing problems.

DAT-2.A.3 Digitally processed data may show correlation between variables. A correlation found in data does not necessarily indicate that a causal relationship exists. Additional research is needed to understand the exact nature of the relationship.

DAT-2.A.4 Often, a single source does not contain the data needed to draw a conclusion. It may be necessary to combine data from a continued on next page variety of sources to formulate a conclusion.

DAT-2.B.1 Metadata are data about data. For example, the piece of data may be an image, while the metadata may include the date of creation or the file size of the image.

DAT-2.B.2 Changes and deletions made to metadata do not change the primary data.

DAT-2.B.3 Metadata are used for finding, organizing, and managing information.

DAT-2.B.4 Metadata can increase the effective use of data or data sets by providing additional information.

DAT-2.B.5 Metadata allow data to be structured and organized.

DAT-2.C.1 The ability to process data depends on the capabilities of the users and their tools.

DAT-2.C.2 Data sets pose challenges regardless of size, such as:
§§ the need to clean data
§§ incomplete data
§§ invalid data
§§ the need to combine data sources

DAT-2.C.3 Depending on how data were collected, they may not be uniform. For example, if users enter data into an open field, the way they choose to abbreviate, spell, or capitalize something may vary from user to user.

DAT-2.C.4 Cleaning data is a process that makes the data uniform without changing their meaning (e.g., replacing all equivalent abbreviations, spellings, and capitalizations with the same word).

DAT-2.C.5 Problems of bias are often created by the type or source of data being collected. Bias is not eliminated by simply collecting more data.

DAT-2.C.6 The size of a data set affects the amount of information that can be extracted from it.

DAT-2.C.7 Large data sets are difficult to process using a single computer and may require parallel systems.

DAT-2.C.8 Scalability of systems is an important consideration when working with data sets, as the computational capacity of a system affects how data sets can be processed and stored.

