AP CSP Exam Preparation

This packet contains the following:

AP Computer Science Principles End-of-Course Exam

Weight: 60% of the AP Computer Science Principles final score

Hours: 2 hours

Date: May (in the AP Exam administration window)

Note: The end-of-course exam will be administered using the same procedures and guidelines as all other AP Exams.

Overview

The AP Computer Science Principles Exam is composed of two types of multiple-choice questions:

- **Single-select multiple-choice questions:** Students select one answer from among four options.
- **Multiple-select multiple-choice questions:** Students select two answers from among four options.

Multiple-choice questions on the exam are classified according to learning objectives within each big idea in the AP Computer Science Principles curriculum framework. Some exam questions may be aligned to more than one learning objective. For example, a question on programming might implement an algorithm and contain abstractions. In any case, a primary learning objective is identified and is used to ensure the appropriate distribution of test questions in the AP Exam. The table below intends to show the approximate percentages of test question per big idea. Teachers should examine this table as one of many other important features of the course to plan their instruction.

Big Ideas	Approximate Percentage of Multiple-Choice Questions
Big Idea 1: Creativity	---
Big Idea 2: Abstraction	19%
Big Idea 3: Data and Information	18%
Big Idea 4: Algorithms	20%
Big Idea 5: Programming	20%
Big Idea 6: The Internet	13%
Big Idea 7: Global Impact	10%

The AP Computer Science Principles Exam Reference Sheet is found on page 114 and provides programming instructions and explanations to help students understand questions they will see on the AP Exam.

© 2016 The College Board

Vocabulary

UNIT 1 - THE INTERNET	
ASCII	ASCII - American Standard Code for Information Interchange. ASCII is the universally recognized raw text format that any computer can understand.
Bandwidth	Transmission capacity measure by bit rate
Binary	A way of representing information using only two options.
Bit	A contraction of "Binary Digit". A bit is the single unit of information in a computer, typically represented as a 0 or 1.
Bit rate or bitrate	the number of bits that are conveyed or processed per unit of time. e.g. 8 bits/sec.
DNS	The service that translates URLs to IP addresses.
HTTP	HyperText Transfer Protocol - the protocol used for transmitting web pages over the Internet
IETF	Internet Engineering Task Force - develops and promotes voluntary Internet standards and protocols, in particular the standards that comprise the Internet protocol suite (TCP/IP).
Innovation	A novel or improved idea, device, product, etc, or the development thereof.
Internet	A group of computers and servers that are connected to each other.
IP Address	A number assigned to any item that is connected to the Internet.
Latency	Time it takes for a bit to travel from its sender to its receiver.
Net Neutrality	the principle that all Internet traffic should be treated equally by Internet Service Providers.
Packets	Small chunks of information that have been carefully formed from larger chunks of information.
Phishing Attack	the attempt to obtain sensitive information such as usernames, passwords, and credit card details (and, indirectly, money), often for malicious reasons, by disguising as a trustworthy entity in an electronic communication.
Protocol	A set of rules governing the exchange or transmission of data between devices.
TCP	Transmission Control Protocol - provides reliable, ordered, and error-checked delivery of a stream of packets on the internet. TCP is tightly linked with IP and usually seen as TCP/IP in writing.
URL	An easy-to-remember address for calling a web page (like www.code.org).

Vocabulary

UNIT 2 - DIGITAL INFORMATION	
Abstraction	Pulling out specific differences to make one solution work for multiple problems.
Aggregation	a computation in which rows from a data set are grouped together and used to compute a single value of more significant meaning or measurement. Common aggregations include
Heuristic	a problem solving approach (algorithm) to find a satisfactory solution where finding an optimal or exact solution is impractical or impossible.
Image	A type of data used for graphics or pictures.
Lossless Compression	a data compression algorithm that allows the original data to be perfectly reconstructed from the compressed data.
Lossy Compression	(or irreversible compression) a data compression method that uses inexact approximations, discarding some data to represent the content. Most commonly seen in image formats like .jpg.
metadata	is data that describes other data. For example, a digital image my include metadata that describe the size of the image, number of colors, or resolution.
Pivot Table	in most spreadsheet software it is the name of the tool used to create summary tables.
pixel	short for "picture element" it is the fundamental unit of a digital image, typically a tiny square or dot which contains a single point of color of a larger image.
RGB	the RGB color model uses varying intensities of (R)ed, (G)reen, and (B)lue light are added together in to reproduce a broad array of colors.
Summary Table	a table that shows the results of aggregations performed on data from a larger data set, hence a "summary" of larger data. Spreadsheet software typically calls them "pivot tables".

Vocabulary

UNIT 3 - ALGORITHMS AND PROGRAMMING	
Abstraction	Pulling out specific differences to make one solution work for multiple problems.
Algorithm	A precise sequence of instructions for processes that can be executed by a computer
API	a collection of commands made available to a programmer
Documentation	a description of the behavior of a command, function, library, API, etc.
Function	A piece of code that you can easily call over and over again.
Library	a collection of commands / functions, typically with a shared purpose
Loop	The action of doing something over and over again.
Parameter	An extra piece of information that you pass to the function to customize it for a specific need.

Vocabulary

UNIT 4 - BIG DATA AND PRIVACY	
asymmetric encryption	used in public key encryption, it is scheme in which the key to encrypt data is different from the key to decrypt.
Cipher	the generic term for a technique (or algorithm) that performs encryption
Computationally Hard	a "hard' problem for a computer is one in which it cannot arrive at a solution in a reasonable amount of time.
Cracking encryption	When you attempt to decode a secret message without knowing all the specifics of the cipher, you are trying to "crack" the encryption.
Decryption	a process that reverses encryption, taking a secret message and reproducing the original plain text
Digital Certificate	Used to verify that a user sending a message is who he or she claims to be, and to provide the receiver with the means to encode a reply
Distributed Denial of Service Attack (DDoS)	occurs when multiple compromised systems flood the bandwidth or resources of a targeted system, usually one or more web servers.
Encryption	a process of encoding messages to keep them secret, so only "authorized" parties can read it.
Modulo or mod	a mathematical operation that returns the remainder after integer division. Example
Moore's Law	a predication made by Gordon Moore in 1965 that computing power will double every 1.5-2 years, it has remained more or less true ever since.
Private Key	In an asymmetric encryption scheme the decryption key is kept private and never shared, so only the intended recipient has the ability to decrypt a message that has been encrypted with a public key.
Public Key Encryption	Used prevalently on the web, it allows for secure messages to be sent between parties without having to agree on, or share, a secret key. It uses an asymmetric encryption scheme in which the encryption key is made public, but the decryption key is kept private.

Vocabulary

UNIT 5 - BUILDING APPS	
Boolean	A single value of either TRUE or FALSE
Boolean Expression	in programming, an expression that evaluates to True or False.
Concatenate	to link together or join. Typically used when joining together text Strings in programming (e.g. "Hello, "+name)
Conditionals	Statements that only run under certain conditions.
Data Type	All values in a programming language have a "type" - such as a Number, Boolean, or String - that dictates how the computer will interpret it. For example 7+5 is interpreted differently from "7"+"5"
Debugging	Finding and fixing problems in your algorithm or program.
Expression	Any valid unit of code that resolves to a value.
Global Variable	A variable whose scope is "global" to the program, it can be used and updated by any part of the code. Its global scope is typically derived from the variable being declared (created) outside of any function, object, or method.
Iteration	To repeat in order to achieve, or get closer to, a desired goal.
List	A generic term for a programming data structure that holds multiple items.
Local Variable	A variable with local scope is one that can only be seen, used and updated by code within the same scope. Typically this means the variable was declared (created) inside a function -- includes function parameter variables.
Models and Simulations	a program which replicates or mimics key features of a real world event in order to investigate its behavior without the cost, time, or danger of running an experiment in real life.
Return Value	A value sent back by a function to the place in the code where the function was called form - typically asking for value or the result of a calculation or computation of some kind.
Selection	A generic term for a type of programming statement (usually an if-statement) that uses a Boolean condition to determine, or select, whether or not to run a certain block of statements.
String	Any sequence of characters between quotation marks
UI Elements	on-screen objects, like buttons, images, text boxes, pull down menus, screens and so on.
User Interface	The visual elements of a program through which a user controls or communications the application. Often abbreviated UI.
Variable	A placeholder for a piece of information that can change.
Variable Scope	dictates what portions of the code can "see" or use a variable, typically derived from where the variable was first created. (See Global v. Local)

AP Computer Science Principles Exam Reference Sheet

As AP Computer Science Principles does not designate any particular programming language, this reference sheet provides instructions and explanations to help students understand the format and meaning of the questions they will see on the exam. The reference sheet includes two programming formats: text based and block based.

