

CSP @ Home Hard-copy Packet

Student Directions:

Use the activities or links in this packet to guide individual work on the CSP course content including:

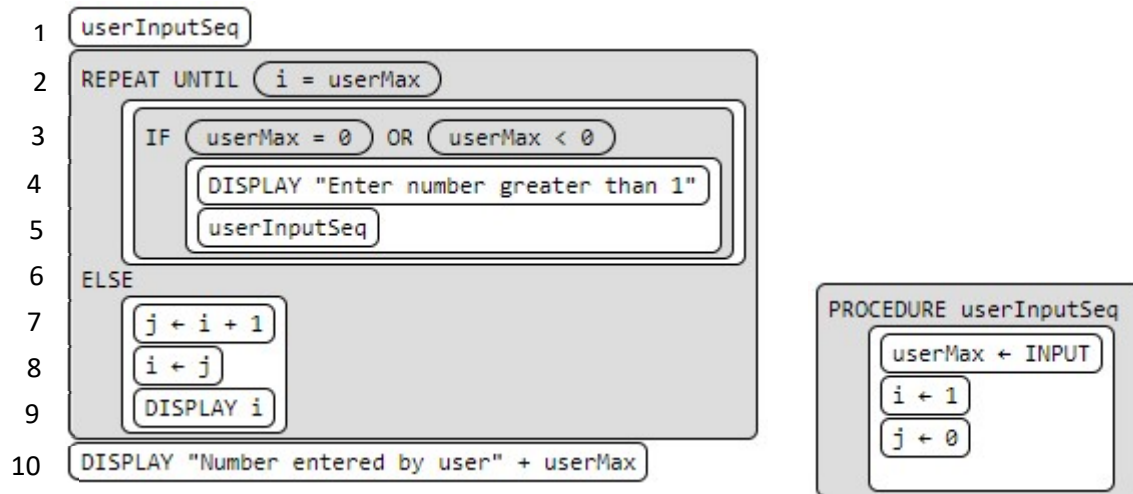
- Writing code to practice application and syntax in CSP pseudo code languages (text and Blockly)
- Reviewing the CSP course learning objectives and vocabulary to increase content knowledge of CS
- Watching video demonstrations and lectures to review concepts and application of syntax and code
- Completing online lessons to code given problems or answer questions

NOTES:

Code Trace Analysis Problems using CSP Exam Reference Sheet (provided on last 6 pages)

Problem 1

Consider the following sequence of code that uses a procedure, `userInputSeq`, and includes a repetitive loop with nested if, else condition:



- What is the action performed by this program segment lines 1-10? _____

- What action is performed by the `userInputSeq` procedure? _____
- What terminates this iteration? _____
- What type of Boolean condition is used on line 3 to evaluate input? _____
- On which line is concatenation used? _____
- If the `userInputSeq` procedure were not called on line 5, what would result?

Problem 2

Consider the following sequence of programming code to complete prompts below:



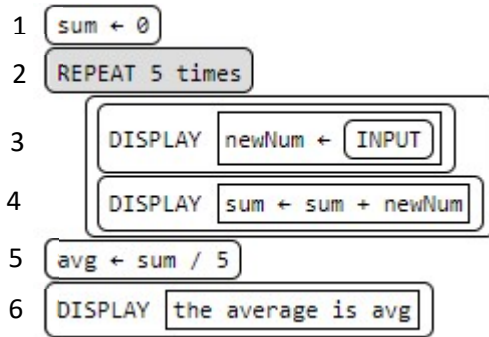
1. What type of algorithm is the code segment? _____
2. What variable is used in the segment? _____
3. Use in a program by writing code in the box below so that user input is received for the variable and adding statements to produce the output shown to the right:

Desired output:

```
Hello
Goodbye
Hello
Goodbye
Hello
Goodbye
Hello
Goodbye
User entered #
      //call variable do
      not hard code #
Program terminated
```

Problem 3

Consider the following sequence of code that uses a repetitive loop:



1. What is the purpose of this program sequence? _____
2. Describe what could be changed on lines 2 and 6 to produce a more optimal program:

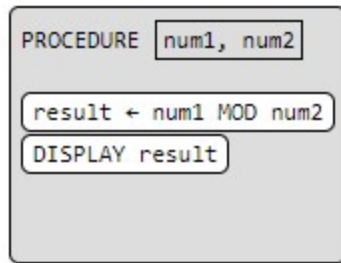
3. Write pseudo code to describe the algorithm in line 5 of the program:

4. Re-write line 6 so that it produces the desired output of the computed average rather than simply text:

5. Rewrite the code above creating it into a procedure, findAverage, and utilizing a control structure so that until the user enters number zero (0) the program terminates displaying the average

Problem 4

Consider the following sequence that receives two parameters and produces their remainder. Use this procedure to solve the problems given.

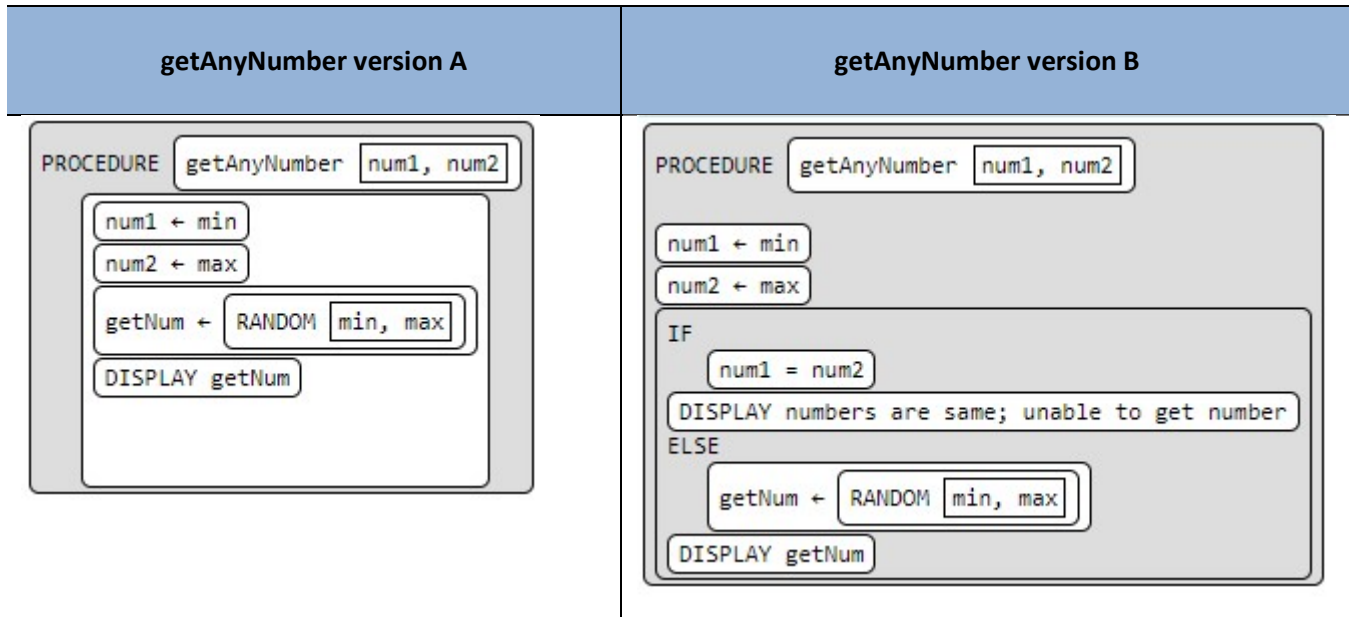


	num1	num2	result
a.	4	2	
b.	15	2	
c.	17	3	
d.	25	4	
e.	40	6	
f.	55	5	
g.	126	4	
h.	151	5	
i.	170	4	
j.	181	2	

-
- Which of the following is the best name for the procedure above?
 - findNum
 - findREM
 - findProduct
 - findRemainder
 - Write a program that calls the procedure above to receive user inputs for num1 and num2.