DAT-1.D.8	In situations where minimizing data size or short transmission time is max	8.9 Lossy Compression
DAT-2.A.1	Information is the collection of facts and patterns extracted from data.	13.1 Getting Started with Data 13.2 Visualizing and Interpreting Data
DAT-2.A.2	Data provide opportunities for identifying trends, making connections, an	13.1 Getting Started with Data 13.2 Visualizing and Interpreting Data
DAT-2.A.3	Digitally processed data may show correlation between variables. A corre	13.3 Data Collection & Limitations
DAT-2.A.4	Often a single data source does not contain the necessary data to draw a	13.3 Data Collection & Limitations
DAT-2.B.1	Metadata are data about data. Metadata is associated with the primary da	11.7 Packets and Protocols 13.3 Data Collection & Limitations
DAT-2.B.2	Changes and deletions made to metadata do not change the primary dat	13.3 Data Collection & Limitations
DAT-2.B.3	Metadata are used for finding, organizing and managing information.	11.7 Packets and Protocols 13.3 Data Collection & Limitations
DAT-2.B.4	Metadata can increase the effective use of data or data sets by providing	13.3 Data Collection & Limitations
DAT-2.B.5	Metadata allows data to be structured and organized.	11.7 Packets and Protocols 13.3 Data Collection & Limitations
DAT-2.C.1	The ability to process data depends on the capabilities of the users and t	13.1 Getting Started with Data
DAT-2.C.2	Data pose challenges regardless of size. Such as: the need to clean data;	13.3 Data Collection & Limitations
DAT-2.C.3	Depending on how data are being collected, the data may not be uniform	13.3 Data Collection & Limitations
DAT-2.C.4	Cleaning data is a process that makes the data uniform without changing	13.3 Data Collection & Limitations
DAT-2.C.5	Problems of bias are often caused by the type or source of data that is be	13.3 Data Collection & Limitations
DAT-2.C.6	The size of the data set affects the amount of information that can be extr	13.3 Data Collection & Limitations
DAT-2.C.7	Large data sets are difficult to process using a single computer and may r	11.8 Sequential, Parallel & Distributed Computing
DAT-2.C.8	Scalability of systems is an important consideration when working with la	11.8 Sequential, Parallel & Distributed Computing
DAT-2.D.1	Programs can be used to process data to acquire information.	13.1 Getting Started with Data 13.2 Visualizing and Interpreting Data
DAT-2.D.2	Tables, diagrams, and textual displays or other visual tools can be used ir	13.2 Visualizing and Interpreting Data
DAT-2.D.3	Search tools are useful for efficiently finding information.	7.4 Array Length and Looping Through Arrays 13.1 Getting Started with Data 13.2 Visualizing and Interpreting Data
DAT-2.D.4	Data filtering systems are important tools for finding information and reco	7.4 Array Length and Looping Through Arrays 13.1 Getting Started with Data 13.2 Visualizing and Interpreting Data
DAT-2.D.5	Programs, including spreadsheets, help to efficiently organize and find tre	13.2 Visualizing and Interpreting Data
DAT-2.D.6	Some processes that can be used to extract or modify information from d	7.4 Array Length and Looping Through Arrays 13.2 Visualizing and Interpreting Data 13.3 Data Collection & Limitations
DAT-2.E.1	Programs are used in an iterative and interactive way when processing in	13.2 Visualizing and Interpreting Data
DAT-2.E.2	Programmers can use programs to filter and clean digital data, thereby ga	7.4 Array Length and Looping Through Arrays 13.1 Getting Started with Data
DAT-2.E.3	Combining data sources, clustering data, and classifying data are parts of	13.2 Visualizing and Interpreting Data
DAT-2.E.4	Insight and knowledge can be obtained from translating and transforming	7.4 Array Length and Looping Through Arrays
DAT-2.E.5	Patterns can emerge when data is transformed using programs.	7.4 Array Length and Looping Through Arrays 13.2 Visualizing and Interpreting Data
AAP-1.A.1	A variable is an abstraction inside the program that can hold a value. Eacl	3.4 Variables 7.1 Intro to Lists/Arrays
AAP-1.A.2	Using meaningful variable names helps computer scientists understand v	3.4 Variables
AAP-1.A.3	Some programming languages provide types to represent data, which are	3.4 Variables
AAP-1.A.4	Some values are better suited to representation using one type of data ra	3.4 Variables
AAP-1.B.1	The assignment operator allows a program to change the value represen	3.4 Variables
AAP-1.B.2	The exam reference sheet provides the "←" operator to use for assignme	3.4 Variables
AAP-1.B.3	The value stored in a variable will be the most recent value assigned. For	3.4 Variables
AAP-1.C.1	A list is an ordered sequence of elements. For example, [value1, value2, v	7.1 Intro to Lists/Arrays
AAP-1.C.2	An element is an individual value in a list that is assigned a unique index.	7.1 Intro to Lists/Arrays
AAP-1.C.3	An index is a common method for referencing the elements in a list or str	7.1 Intro to Lists/Arrays