Programming instructions use four data types: numbers, Booleans, strings, and lists.

Instructions from any of the following categories may appear on the exam:

- Assignment, Display, and Input
- Arithmetic Operators and Numeric Procedures
- Relational and Boolean Operators
- Selection
- Iteration
- List Operations
- Procedures
- Robot

Instruction	Explanation
Assignment, Display, and Input	
Text: `a ← expression` Block: `a ← expression`	Evaluates `expression` and assigns the result to the variable `a`.

Text:
`DISPLAY (expression)`
 	
Block:
`DISPLAY` `expression`
	Displays the value of `expression`, followed by a space.

Text:
`INPUT ()`
 	
Block:
`INPUT`
	Accepts a value from the user and returns it.

Arithmetic Operators and Numeric Procedures	

Text and Block:
`a + b`
`a - b`
`a * b`
`a / b`
	The arithmetic operators `+`, `–`, `*`, and `/` are used to perform arithmetic on `a` and `b`.
	
	For example, `3 / 2` evaluates to `1.5`.

Text and Block: `a MOD b`	Evaluates to the remainder when `a` is divided by `b`. Assume that `a` and `b` are positive integers. For example, `17 MOD 5` evaluates to `2`.

Text:
`RANDOM (a, b)`
 	
Block:
`RANDOM` `a, b`
	Evaluates to a random integer from `a` to `b`, including `a` and `b`.
	
	For example, `RANDOM (1, 3)` could evaluate to `1`, `2`, or `3`.

Relational and Boolean Operators	

Text and Block:
`a = b`
`a ≠ b`
`a > b`
`a < b`
`a ≥ b`
`a ≤ b`
	The relational operators `=`, `≠`, `>`, `<`, `≥`, and `≤` are used to test the relationship between two variables, expressions, or values.
	
	For example, `a = b` evaluates to `true` if `a` and `b` are equal; otherwise, it evaluates to `false`.

Instruction	Explanation
Relational and Boolean Operators (continued)	

Text:
`NOT condition`
 	
Block:
`NOT` `condition`
	Evaluates to `true` if condition is `false`; otherwise evaluates to `false`.

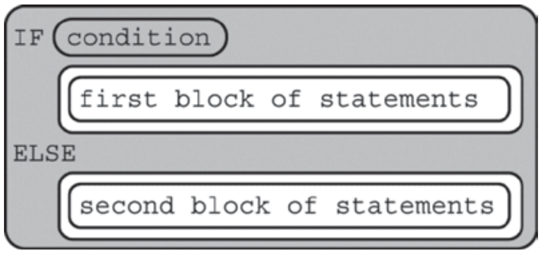
Text:
`condition1 AND condition2`
 	
Block:
`condition1` `AND` `condition2`
	Evaluates to `true` if both `condition1` and `condition2` are `true`; otherwise, evaluates to `false`.

Text:
`condition1 OR condition2`
 	
Block:
`condition1` `OR` `condition2`
	Evaluates to `true` if `condition1` is `true` or if `condition2` is `true` or if both `condition1` and `condition2` are `true`; otherwise, evaluates to `false`.

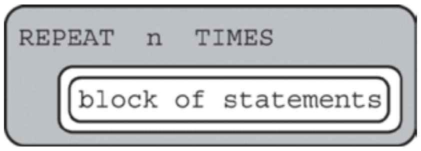
Selection	

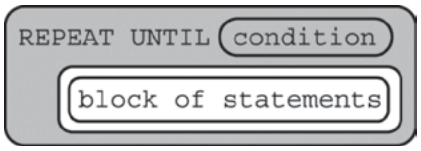
Text:
`IF (condition)`
`{`
`   <block of statements>`
`}`
 	
Block:
`IF` `condition`
`   block of statements`
	The code in `block of statements` is executed if the Boolean expression `condition` evaluates to `true`; no action is taken if `condition` evaluates to `false`.

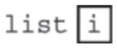
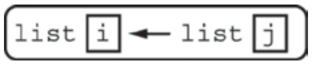
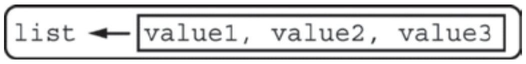
Instruction	Explanation
Selection (continued)	

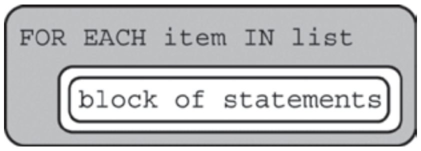
Text:
`IF (condition)`
`{`
`   <first block of statements>`
`}`
`ELSE`
`{`
`   <second block of statements>`
`}`
Block:
 IF
condition
first block of statements
ELSE
second block of statements
	The code in `first block of statements` is executed if the Boolean expression `condition` evaluates to `true`; otherwise, the code in `second block of statements` is executed.

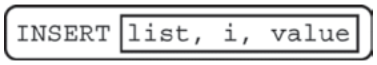
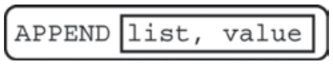
Iteration	

Text:
`REPEAT n TIMES`
`{`
`   <block of statements>`
`}`
Block:

	The code in `block of statements` is executed `n` times.

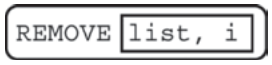
Text:
`REPEAT UNTIL (condition)`
`{`
`   <block of statements>`
`}`
Block:

	The code in `block of statements` is repeated until the Boolean expression `condition` evaluates to `true`.

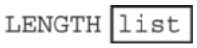
Instruction	Explanation
List Operations	
For all list operations, if a list index is less than 1 or greater than the length of the list, an error message is produced and the program terminates.	
Text: `list[i]` Block: 	Refers to the element of `list` at index `i`. The first element of `list` is at index `1`.
Text: `list[i] ← list[j]` Block: 	Assigns the value of `list[j]` to `list[i]`.
Text: `list ← [value1, value2, value3]` Block: 	Assigns `value1`, `value2`, and `value3` to `list[1]`, `list[2]`, and `list[3]`, respectively.

Text:
`FOR EACH item IN list`
`{`
`   <block of statements>`
`}`
 	
Block:

	The variable `item` is assigned the value of each element of `list` sequentially, in order from the first element to the last element.
	The code in `block of statements` is executed once for each assignment of `item`.

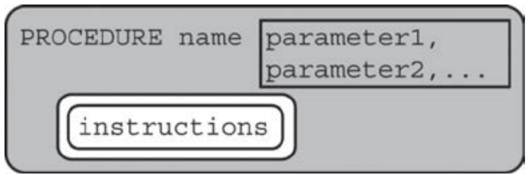
Text: `INSERT (list, i, value)` Block: 	Any values in `list` at indices greater than or equal to `i` are shifted to the right. The length of list is increased by 1, and `value` is placed at index `i` in `list`.
Text: `APPEND (list, value)` Block: 	The length of `list` is increased by 1, and `value` is placed at the end of `list`.