Problem 5

Consider the following procedure then answer the questions/prompts below:



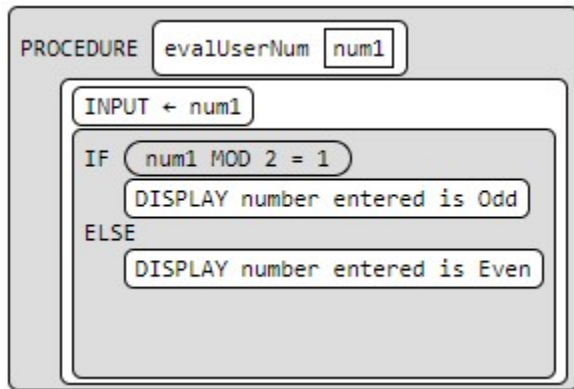
- Given parameters (1, 100), in maximum value could getNum result? _____
- Explain the functionality check of the if, else control structure in Program B:

- List the variables required for the getAnyNumber procedure:

- Consider getAnyNumber version B above with Problem 1 (page 1) userInputSeq procedure. How might getAnyNumber version B be improved with such a procedure call and what functionality would it create?

Problem 6

Consider the following procedure `evalUserNum`:



1. Describe the procedure functionality and output in natural language:

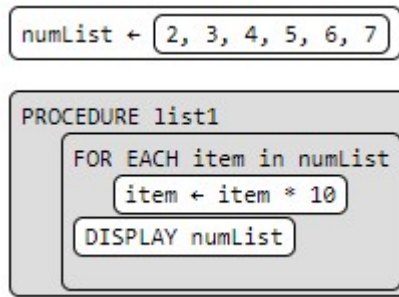
2. Call `evalUserNum (55)`. What is the result? _____
3. Using `getAnyNum` version A (problem 5) and `evalUserNum` (problem 6), diagram and pseudocode below the **simulation of rolling of a single die and display of both the number output and the number's odd/even type.**

Diagram:

Pseudocode/Natural Language:

Problem 7

Consider the `numList` below. Use the list and the `list1` procedure to respond to the given prompts:



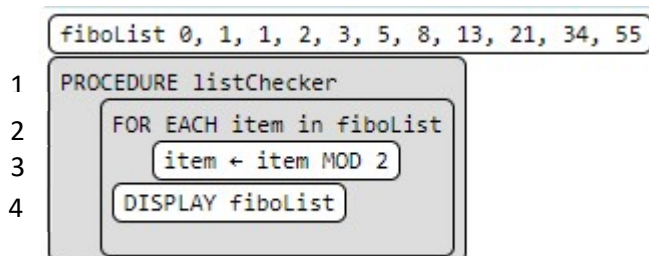
1. What is the result of a call to the `list1` procedure (write out DISPLAY):

2. Edit `numList` as follows using correct CSP list operations:

	Action:	Write CSP syntax here:
a)	insert number 1 at position 1	
b)	add the number 8 to end	
c)	add the number 9 to end	
d)	remove element at index 2	
e)	remove element at index 5	
f)	call <code>numList</code>	
g)	record result from a-f	
h)	remove the number 4 from <code>numList</code>	
i)	add the number 10 to end	
j)	display <code>LENGTH (numList)</code>	
k)	call <code>numList</code> after a-k recording elements	

Problem 8

Consider `fiboList` and `listChecker` procedure shown below. Evaluate according to prompts/questions:



- Describe in natural language the action of line 3 in the `listChecker` procedure:

- Record the output of a call to `listChecker`: _____
- What element is an index 1 of `fibolist`? (circle) 0 1
- What is the `LENGTH` of `fibolist` after call to `listChecker`? _____
- After `listChecker` is called, the display resembles what type of sequence? _____
- Create a new list with a length of 8 that once called in `listChecker` will result in the binary representation of 150. For HINT see bottom of page.

bin150List _____, _____, _____, _____, _____, _____, _____, _____
- Create a new list with `n` length that once called in `listChecker` will result in the binary representation of 226.

bin226List _____, _____, _____, _____, _____, _____, _____, _____
- Create a new list with `n` length that once called in `listChecker` will result in the binary representation of your age.