AAP-1.C.4	A string is an ordered sequence of characters.	3.5 User Input
AAP-1.D.1	Data abstraction provides a separation between the abstract properties c	7.3 Adding/Removing From an Array
AAP-1.D.2	Data abstractions manage complexity in the program by giving a collectic	7.4 Array Length and Looping Through Arrays
AAP-1.D.3	Data abstractions can be created using lists.	7.4 Array Length and Looping Through Arrays
AAP-1.D.4	Developing a data abstraction to implement in a program can result in a p	7.4 Array Length and Looping Through Arrays
AAP-1.D.5	Data abstractions often contain different types of elements.	7.3 Adding/Removing From an Array
AAP-1.D.6	The use of lists allows multiple related items to be treated as a single valu	7.1 Intro to Lists/Arrays
AAP-1.D.7	The exam reference sheet provides the notation [value1, value2, value 3..	7.1 Intro to Lists/Arrays
AAP-1.D.8	The exam reference sheet describes a list structure whose index values a	7.1 Intro to Lists/Arrays
AAP-2.A.1	An algorithm is a finite set of instructions that accomplish a specific task.	3.6 Basic Math in JavaScript
AAP-2.A.2	Beyond visual and textual programming languages, algorithms can be ex	3.1 What is Code 3.6 Basic Math in JavaScript
AAP-2.A.3	Algorithms executed by a program are implemented using programming	3.1 What is Code 3.6 Basic Math in JavaScript
AAP-2.A.4	Every algorithm can be constructed using combinations of sequencing, se	1.17 Karel Algorithms 3.6 Basic Math in JavaScript
AAP-2.B.1	Sequencing is the application of each step of an algorithm in the order in	1.17 Karel Algorithms
AAP-2.B.2	A code statement is a part of program code that expresses an action to b	1.17 Karel Algorithms
AAP-2.B.3	Expressions can consist of a value, a variable, operators, or procedure ca	3.6 Basic Math in JavaScript
AAP-2.B.4	Expressions are evaluated to produce a single value.	3.6 Basic Math in JavaScript
AAP-2.B.5	The evaluation of expressions follows a set order of operations defined b	3.6 Basic Math in JavaScript
AAP-2.B.6	Sequential statements execute in the order they appear in the code segn	1.17 Karel Algorithms
AAP-2.B.7	Clarity and readability are important considerations when expressing an a	1.17 Karel Algorithms
AAP-2.C.1	Arithmetic operators are part of most programming languages and includ	3.6 Basic Math in JavaScript
AAP-2.C.2	The exam reference sheet provides a MOD b, which evaluates to the rem	3.6 Basic Math in JavaScript
AAP-2.C.3	The exam reference sheet provides the arithmetic operators +, -, *, /, and	3.6 Basic Math in JavaScript
AAP-2.C.4	The order of operations used in mathematics applies when evaluating ex	3.6 Basic Math in JavaScript
AAP-2.D.1	String concatenation joins together two or more strings end-to-end to ma	3.6 Basic Math in JavaScript
AAP-2.D.2	A substring is part of an existing string.	3.6 Basic Math in JavaScript
AAP-2.E.1	A Boolean value is either true or false.	4.1 Booleans
AAP-2.E.2	The exam reference sheet provides the following relational operators =, ≠	4.3 Comparison Operators
AAP-2.F.1	The exam reference sheet provides the following logic operators: NOT, A	4.3 Comparison Operators
AAP-2.F.2	The exam reference sheet provides NOT condition, which evaluates to tr	4.3 Comparison Operators
AAP-2.F.3	The exam reference sheet provides condition1 AND condition2, which ev	4.3 Comparison Operators
AAP-2.F.4	The exam reference sheet provides condition1 OR condition2, which eval	4.3 Comparison Operators
AAP-2.F.5	The operands for a logic operator are either a Boolean expression or a si	4.3 Comparison Operators
AAP-2.G.1	Selection determines which parts of an algorithm are executed based on	1.13 If/Else Statements 4.4 If Statements
AAP-2.H.1	Conditional statements or "if-statements" affect the sequential flow of cor	4.4 If Statements
AAP-2.H.2	The exam reference sheet provides IF(condition) {<block of statements> }	4.4 If Statements
AAP-2.H.3	The exam reference sheet provides IF(condition) { <first block of statemer	4.4 If Statements
AAP-2.I.1	Nested conditional statements or "else if" statements consist of conditior	4.4 If Statements
AAP-2.I.2	If the Boolean condition of the initial conditional statement evaluates to fa	4.4 If Statements
AAP-2.J.1	Iteration is a repetitive portion of an algorithm. Iteration repeats until a giv	1.14 While Loops in Karel
AAP-2.K.1	Iteration statements change the sequential flow of control by repeating a	1.11 For Loops
AAP-2.K.2	The exam reference sheet provides REPEAT n TIMES { <block of stateme	4.9 While Loops
AAP-2.K.3	The exam reference sheet provides REPEAT UNTIL(condition) { <block of	4.9 While Loops
AAP-2.K.4	In REPEAT UNTIL(condition) iteration, an infinite loop occurs when the en	4.9 While Loops
AAP-2.K.5	In REPEAT UNTIL(condition) iteration, if the conditional evaluates to true i	4.9 While Loops
AAP-2.L.1	Algorithms can be written in different ways and still accomplish the same	4.9 While Loops
AAP-2.L.2	Algorithms that appear similar can yield different side-effects or results.	4.9 While Loops