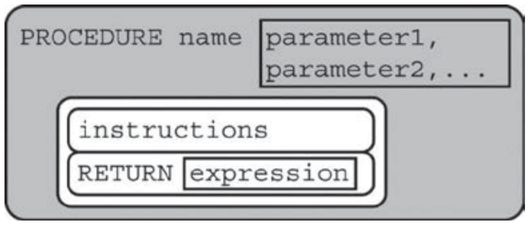
Instruction	Explanation
List Operations (continued)	

Text:
`REMOVE (list, i)`
Block:

`REMOVE list, i`
	Removes the item at index `i` in `list` and shifts to the left any values at indices greater than `i`. The length of `list` is decreased by 1.

Text:
`LENGTH (list)`
Block:

`LENGTH list`
	Evaluates to the number of elements in `list`.

Procedures	

Text:
`PROCEDURE name (parameter1,`
`                parameter2, ...)`
`{`
`    <instructions>`
`}`
Block:

`PROCEDURE name parameter1, parameter2,...`
`instructions`
	A procedure, `name`, takes zero or more parameters. The procedure contains programming instructions.

Text:
`PROCEDURE name (parameter1,`
`                parameter2, ...)`
`{`
`    <instructions>`
`    RETURN (expression)`
`}`
Block:

`PROCEDURE name parameter1, parameter2,...`
`instructions`
`RETURN expression`
	A procedure, `name`, takes zero or more parameters. The procedure contains programming instructions and returns the value of `expression`. The `RETURN` statement may appear at any point inside the procedure and causes an immediate return from the procedure back to the calling program.

Instruction	Explanation
Robot	
If the robot attempts to move to a square that is not open or is beyond the edge of the grid, the robot will stay in its current location and the program will terminate.	

Text:
`MOVE_FORWARD ()`
Block:
`MOVE_FORWARD`
	The robot moves one square forward in the direction it is facing.

Text:
`ROTATE_LEFT ()`
Block:
`ROTATE_LEFT`
	The robot rotates in place 90 degrees counterclockwise (i.e., makes an in-place left turn).

Text:
`ROTATE_RIGHT ()`
Block:
`ROTATE_RIGHT`
	The robot rotates in place 90 degrees clockwise (i.e., makes an in-place right turn).

Text:
`CAN_MOVE (direction)`
Block:
`CAN_MOVE` `direction`
	Evaluates to `true` if there is an open square one square in the direction relative to where the robot is facing; otherwise evaluates to `false`. The value of direction can be `left`, `right`, `forward`, or `backward`.

Important Differences Between the JavaScript and the Reference Language

There are some differences between JavaScript and the Reference Language. Some of the key differences are:

Type of Operation	JavaScript	Reference Language

Assignment Operator	`var a = expression;`
	`a = expression`
		Text:
		`a ← expression`
		Block:
		`a ← expression`

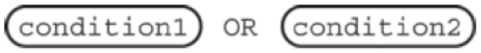
Input	`getText("textBox");`
	`promptNum("Enter a value");`
	`prompt("Enter a value");`
		Text:
		`INPUT ()`
		Block:
		`INPUT`

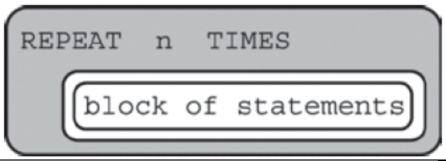
Output	`write(expression);`
	`console.log(expression);`
		Text:
		`DISPLAY (expression)`
		Block:
		`DISPLAY expression`

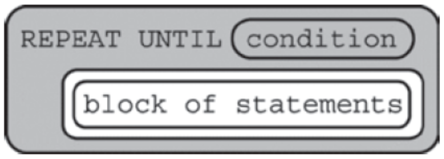
Conditional Operators	`a == b`
	`a != b`
	`a > b`
	`a < b`
	`a >= b`
	`a <= b`
		Text and Block:
		`a = b`
		`a ≠ b`
		`a > b`
		`a < b`
		`a ≥ b`
		`a ≤ b`

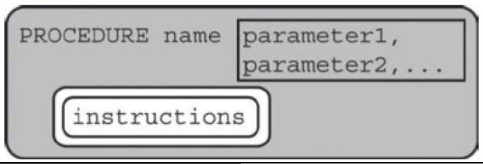
Boolean Expressions	`!condition`
	
	`condition1 && condition2`
		Text:
		`NOT condition`
		Block:
		`NOT condition`
		Text:
		`condition1 AND condition2`
		Block:
		`condition1 AND condition2`

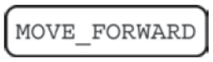
Important Differences Between the JavaScript and the Reference Language

	`condition1 || condition2`	Text: `condition1 OR condition2` Block: `condition1` OR `condition2`

Repeat Loop	`for (var i = 0; i < n; i++)`
	`{`
	
	`}`
		Text:
		`REPEAT n TIMES`
		`{`
		`   <block of statements>`
		`}`
		Block:
		REPEAT n TIMES
		block of statements

While Loop	`while (condition) {`
	
	`}`
	
	It is important to note that these loops work in the opposite ways! In JavaScript the loop continues ***while the condition is true***. In the reference language the loop continues ***while the condition is false***.
		Text:
		`REPEAT UNTIL (condition)`
		`{`
		`   <block of statements>`
		`}`
		Block:
		REPEAT UNTIL condition
		block of statements

Procedure/ Function	`function name(parameter1, parameter2, ...) {`
	`  <instructions>`
	`}`
		Text:
		`PROCEDURE name (parameter1,`
		`                parameter2, ...)`
		`{`
		`    <instructions>`
		`}`
		Block:
		PROCEDURE name parameter1, parameter2,...
		instructions

Turtle Moves	`moveForward();`	Text: `MOVE_FORWARD ()` Block: MOVE_FORWARD

Important Differences Between the JavaScript and the Reference Language

	`turnLeft(90);`	Text: `ROTATE_LEFT ()` Block: ROTATE_LEFT
	`turnRight(90);`	Text: `ROTATE_RIGHT ()` Block: ROTATE_RIGHT
	There is no such function in App Lab. CAN_MOVE returns true if the turtle can move in the direction requested (i.e. the square in the direction given is white and not off the grid)	Text: `CAN_MOVE (direction)` Block: CAN_MOVE direction
Lists – See the section on lists.		

Practicing with the Reference Language

1. What is the output of the following program if the user inputs 7?

```
total ← 0
number ← INPUT()
REPEAT number TIMES
{
    total ← total + 5
}
DISPLAY (total)
```

2. Write this JavaScript code in the Reference Language

```
var a = 2;
var b = 5;
var c = 3;
var maxValue;
if (a > b && a > c) {
    maxValue = a;
} else {
    if (b > a && b > c) {
        maxValue = b;
    } else {
        maxValue = c;
    }
}
write(maxValue);
```

3. Write this Reference Language code in JavaScript:

```
i ← 0
sum ← 0
REPEAT UNTIL (i=4)
{
    sum ← sum + i
    i ← i + 1
}
DISPLAY (sum)
```

Practicing with the Reference Language

1. The output is 35. The loop will execute 5 times and each time add 5 to `total`

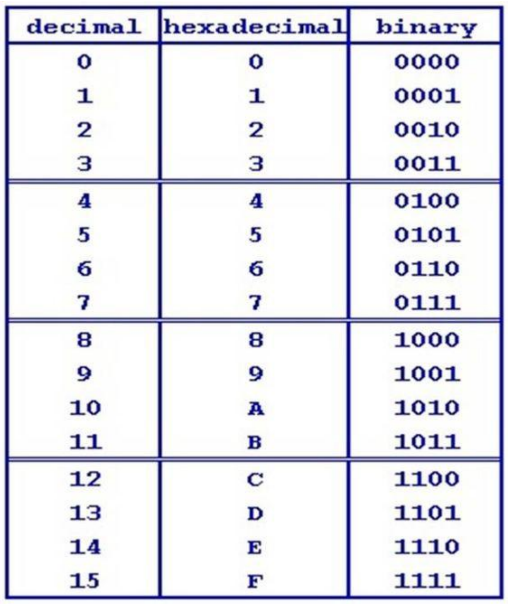
2. The program is:

```
a ← 2
b ← 5
c ← 3
IF ((a > b) AND (a > c))
{
    maxValue = a;
}
ELSE
{
    IF (b > a) AND (b > c)
    {
        maxValue = b;
    }
    ELSE
    {
        maxValue = c;
    }
}
DISPLAY(maxValue);
```

3. The program is:

```
var i = 0;
var sum = 0;
while ((i != 4)) {
    sum = sum+i;
    i = i+1;
}
write(maxValue);
```

Decimal-Binary-Hex Conversion



decimal	hexadecimal	binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Binary Numbers:

Start at 1 and multiple by 2 every time you move left. **Know this to at least 8 Binary digits!**

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Review how to convert from binary to decimal and from decimal to binary. Create the first line of table above and think “Flippy do”!

Always remember that if you add a bit you can represent twice as many numbers. If you add 4 bits, then you can represent 2*2*2*2 or 2^4 as many numbers. If you go from a 32 bit number to a 64 bit number you can represent 2^{32} more numbers not 32 more numbers.

Decimal-Binary-Hex Conversion

<u>Hex Numbers:</u>

Hex is base 16. The digits go from 0-9 and then A-F. See the table above. **<u>Know</u>** that A=10, B=11, C=12, D=13, E=14 and F=15.

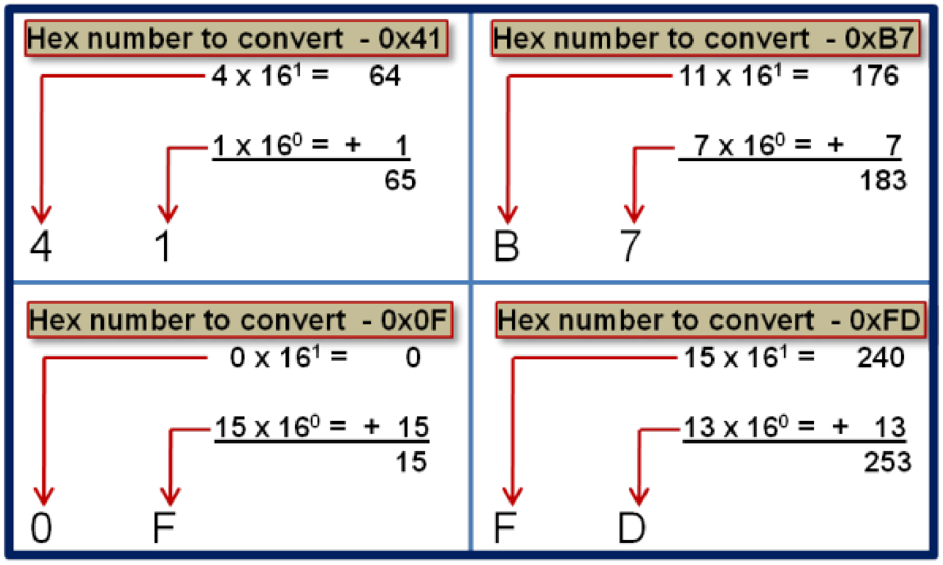
<u>You will need to know how to convert 2 digit Hex numbers to decimal</u>. Simply take the second digit (left) * 16 and then add the first digit (right).

Hex 21 is (2*16)+1 = 33

Hex AA = (10*16)+10= 170

Hex is often written as 0x and then the number but I don't see that that used on the sample exams.

Examples:



Can you do it? Answer this question:

Order these numbers from least to greatest:

Binary: 11101 Decimal: 27 Hex: 1C

Decimal-Binary-Hex Conversion

Answer:

Binary: 11101 = 16 + 8 + 4 + 1 = 29

Decimal: 27

Hex: 1C = 16 + 12 = 28

So the answer is:

27, 1C, 11101

<u>For more practice use the odometer widget!</u>

Unit 1, Stage 5, Step 2

https://studio.code.org/s/csp1/stage/5/puzzle/2

Conditional Logic

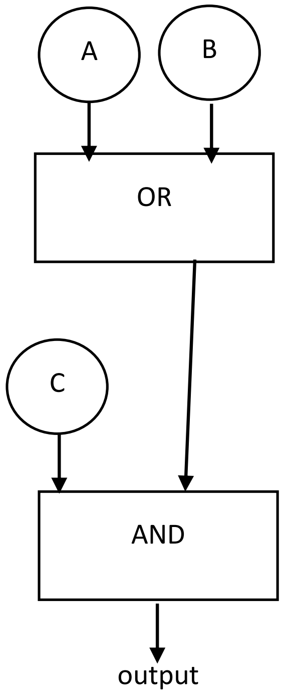
Know the conditional operators:

AND: both inputs need to be true for the output to be true

OR: If any of the inputs are true, then the output is true.

NOT: Flips the answer - Changes true to false and false to true.

On the exam you see questions like:



You need to be able to analyze what the inputs and outputs would be given certain conditions.

Can you do it? Answer this question:

If A is *true* and B is *false*, is there a value of C so that the output is *true*?

Conditional Logic

Answer:

If A is *true* and B is *false* then the top box ***A OR B*** is true.

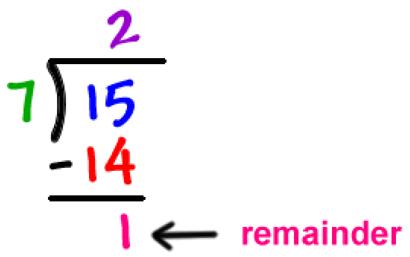
If C is *true* then the bottom box ***C AND (A OR B*****)** is *true* since both C and (A OR B) are *true*

MOD function

In computing, the modulo or mod function finds the remainder after division of one number by another. For example 11 mod 3 = 2 because 11/3 is 3 with remainder of 2.

Look at this example:

15 MOD 7



So the result is:

15 MOD 7 = 1

Can you do it? Answer these questions:

3 MOD 2

17 MOD 4

512 MOD 2

13 MOD 7

4 MOD 5

MOD function

Answers:

3 MOD 2 = 1

3/2 is 1 with a remainder of 1

17 MOD 4 = 1

17/4 is 4 with a remainder of 1

512 MOD 2 = 0

512/2 is 256 with a remainder of 0

13 MOD 7 = 6

13/7 is 1 with a remainder of 6

4 MOD 5 = 4

4/5 is 0 with a remainder of 4

For more practice use the Mod Clock widget!

Unit 4, Stage 7, Step 5

https://studio.code.org/s/csp4/stage/7/puzzle/5

Lists

It is very common to want to store lists of information in programs. This could be done with variables but they have many shortcomings when it comes to doing so. Most programming languages include a list structure of some sort to make it easier to hold many pieces of related information.

Creating a List

Lists have many features which make them different from variables, but most of what you've learned about variables also applies to lists. For example, just like a variable:

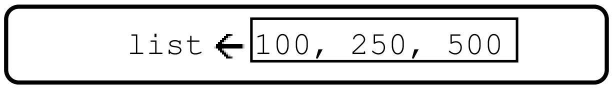
- Lists should be given a descriptive and meaningful name.
- Lists can be initialized/set.

Example: ***Creating an List using the Reference Language***

Text:

```
    list ← [100, 250, 500]
```

Block:



```
list ← 100, 250, 500
```

This list contains 3 values: 100, 250, 500. Notice that the values are separated with commas , and that the entire list is enclosed in brackets [ ]. In this example each item in the list can be accessed individually. The value of the list items are:

```
list[1] = 100
list[2] = 250
list[3] = 500
```

Note that the first index for a list in the reference language is 1. In many programming languages it is 0. For the purpose of the AP Exam you must always assume the first item in the list has an index of 1.