DAT-2.D.1	Programs can be used to process data to acquire information.
DAT-2.D.2	Tables, diagrams, text, and other visual tools can be used to communicate insight and knowledge gained from data.
DAT-2.D.3	Search tools are useful for efficiently finding information.
DAT-2.D.4	Data filtering systems are important tools for finding information and recognizing patterns in data.
DAT-2.D.5	Programs such as spreadsheets help efficiently organize and find trends in information.
DAT-2.D.6	Some processes that can be used to extract or modify information from data include the following: §§ transforming every element of a data set, such as doubling every element in a list, or adding a parent's email to every student record §§ filtering a data set, such as keeping only the positive numbers from a list, or keeping only students who signed up for band from a record of all the students § combining or comparing data in some way, such as adding up a list of numbers, or finding the student who has the highest GPA §§ visualizing a data set through a chart, graph, or other visual representation
DAT-2.E.1	Programs are used in an iterative and interactive way when processing information to allow users to gain insight and knowledge about data.
DAT-2.E.2	Programmers can use programs to filter and clean digital data, thereby gaining insight and knowledge.
DAT-2.E.3	Combining data sources, clustering data, and classifying data are parts of the process of using programs to gain insight and knowledge from data.
DAT-2.E.4	Insight and knowledge can be obtained from translating and transforming digitally represented information.
DAT-2.E.5	Patterns can emerge when data are transformed using programs.