Just like a variable, each item in a list can be number, string, boolean, etc.

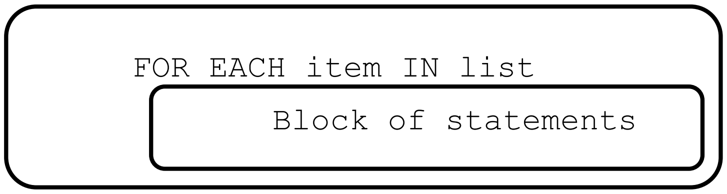
Lists

Accessing all items in a List

Text:

```
FOR EACH item IN list
{
     <block of statements>
}
```

Block:



```
FOR EACH item IN list
    Block of statements
```

Consider the following code:

```
total ← 0
numList ← [100, 250, 500]
FOR EACH number IN numList
{
     total ← total + number
}
DISPLAY (total)
```

In this example the loop will be executed 3 times since the length of the list is 3 items. The first time the loop is executed `number` is set to the first item in `numList` or `numList[1]` which is 100. The second time through the loop `number` is set to `numlist[2]` which is 250 and the third time `number` is set to `numList[3]` which is 500. The result of the code is that `total` will equal the sum of the items in the list (850).

Lists

Adding items to a List

There are 2 ways to add items to a list. INSERT and APPEND.

INSERT:

Text:

```
INSERT (list, i, value)
```

Block:

```
INSERT [list, i, value]
```

INSERT will add an item into the list at index **`i`** and shift the remaining items in the original list to the right.

Consider the following code:

```
numList ← [100, 250, 500]
INSERT (numList, 2, 200)
```

In this example the value 200 will be inserted into the list in the second item and the values 250 and 500 will be shifted right one place. The resulting is:

```
numList = [100, 200, 250, 500]
```

APPEND:

Text:

```
APPEND (list, value)
```

Block:

```
APPEND [list, value]
```

APPEND adds an item to the end of the list. Consider the following code:

```
numList ← [100, 250, 500]
APPEND (numList, 200)
```

In this example the value 200 will be added to the end of the list. The result is:

```
numList = [100, 250, 500, 200]
```

Removing items from a List

REMOVE:

Text:

```
REMOVE (list, i )
```

Block:

```
REMOVE [list, i ]
```

REMOVE removes an item in position i from the list. All items after i are shifted left one place. Consider the following code:

```
numList ← [100, 250, 500]
REMOVE (numList, 2)
```

In this example the value 250 will be removed from the end of the list. The result is:

```
numList = [100, 500]
```

Lists

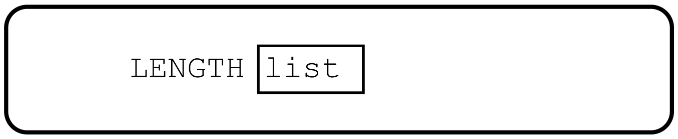
Finding the length of a List

LENGTH:

Text:

```
    LENGTH (list)
```

Block:



LENGTH returns the number of items in a list. Consider the following code:

```
    numList ← [100, 250, 500]
    listLength ← LENGTH (numList)
```

The result is:

```
numList = [100, 250,500] - note: it is unchanged

listLength = 3
```

Lists

Can you do it? Answer this question:

Consider the following. A company needs to make sure all of its employees have completed their training for the year. The company has a list of all their employees named `allEmployees` and a list that contains the employees that completed their training called `trainingComplete`. The want to create a list called `needTraining` that contains a list of all employees that have not taken the training. The company has written a function `notInList(list, item)` that returns `true` if the item is not in the list and `false` if it is. Complete to code to create the `needTraining` list.

```
needTraining ← []

FOR EACH name IN allEmployees
{











}
DISPLAY(needTraining)
```

Lists

Answer:

```
needTraining ← []

FOR EACH name IN allEmployees
{
    IF (notInList(trainingComplete, name)
    {
        APPEND(needTraining, name)
    }
}
DISPLAY(needTraining)
```

The code uses the `notInList` fuction to check if an employee has taken the training. If the employee is not in the list of employees who took the training then that name is added to the `needTraining` list.

Sample Exam Questions

To elicit evidence of student achievement of the course learning objectives, exam questions assess both the application of the computational thinking practices and an understanding of the big ideas. Exam questions may assess achievement of multiple learning objectives. They may also address content from more than one essential knowledge statement. Exam questions may be accompanied by nontextual stimulus material such as diagrams, charts, or other graphical illustrations. The sample questions that follow illustrate the relationship between the curriculum framework and the AP Computer Science Principles Exam and serve as examples of the types of questions that will appear on the exam. Each question is accompanied by a table containing the enduring understandings, learning objectives, computational thinking practices, and essential knowledge statements that the question addresses. Note that in cases where multiple learning objectives are provided for a question, the primary learning objective is listed first, along with the associated computational thinking practice and essential knowledge statement(s).

© 2014 The College Board

1. A video-streaming Web site uses 32-bit integers to count the number of times each video has been played. In anticipation of some videos being played more times than can be represented with 32 bits, the Web site is planning to change to 64-bit integers for the counter. Which of the following best describes the result of using 64-bit integers instead of 32-bit integers?

   (A) 2 times as many values can be represented.

   (B) 32 times as many values can be represented.

   (C) 2^{32} times as many values can be represented.

   (D) 32^2 times as many values can be represented.

Enduring Understandings	**Learning Objectives**	**Computational Thinking Practices**	**Essential Knowledge**
2.1 A variety of abstractions built upon binary sequences can be used to represent all digital data.	**2.1.1** Describe the variety of abstractions used to represent data. [P3]	**P3** Abstracting	**2.1.1A** **2.1.1B** **2.1.1E**

2. A programmer completes the user manual for a video game she has developed and realizes she has reversed the roles of goats and sheep throughout the text. Consider the programmer's goal of changing all occurrences of "goats" to "sheep" and all occurrences of "sheep" to "goats." The programmer will use the fact that the word "foxes" does not appear anywhere in the original text.

   Which of the following algorithms can be used to accomplish the programmer's goal?

   (A) First, change all occurrences of "goats" to "sheep."
   Then, change all occurrences of "sheep" to "goats."

   (B) First, change all occurrences of "goats" to "sheep."
   Then, change all occurrences of "sheep" to "goats."
   Last, change all occurrences of "foxes" to "sheep."

   (C) First, change all occurrences of "goats" to "foxes."
   Then, change all occurrences of "sheep" to "goats."
   Last, change all occurrences of "foxes" to "sheep."

   (D) First, change all occurrences of "goats" to "foxes."
   Then, change all occurrences of "foxes" to "sheep."
   Last, change all occurrences of "sheep" to "goats."

Enduring Understandings	**Learning Objectives**	**Computational Thinking Practices**	**Essential Knowledge**
4.1 Algorithms are precise sequences of instructions for processes that can be executed by a computer and are implemented using programming languages.	**4.1.1** Develop an algorithm for implementation in a program. [P2]	**P2** Creating computational artifacts	**4.1.1A** **4.1.1B**

© 2016 The College Board

3. ASCII is a character-encoding scheme that uses a numeric value to represent each character. For example, the uppercase letter "G" is represented by the decimal (base 10) value 71. A partial list of characters and their corresponding ASCII values are shown in the table below.

Decimal	ASCII Character
65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M

Decimal	ASCII Character
78	N
79	O
80	P
81	Q
82	R
83	S
84	T
85	U
86	V
87	W
88	X
89	Y
90	Z

ASCII characters can also be represented by hexadecimal numbers. According to ASCII character encoding, which of the following letters is represented by the hexadecimal (base 16) number 56?

(A) A

(B) L

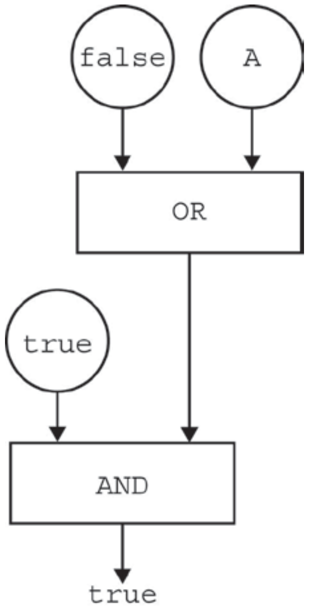
(C) V

(D) Y

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
2.1 A variety of abstractions built upon binary sequences can be used to represent all digital data.	**2.1.1** Describe the variety of abstractions used to represent data. [P3]	**P3** Abstracting	**2.1.1A** **2.1.1C** **2.1.1D** **2.1.1E** **2.1.1G**

© 2014 The College Board

4. The figure below shows a circuit composed of two logic gates. The output of the circuit is `true`.



Which of the following is a true statement about input `A`?

(A) Input `A` must be `true`.

(B) Input `A` must be `false`.

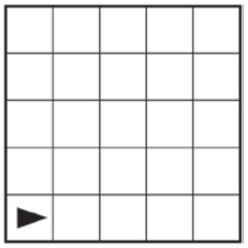
(C) Input `A` can be either `true` or `false`.

(D) There is no possible value of input `A` that will cause the circuit to have the output `true`.

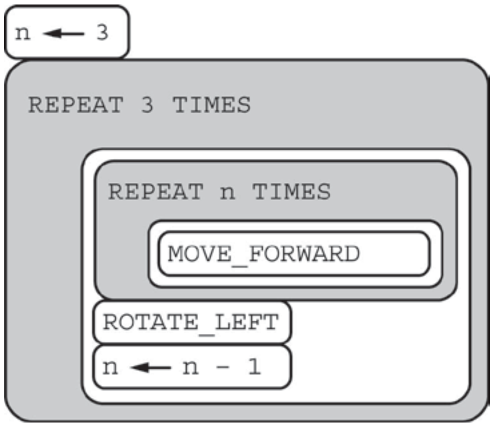
Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
2.2 Multiple levels of abstraction are used to write programs or to create other computational artifacts.	**2.2.3** Identify multiple levels of abstractions being used when writing programs. [P3]	**P3** Abstracting	**2.2.3E** **2.2.3F**

© 2016 The College Board

5. The following question uses a robot in a grid of squares. The robot is represented as a triangle, which is initially in the bottom left square of the grid and facing right.

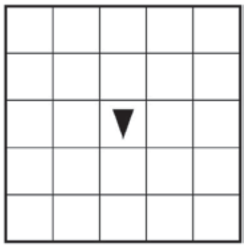


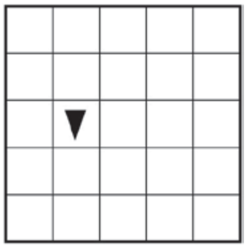
Consider the following code segment, which moves the robot in the grid.

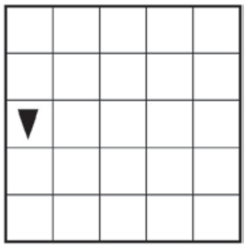


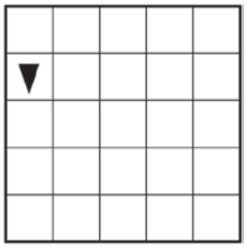
```
n ← 3
REPEAT 3 TIMES
    REPEAT n TIMES
        MOVE_FORWARD
    ROTATE_LEFT
    n ← n - 1
```

Which of the following shows the location of the robot after running the code segment?

(A)


(B)


(C)


(D)


© 2014 The College Board

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
5.2 People write programs to execute algorithms.	**5.2.1** Explain how programs implement algorithms. [P3]	**P3** Abstracting	**5.2.1A** **5.2.1B** **5.2.1C**

6. Which of the following statements describes a limitation of using a computer simulation to model a real-world object or system?

   (A) Computer simulations can only be built after the real-world object or system has been created.

   (B) Computer simulations only run on very powerful computers that are not available to the general public.

   (C) Computer simulations usually make some simplifying assumptions about the real-world object or system being modeled.

   (D) It is difficult to change input parameters or conditions when using computer simulations.

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
2.3 Models and simulations use abstraction to generate new understanding and knowledge.	**2.3.1** Use models and simulations to represent phenomena. [P3]	**P3** Abstracting	**2.3.1A** **2.3.1C** **2.3.1D**

7. A certain social media Web site allows users to post messages and to comment on other messages that have been posted. When a user posts a message, the message itself is considered data. In addition to the data, the site stores the following metadata.

   - The time the message was posted
   - The name of the user who posted the message
   - The names of any users who comment on the message and the times the comments were made

© 2016 The College Board

For which of the following goals would it be more useful to analyze the data instead of the metadata?

(A) To determine the users who post messages most frequently

(B) To determine the time of day that the site is most active

(C) To determine the topics that many users are posting about

(D) To determine which posts from a particular user have received the greatest number of comments

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
3.2 Computing facilitates exploration and the discovery of connections in information.	**3.2.1** Extract information from data to discover and explain connections or trends. [P1]	**P1** Connecting computing	**3.2.1B** **3.2.1G** **3.2.1H** **3.2.1I**

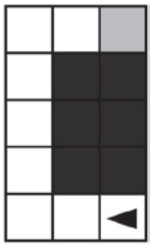
8. The program segment below is intended to move a robot in a grid to a gray square. The program segment uses the procedure `GoalReached`, which evaluates to `true` if the robot is in the gray square and evaluates to `false` otherwise. The robot in each grid is represented as a triangle and is initially facing left. The robot can move into a white or gray square but cannot move into a black region.

```
REPEAT UNTIL (GoalReached ())
{
   IF (CAN_MOVE (forward))
   {
      MOVE_FORWARD ()
   }
   IF (CAN_MOVE (right))
   {
      ROTATE_RIGHT ()
   }
   IF (CAN_MOVE (left))
   {
      ROTATE_LEFT ()
   }
}
```

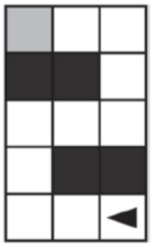
© 2014 The College Board

For which of the following grids does the program NOT correctly move the robot to the gray square?

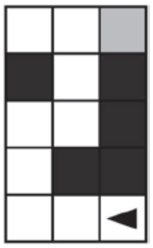
(A)



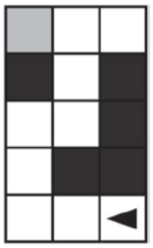
(B)



(C)



(D)



Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
4.2 Algorithms can solve many, but not all, computational problems.	**4.2.4** Evaluate algorithms analytically and empirically for efficiency, correctness, and clarity. [P4]	**P4** Analyzing problems and artifacts	**4.2.4B**

9. The table below shows the time a computer system takes to complete a specified task on the customer data of different-sized companies.

Task	**Small Company (approximately 100 customers)**	**Medium Company (approximately 1,000 customers)**	**Large Company (approximately 10,000 customers)**
Backing up data	2 hours	20 hours	200 hours
Deleting entries from data	100 hours	200 hours	300 hours
Searching through data	250 hours	300 hours	350 hours
Sorting data	0.01 hour	1 hour	100 hours

© 2016 The College Board

Based on the information in the table, which of the following tasks is likely to take the longest amount of time when scaled up for a very large company of approximately 100,000 customers?