AAP-2.L.3	Some selections can be written as equivalent Boolean expressions.	4.4 If Statements
AAP-2.L.4	Some Boolean expressions can be written as equivalent selections.	4.4 If Statements
AAP-2.L.5	Different algorithms can be developed or used to solve the same probler	4.9 While Loops
AAP-2.M.1	Algorithms can be created from an idea, by combining existing algorithm:	1.7 Top Down Design and Decomposition in Karel
AAP-2.M.2	Knowledge of existing algorithms can help in constructing algorithms. So	1.17 Karel Algorithms 5.2 Functions and Parameters 2 5.5 Functions and Return Values 2
AAP-2.M.3	Using existing correct algorithms as building blocks for constructing anot	1.10 Super Karel
AAP-2.N.1	The exam reference sheet provides basic operations on lists include: acc	7.1 Intro to Lists/Arrays 7.7 Removing an Element From an Array
AAP-2.N.2	List procedures are implemented in accordance with the syntax rules of t	7.1 Intro to Lists/Arrays 7.2 Indexing Into an Array
AAP-2.O.1	Traversing a list can be a complete traversal where all elements in the list	7.5 Iterating Over an Array 7.7 Removing an Element From an Array
AAP-2.O.2	Iteration statements can be used to traverse a list.	7.5 Iterating Over an Array 7.7 Removing an Element From an Array
AAP-2.O.3	The exam reference sheet provides FOR EACH item IN aList { <block of s	7.7 Removing an Element From an Array
AAP-2.O.4	Knowledge of existing algorithms that use iteration can help in constructi	7.4 Array Length and Looping Through Arrays
AAP-2.O.5	Linear search or sequential search algorithms check each element of a lis	7.6 Finding an Element in a List
AAP-2.P.1	The binary search algorithm starts at the middle of a sorted data set of nu	7.6 Finding an Element in a List
AAP-2.P.2	Data must be in sorted order to use the binary search algorithm.	7.6 Finding an Element in a List
AAP-2.P.3	Binary search is often more efficient than sequential / linear search when	7.6 Finding an Element in a List
AAP-3.A.1	A procedure is a named group of programming instructions that may hav	5.1 Functions and Parameters 1
AAP-3.A.2	Procedures are referred to by different names, such as method or functio	5.1 Functions and Parameters 1
AAP-3.A.3	Parameters are input variables of a procedure. Arguments specify the val	5.1 Functions and Parameters 1
AAP-3.A.4	A procedure call interrupts the sequential execution of statements, causir	5.1 Functions and Parameters 1
AAP-3.A.5	The exam reference sheet provides procName (arg1, arg2, ...) as a way to	7.7 Removing an Element From an Array
AAP-3.A.6	The exam reference sheet provides the procedure DISPLAY(expression) t	3.5 User Input 7.7 Removing an Element From an Array
AAP-3.A.7	The exam reference sheet provides the RETURN(expression) statement, v	7.7 Removing an Element From an Array
AAP-3.A.8	The exam reference sheet provides result ß procName(arg1, arg2, ...) to a	7.7 Removing an Element From an Array
AAP-3.A.9	The exam reference sheet provides procedure INPUT(), which accepts a v	3.5 User Input
AAP-3.B.1	One common type of abstraction is procedural abstraction which provide	1.9 Abstraction
AAP-3.B.2	Procedural abstraction allows a solution to a large problem to be based c	1.9 Abstraction
AAP-3.B.3	The process of subdividing a computer program into separate sub-progra	1.9 Abstraction
AAP-3.B.4	A procedural abstraction may extract shared features to generalize functi	1.9 Abstraction
AAP-3.B.5	Using parameters allows procedures to be generalized, enabling the pro	5.1 Functions and Parameters 1
AAP-3.B.6	Using procedural abstraction helps improve code readability.	1.9 Abstraction
AAP-3.B.7	Using procedural abstraction in a program allows programmers to chang	1.9 Abstraction
AAP-3.C.1	The exam reference sheet provides PROCEDURE procName(parameter1,	5.1 Functions and Parameters 1 7.7 Removing an Element From an Array
AAP-3.C.2	The exam reference sheet provides PROCEDURE procName(parameter1,	5.4 Functions and Return Values 1 7.7 Removing an Element From an Array
AAP-3.D.1	A software library contains procedures that may be used in creating new	1.10 Super Karel 1.19 Ultra Karel
AAP-3.D.2	Existing code segments can come from internal or external sources, such	1.10 Super Karel 1.19 Ultra Karel
AAP-3.D.3	The use of libraries simplifies the task of creating complex programs.	1.10 Super Karel 1.19 Ultra Karel
AAP-3.D.4	Application program interfaces (APIs) are specifications for how the proce	1.10 Super Karel 1.19 Ultra Karel
AAP-3.D.5	Documentation for an API/library is necessary in understanding the behav	1.10 Super Karel 1.19 Ultra Karel
AAP-3.E.1	The exam reference sheet provides RANDOM(a, b) which generates and	7.7 Removing an Element From an Array
AAP-3.E.2	Using random number generation in a program means each execution cc	4.8 Random Numbers