(A) Backing up data

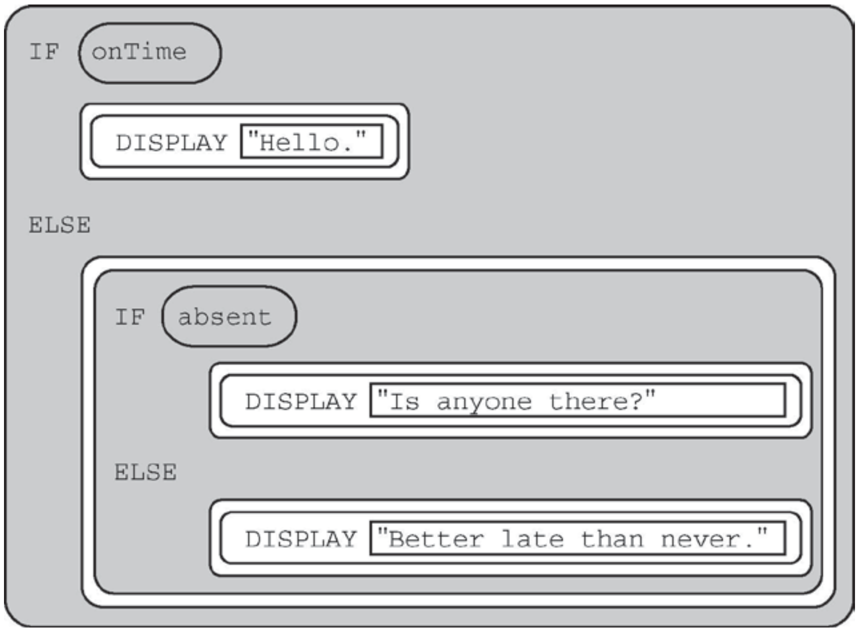
(B) Deleting entries from data

(C) Searching through data

(D) Sorting data

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
3.2 Computing facilitates exploration and the discovery of connections in information.	**3.2.2** Determine how large data sets impact the use of computational processes to discover information and knowledge. [P3]	**P3** Abstracting	**3.2.2E** **3.2.2F** **3.2.2H**

10. Consider the code segment below.



```
IF onTime
    DISPLAY "Hello."
ELSE
    IF absent
        DISPLAY "Is anyone there?"
    ELSE
        DISPLAY "Better late than never."
```

If the variables `onTime` and `absent` both have the value `false`, what is displayed as a result of running the code segment?

(A) `Is anyone there?`

(B) `Better late than never.`

(C) `Hello. Is anyone there?`

(D) `Hello. Better late than never.`

© 2014 The College Board

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
4.1 Algorithms are precise sequences of instructions for processes that can be executed by a computer and are implemented using programming languages.	**4.1.1** Develop an algorithm for implementation in a program. [P2]	**P2** Creating computational artifacts	**4.1.1A** **4.1.1C**

11. Under which of the following conditions is it most beneficial to use a heuristic approach to solve a problem?

(A) When the problem can be solved in a reasonable time and an approximate solution is acceptable

(B) When the problem can be solved in a reasonable time and an exact solution is needed

(C) When the problem cannot be solved in a reasonable time and an approximate solution is acceptable

(D) When the problem cannot be solved in a reasonable time and an exact solution is needed

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
4.2 Algorithms can solve many, but not all, computational problems.	**4.2.2** Explain the difference between solvable and unsolvable problems in computer science. [P1]	**P1** Connecting computing	**4.2.2A** **4.2.2B** **4.2.2C**

12. Which of the following are true statements about digital certificates in Web browsers?

I. Digital certificates are used to verify the ownership of encrypted keys used in secured communication.

II. Digital certificates are used to verify that the connection to a Web site is fault tolerant.

(A) I only

(B) II only

(C) I and II

(D) Neither I nor II

© 2016 The College Board

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
6.3 Cybersecurity is an important concern for the Internet and the systems built on it.	**6.3.1** Identify existing cybersecurity concerns and potential options that address these issues with the Internet and the systems built on it. [P1]	**P1** Connecting computing	**6.3.1H** **6.3.1L** **6.3.1M**

13. There are 32 students standing in a classroom. Two different algorithms are given for finding the average height of the students.

Algorithm A

Step 1: All students stand.

Step 2: A randomly selected student writes his or her height on a card and is seated.

Step 3: A randomly selected standing student adds his or her height to the value on the card, records the new value on the card, and is seated. The previous value on the card is erased.

Step 4: Repeat step 3 until no students remain standing.

Step 5: The sum on the card is divided by 32. The result is given to the teacher.

Algorithm B

Step 1: All students stand.

Step 2: Each student is given a card. Each student writes his or her height on the card.

Step 3: Standing students form random pairs at the same time. Each pair adds the numbers written on their cards and writes the result on one student's card; the other student is seated. The previous value on the card is erased.

Step 4: Repeat step 3 until one student remains standing.

Step 5: The sum on the last student's card is divided by 32. The result is given to the teacher.

Which of the following statements is true?

(A) Algorithm A always calculates the correct average, but Algorithm B does not.

(B) Algorithm B always calculates the correct average, but Algorithm A does not.

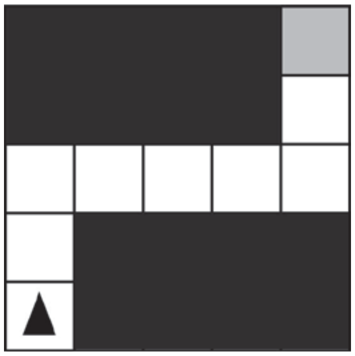
(C) Both Algorithm A and Algorithm B always calculate the correct average.

(D) Neither Algorithm A nor Algorithm B calculates the correct average.

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
4.2 Algorithms can solve many, but not all, computational problems.	**4.2.4** Evaluate algorithms analytically and empirically for efficiency, correctness, and clarity. [P4]	**P4** Analyzing problems and artifacts	**4.2.4C**

© 2014 The College Board

14. The figure below shows a robot in a grid of squares. The robot is represented as a triangle, which is initially facing upward. The robot can move into a white or gray square but cannot move into a black region.



Consider the procedure `MoveAndTurn` below.

```
PROCEDURE MoveAndTurn (numMoves, numTurns)
    REPEAT numMoves TIMES
        MOVE_FORWARD
    REPEAT numTurns TIMES
        ROTATE_RIGHT
```

Which of the following code segments will move the robot to the gray square?

(A)
```
MoveAndTurn (1, 2)
MoveAndTurn (3, 4)
MoveAndTurn (0, 2)
```

(B)
```
MoveAndTurn (2, 1)
MoveAndTurn (4, 1)
MoveAndTurn (2, 0)
```

(C)
```
MoveAndTurn (2, 1)
MoveAndTurn (4, 3)
MoveAndTurn (2, 0)
```

(D)
```
MoveAndTurn (3, 1)
MoveAndTurn (5, 3)
MoveAndTurn (3, 0)
```

© 2016 The College Board

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
5.3 Programming is facilitated by appropriate abstractions.	**5.3.1** Use abstraction to manage complexity in programs. [P3]	**P3** Abstracting	**5.3.1A** **5.3.1B** **5.3.1D** **5.3.1E** **5.3.1F** **5.3.1G**

15. Biologists often attach tracking collars to wild animals. For each animal, the following geolocation data is collected at frequent intervals.

- The time
- The date
- The location of the animal

Which of the following questions about a particular animal could NOT be answered using <u>only</u> the data collected from the tracking collars?

(A) Approximately how many miles did the animal travel in one week?

(B) Does the animal travel in groups with other tracked animals?

(C) Do the movement patterns of the animal vary according to the weather?

(D) In what geographic locations does the animal typically travel?

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
3.1 People use computer programs to process information to gain insight and knowledge.	**3.1.1** Find patterns and test hypotheses about digitally processed information to gain insight and knowledge. [P4]	**P4** Analyzing problems and artifacts	**3.1.1E**

16. A summer camp offers a morning session and an afternoon session. The list `morningList` contains the names of all children attending the morning session, and the list `afternoonList` contains the names of all children attending the afternoon session.