AAP-3.F.1	Simulations are abstractions of more complex objects or phenomena for	7.8 Simulation
AAP-3.F.2	A simulation is a representation that uses varying sets of values to reflect	7.8 Simulation
AAP-3.F.3	Simulations often mimic real-world events with the purpose of drawing in	7.8 Simulation
AAP-3.F.4	The process of developing an abstract simulation involves removing spec	7.8 Simulation
AAP-3.F.5	Simulations can contain bias derived from the choices of elements of the	7.8 Simulation
AAP-3.F.6	Simulations are most useful when real-world events are impractical for ex	7.8 Simulation
AAP-3.F.7	Simulations facilitate the formulation and refinement of hypotheses relate	7.8 Simulation
AAP-3.F.8	Random number generators can be used to simulate the variability that e	7.8 Simulation
AAP-4.A.1	A problem is a general description of a task that may (or may not) be solv	7.6 Finding an Element in a List
AAP-4.A.2	A decision problem is a problem with a yes-no answer. An optimization pi	1.17 Karel Algorithms
AAP-4.A.3	Efficiency measures the number of steps an algorithm requires before it t	7.6 Finding an Element in a List
AAP-4.A.4	Determining an algorithm's efficiency is done by reasoning formally or ma	1.17 Karel Algorithms
AAP-4.A.5	An algorithm's efficiency can be informally measured by determining the	1.17 Karel Algorithms
AAP-4.A.6	Different correct algorithms for the same problem can have different effic	1.17 Karel Algorithms
AAP-4.A.7	Algorithms with efficiencies that grow at a polynomial rate or slower (con:	7.6 Finding an Element in a List
AAP-4.A.8	Some problems cannot be solved in a reasonable amount of time becaus	7.6 Finding an Element in a List
AAP-4.A.9	A heuristic is an approach to a problem that produces a solution that is nc	7.6 Finding an Element in a List
AAP-4.B.1	A decidable problem is a decision problem for which an algorithm can be	8.10 Cryptography
AAP-4.B.2	An undecidable problem is one in which no algorithm can be constructed	8.10 Cryptography
AAP-4.B.3	An undecidable problem may have some instances that have an algorithr	8.10 Cryptography
CSN-1.A.1	A computing device is a physical artifact that can run a program. Some ex	11.2 Internet Hardware
CSN-1.A.2	A computing system is a group of computing devices and programs work	11.2 Internet Hardware
CSN-1.A.3	A computer network is a group of interconnected computing devices cap	11.2 Internet Hardware
CSN-1.A.4	A computer network is a type of a computing system.	11.2 Internet Hardware
CSN-1.A.5	A path between two computing devices on a computer network (a sende	11.6 Routing
CSN-1.A.6	Routing is the process of finding a path from sender to receiver.	11.6 Routing
CSN-1.A.7	The bandwidth of a computer network is the maximum amount of data th	11.2 Internet Hardware
CSN-1.A.8	Bandwidth is usually measured in bits per second.	11.2 Internet Hardware
CSN-1.B.1	The Internet is a computer network consisting of interconnected network	11.7 Packets and Protocols
CSN-1.B.2	Access to the Internet depends on the ability to connect a computing dev	11.7 Packets and Protocols
CSN-1.B.3	A protocol is an agreed-upon set of rules that specify the behavior of som	11.1 Welcome to the Internet
CSN-1.B.4	The protocols used in the Internet are open which allows users to easily c	11.3 Internet Addresses
CSN-1.B.5	Routing on the Internet is usually dynamic; it is not specified in advance.	11.6 Routing
CSN-1.B.6	Scalability of a system is the capacity for the system to change in size anc	11.6 Routing
CSN-1.B.7	The Internet was designed to be scalable.	11.6 Routing
CSN-1.C.1	Information is passed through the Internet as a datastream. Datastreams	11.7 Packets and Protocols
CSN-1.C.2	Packets contain a chunk of data and metadata used for routing a packet l	11.7 Packets and Protocols
CSN-1.C.3	Packets may arrive at the destination in order, out-of-order, or not at all.	11.7 Packets and Protocols
CSN-1.C.4	IP, TCP, and UDP are common protocols used on the Internet.	11.7 Packets and Protocols
CSN-1.D.1	The world wide web is a system of linked pages, programs, and files.	11.7 Packets and Protocols
CSN-1.D.2	The HTTP protocol is the used on the World Wide Web.	11.7 Packets and Protocols
CSN-1.D.3	The World Wide Web uses the Internet.	11.7 Packets and Protocols
CSN-1.E.1	The Internet has been engineered to be fault-tolerant, with abstractions f	11.6 Routing
CSN-1.E.2	Redundancy is the inclusion of extra components that can be used to mit	11.6 Routing
CSN-1.E.3	One way redundancy is accomplished in networks is by having more thar	11.6 Routing
CSN-1.E.4	If a particular device or connection on the Internet fails, subsequent data	11.6 Routing
CSN-1.E.5	When a system can support failures and still continue to function, it is call	11.6 Routing
CSN-1.E.6	Redundancy within a system often requires additional resources but can	11.6 Routing

CSN-1.E.7	The redundancy of routing between two points on the Internet increases	11.6 Routing
CSN-2.A.1	Sequential computing is a computational model in which operations are p	11.8 Sequential, Parallel & Distributed Computing
CSN-2.A.2	Parallel computing is a computational model where the program is broke	11.8 Sequential, Parallel & Distributed Computing
CSN-2.A.3	Distributed computing is a computational model in which multiple device	11.8 Sequential, Parallel & Distributed Computing
CSN-2.A.4	Comparing efficiency of solutions can be done by comparing the time it t	11.8 Sequential, Parallel & Distributed Computing
CSN-2.A.5	A sequential solution takes as long as the sum of all of its steps.	11.8 Sequential, Parallel & Distributed Computing
CSN-2.A.6	A parallel computing solution takes as long as its sequential tasks plus th	11.8 Sequential, Parallel & Distributed Computing
CSN-2.A.7	The “speedup” of a parallel solution is measured in the time it took to cor	11.8 Sequential, Parallel & Distributed Computing
CSN-2.B.1	Parallel computing consists of a parallel portion and a sequential portion.	11.8 Sequential, Parallel & Distributed Computing
CSN-2.B.2	Solutions that use parallel computing can scale more effectively than solu	11.8 Sequential, Parallel & Distributed Computing
CSN-2.B.3	Distributed computing allows problems to be solved that could not be sol	11.8 Sequential, Parallel & Distributed Computing
CSN-2.B.4	Distributed computing allows much larger problems to be solved quicker	11.8 Sequential, Parallel & Distributed Computing
CSN-2.B.5	When increasing the use of parallel computing in a solution, the efficiency	11.8 Sequential, Parallel & Distributed Computing
IOC-1.A.1	People are creators of computing innovations.	11.10 The Impact of the Internet
IOC-1.A.2	As computing evolves, the way people complete tasks often changes to i	18.1 Intro to Design Thinking
IOC-1.A.3	The total effects of a computing innovation are not always anticipated in a	11.10 The Impact of the Internet
IOC-1.A.4	A single effect can be viewed as both beneficial and harmful based on ar	11.10 The Impact of the Internet
IOC-1.A.5	Advances in computing have generated and increased creativity in other	11.10 The Impact of the Internet
IOC-1.B.1	Computing innovations can be used in ways that the creator had not orig	11.10 The Impact of the Internet
IOC-1.B.2	Some of the unintended ways computing innovations can be used may h	11.10 The Impact of the Internet
IOC-1.B.3	Responsible programmers try to consider the unintended ways their com	11.10 The Impact of the Internet
IOC-1.B.4	It is not possible for a programmer to consider all the ways a computing i	11.10 The Impact of the Internet
IOC-1.B.5	Often computing innovations have had a beneficial effect by leading to a	11.10 The Impact of the Internet
IOC-1.B.6	Rapid sharing of the program or the results of running a program with a la	11.10 The Impact of the Internet
IOC-1.C.1	Internet access varies between socioeconomic, geographic, or demograp	11.10 The Impact of the Internet
IOC-1.C.2	The digital divide refers to differing access to computing devices and the	11.10 The Impact of the Internet
IOC-1.C.3	The digital divide can affect both groups and individuals.	11.10 The Impact of the Internet
IOC-1.C.4	The digital divide raises issues of equity, access, and influence, both glob	11.10 The Impact of the Internet
IOC-1.C.5	The digital divide is affected by individuals, organizations and governmer	11.10 The Impact of the Internet
IOC-1.D.1	Computing innovations can reflect existing human biases because of bias	18.3 Test
IOC-1.D.2	Programmers should take action to reduce bias in algorithms used for co	18.3 Test
IOC-1.D.3	Biases can be embedded at all levels of software development.	18.3 Test
IOC-1.E.1	Widespread access to information and public data facilitates the identifica	11.10 The Impact of the Internet
IOC-1.E.2	Science has been impacted by using scale and “citizen science” to solve	11.10 The Impact of the Internet
IOC-1.E.3	Citizen science is scientific research conducted in whole or part by indivi	11.10 The Impact of the Internet
IOC-1.E.4	Crowdsourcing is the practice of obtaining input or information from a lar	11.10 The Impact of the Internet
IOC-1.E.5	Human capabilities can be enhanced by collaboration via computing.	11.10 The Impact of the Internet
IOC-1.E.6	Crowdsourcing offers new models for collaboration, such as connecting p	11.10 The Impact of the Internet
IOC-1.F.1	Material created on a computer is the intellectual property of the creator	11.10 The Impact of the Internet
IOC-1.F.2	Ease of access and distribution of digitized information raises intellectual	11.10 The Impact of the Internet