Only children who attend both sessions eat lunch at the camp. The camp director wants to create `lunchList`, which will contain the names of children attending both sessions.

© 2014 The College Board

The following code segment is intended to create `lunchList`, which is initially empty. It uses the procedure `IsFound (list, name)`, which returns `true` if `name` is found in `list` and returns `false` otherwise.

```
FOR EACH child IN morningList
{
   <MISSING CODE>
}
```

Which of the following could replace `<MISSING CODE>` so that the code segment works as intended?

(A)
```
IF (IsFound (afternoonList, child))
{
   APPEND (lunchList, child)
}
```

(B)
```
IF (IsFound (lunchList, child))
{
   APPEND (afternoonList, child)
}
```

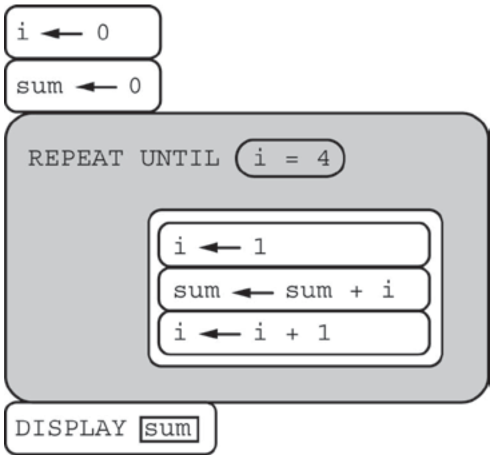
(C)
```
IF (IsFound (morningList, child))
{
   APPEND (lunchList, child)
}
```

(D)
```
IF ((IsFound (morningList, child)) OR
    (IsFound (afternoonList, child)))
{
   APPEND (lunchList, child)
}
```

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
5.3 Programming is facilitated by appropriate abstractions.	**5.3.1** Use abstraction to manage complexity in programs. [P3]	**P3** Abstracting	**5.3.1G** **5.3.1K** **5.3.1L**

© 2016 The College Board

17. Consider the following program code.



```
i ← 0
sum ← 0
REPEAT UNTIL (i = 4)
    i ← 1
    sum ← sum + i
    i ← i + 1
DISPLAY sum
```

Which of the following best describes the result of running the program code?

(A) The number `0` is displayed.

(B) The number `6` is displayed.

(C) The number `10` is displayed.

(D) Nothing is displayed; the program results in an infinite loop.

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
5.4 Programs are developed, maintained, and used by people for different purposes.	**5.4.1** Evaluate the correctness of a program. [P4]	**P4** Analyzing problems and artifacts	**5.4.1E** **5.4.1K** **5.4.1N**

18. Which of the following is a true statement about data compression?

(A) Data compression is only useful for files being transmitted over the Internet.

(B) Regardless of the compression technique used, once a data file is compressed, it cannot be restored to its original state.

(C) Sending a compressed version of a file ensures that the contents of the file cannot be intercepted by an unauthorized user.

(D) There are trade-offs involved in choosing a compression technique for storing and transmitting data.

© 2014 The College Board

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
3.3 There are trade-offs when representing information as digital data.	**3.3.1** Analyze how data representation, storage, security, and transmission of data involve computational manipulation of information. [P4]	**P4** Analyzing problems and artifacts	**3.3.1C** **3.3.1D** **3.3.1E**

19. An office building has two floors. A computer program is used to control an elevator that travels between the two floors. Physical sensors are used to set the following Boolean variables.

Variable	Description
`onFloor1`	set to `true` if the elevator is stopped on floor 1; otherwise set to `false`
`onFloor2`	set to `true` if the elevator is stopped on floor 2; otherwise set to `false`
`callTo1`	set to `true` if the elevator is called to floor 1; otherwise set to `false`
`callTo2`	set to `true` if the elevator is called to floor 2; otherwise set to `false`

The elevator moves when the door is closed and the elevator is called to the floor that it is not currently on. Which of the following Boolean expressions can be used in a selection statement to cause the elevator to move?

(A) `(onFloor1 AND callTo2) AND (onFloor2 AND callTo1)`

(B) `(onFloor1 AND callTo2) OR (onFloor2 AND callTo1)`

(C) `(onFloor1 OR callTo2) AND (onFloor2 OR callTo1)`

(D) `(onFloor1 OR callTo2) OR (onFloor2 OR callTo1)`

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
5.5 Programming uses mathematical and logical concepts.	**5.5.1** Employ appropriate mathematical and logical concepts in programming. [P4]	**P1** Connecting computing	**5.5.1E** **5.5.1F** **5.5.1G**

20. According to the domain name system (DNS), which of the following is a subdomain of the domain `example.com`?

(A) `about.example.com`

(B) `example.co.uk`

(C) `example.com.org`

(D) `example.org`

© 2016 The College Board

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
6.2 Characteristics of the Internet influence the systems built on it.	**6.2.1** Explain characteristics of the Internet and the systems built on it. [P5]	**P5** Communicating	**6.2.1B**

21. Which of the following algorithms require both selection and iteration?

    Select two answers.

    (A) An algorithm that, given two integers, displays the greater of the two integers

    (B) An algorithm that, given a list of integers, displays the number of even integers in the list

    (C) An algorithm that, given a list of integers, displays only the negative integers in the list

    (D) An algorithm that, given a list of integers, displays the sum of the integers in the list

Enduring Understandings	Learning Objectives	Computational Thinking Practices	Essential Knowledge
4.1 Algorithms are precise sequences of instructions for processes that can be executed by a computer and are implemented using programming languages.	**4.1.1** Develop an algorithm for implementation in a program. [P2]	**P2** Creating computational artifacts	**4.1.1A** **4.1.1C** **4.1.1D**

22. A teacher uses the following program to adjust student grades on an assignment by adding 5 points to each student's original grade. However, if adding 5 points to a student's original grade causes the grade to exceed 100 points, the student will receive the maximum possible score of 100 points. The students' original grades are stored in the list `gradeList`, which is indexed from 1 to `n`.

```
i ← 1
REPEAT n TIMES
{
   <MISSING CODE>
   i ← i + 1
}
```

SAMPLE EXAM QUESTIONS

© 2014 The College Board

The teacher has the following procedures available.

Procedure	**Explanation**
`min (a, b)`	Returns the lesser of the two values `a` and `b`
`max (a, b)`	Returns the greater of the two values `a` and `b`

Which of the following code segments can replace `<MISSING CODE>` so that the program works as intended?

Select <u>two</u> answers.

(A) `gradeList[i] ← min (gradeList[i] + 5, 100)`

(B) `gradeList[i] ← max (gradeList[i] + 5, 100)`

(C)
```
gradeList[i] ← gradeList[i] + 5
IF (gradeList[i] > 100)
{
gradeList[i] ← gradeList[i] - 5
}
```

(D)
```
gradeList[i] ← gradeList[i] + 5
IF (gradeList[i] >  100)
{
gradeList[i] ←100
}
```

Enduring Understandings	**Learning Objectives**	**Computational Thinking Practices**	**Essential Knowledge**
5.5 Programming uses mathematical and logical concepts.	**5.5.1** Employ appropriate mathematical and logical concepts in programming. [P1]	**P1** Connecting computing	**5.5.1A** **5.5.1B** **5.5.1H**

© 2016 The College Board

Answers to Sample Exam Questions

1 – C	12 – A
2 – C	13 – C
3 – C	14 – C
4 – A	15 – C
5 – A	16 – A
6 – C	17 – D
7 – C	18 – D
8 – D	19 – B
9 – D	20 – A
10 – B	21 – B, C
11 – C	22 – A, D

Return to
Table of Contents

© 2014 The College Board