IOC-1.F.3	Measures should be taken to safeguard intellectual property.	11.11 Creative Credit & Copyright
IOC-1.F.4	The use of material created by someone else without permission is plagia	11.10 The Impact of the Internet 11.11 Creative Credit & Copyright
IOC-1.F.5	Some examples of legal ways to use materials created by someone else	11.10 The Impact of the Internet 11.11 Creative Credit & Copyright
IOC-1.F.6	The use of material created by someone other than yourself should alway	11.11 Creative Credit & Copyright
IOC-1.F.7	Creative commons, open source, and open access have enabled broad a	11.10 The Impact of the Internet
IOC-1.F.8	Using computing to harm individuals or groups of people raise legal and	11.9 Cybersecurity
IOC-1.F.9	Computing can play a role in social and political issues which in turn ofter	11.10 The Impact of the Internet
IOC-1.F.10	The digital divide raises ethical concerns around computing.	11.10 The Impact of the Internet
IOC-1.F.11	Computing innovations can raise legal and ethical concerns. Some examp	11.11 Creative Credit & Copyright 15.1 The Impacts of Computing 18.3 Test
IOC-2.A.1	Personally identifiable information (PII) is information about an individual t	11.9 Cybersecurity
IOC-2.A.2	Search engines can record and maintain a history of searches made by u	15.1 The Impacts of Computing
IOC-2.A.3	Websites can record and maintain a history of individuals who have viewe	15.1 The Impacts of Computing
IOC-2.A.4	Devices, websites, and networks can collect information about a user’s lo	15.1 The Impacts of Computing
IOC-2.A.5	Technology enables the collection, use, and exploitation of information ab	15.1 The Impacts of Computing
IOC-2.A.6	Search engines can use search history to suggest websites or for target r	15.1 The Impacts of Computing
IOC-2.A.7	Disparate personal data, such as geolocation, cookies, and browsing hist	11.9 Cybersecurity
IOC-2.A.8	PII and other information placed online can be used to enhance a user’s	11.9 Cybersecurity
IOC-2.A.9	PII stored online can be used to simplify making online purchases.	11.9 Cybersecurity
IOC-2.A.10	Commercial and governmental curation of information may be exploited i	15.1 The Impacts of Computing
IOC-2.A.11	Information placed online can be used in ways that were not intended an	11.9 Cybersecurity
IOC-2.A.12	PII can be used to stalk or steal the identity of a person, or to aid in the pl	11.9 Cybersecurity
IOC-2.A.13	It is difficult to delete information once it has been placed online.	11.9 Cybersecurity
IOC-2.A.14	Applications can collect your location and record where you have been, h	15.1 The Impacts of Computing
IOC-2.A.15	Information posted to social media services can be used by others. Comb	11.9 Cybersecurity
IOC-2.B.1	Authentication measures protect devices and information from unauthori	11.9 Cybersecurity
IOC-2.B.2	A strong password is something that is easy for a user to remember but v	11.9 Cybersecurity
IOC-2.B.3	Multi-factor authentication is a method of computer access control in whi	11.9 Cybersecurity
IOC-2.B.4	Multi-factor authentication requires at least two steps to unlock protected	11.9 Cybersecurity
IOC-2.B.5	Encryption is the process of encoding data to prevent unauthorized acce	8.10 Cryptography 11.9 Cybersecurity
IOC-2.B.6	Certificate authorities (CAs) issue digital certificates that validate the own	11.9 Cybersecurity
IOC-2.B.7	Computer virus and malware scanning software can help to protect a cor	11.9 Cybersecurity
IOC-2.B.8	A computer virus is a malicious program that can copy itself and gain acc	8.10 Cryptography
IOC-2.B.9	Malware is software intended to damage a computing system or to take p	11.9 Cybersecurity
IOC-2.B.10	All real-world systems have errors or design flaws that can be exploited t	11.9 Cybersecurity
IOC-2.B.11	Users can control the permissions applications have for collecting user in	11.9 Cybersecurity
IOC-2.C.1	Phishing is a technique that is used to trick a user into providing personal	11.9 Cybersecurity
IOC-2.C.2	Keylogging is the use of a program to record every keystroke made by a	11.9 Cybersecurity
IOC-2.C.3	Data sent over public networks can be intercepted, analyzed and modifie	11.9 Cybersecurity
IOC-2.C.4	A rogue access point is a wireless access point that gives unauthorized a	11.9 Cybersecurity
IOC-2.C.5	A malicious link can be disguised on a web page or in an email message.	11.9 Cybersecurity
IOC-2.C.6	Unsolicited emails, attachments, links, and forms in emails can be used tc	11.9 Cybersecurity
IOC-2.C.7	Untrustworthy (often free) downloads from freeware or shareware sites can contain malware.	