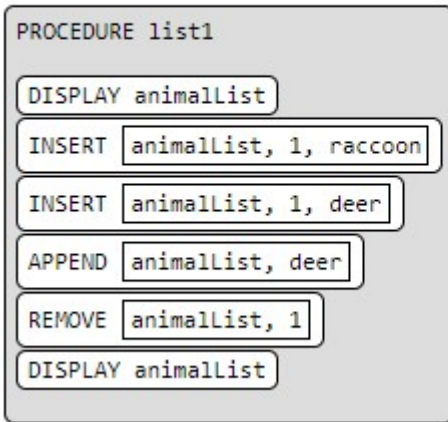
guessMyAgeList _____, _____, _____, _____

HINT: find the binary representation for decimal 150 then determine numbers that would yield that result once run through the `listChecker` procedure above

Problem 9

Consider the following sequence

```
animalList ← [turtle, duck, swan, frog, beaver, lizard]
```



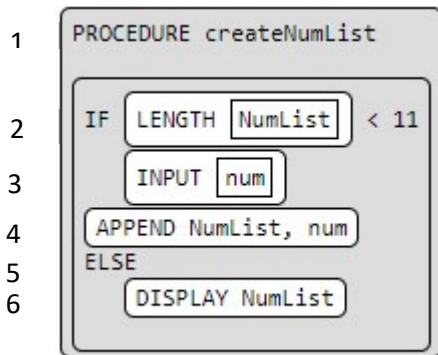
- Write code to copy contents of animalList to animalList2 _____
- What is the result of a call to the list1 procedure (write out DISPLAY):

- Write code sequence to create a for list2 procedure as follows using correct CSP list operations:

	Action:	Write CSP syntax here:
a.	add bluejay to the end of list	
b.	move frog from current index to	
c.	first index (requires 2 statements)	
d.	add snail after beaver	
e.	display animalList2	

Problem 10

Consider the following sequence



- Write pseudocode for the line numbers given:

2 _____

3 _____

4 _____

6 _____

- Explain in natural language the functionality of the selection algorithm (if, else):

- As shown, what will be the size of NumList? _____

- Would it be possible to create a larger or smaller NumList using the createNumList procedure? Why or why not?

- Add/edit the procedure to allow the user to input a variable, `setListSize`, then write a program sequence that correctly receives user input and calls createNumList to achieve the desired result. [use space to right of code]

Number Conversion Review

Record the number systems by base:

Base 2 (binary)

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Base 10 (decimal)

10^3	10^2	10^1	10^0

Base 16 (hexadecimal)

16^3	16^2	16^1	16^0

Supplemental Resources about Numbering Systems:

Hex Colors Conversion	http://www.coloring.com/ Right click the color palette and choose Inspect to expose the HTML and see the RGB values for each color selection; Create an artifact poster of your finished drawing with legend of the RGB values/Hex Code for colors used in your drawing
Flippy Bit Hex Game	http://flippybitandtheattackofthehexadecimalsfrombase16.com/
Hex Game	http://www.purposegames.com/game/049fc90a
Converter Tool	http://www.cut-the-knot.org/binary.shtml Use this one then use any coding language to create your own
YouTube How to Convert	https://www.youtube.com/watch?v=QgVc1TI-JDA

Practice On Numbers You Know

DECIMAL → BINARY || HEXIDECIMAL

Decimal # ↓		Bin Hex ↓
	Your birthday day Example, July 9 th would be the "9"	bin
	Number of Instagram followers or Snap streaks that you currently have is...	bin hex
	Miles from school to your home	bin hex
R G B	RGB value for the color Gold	Red:
	gold Hex #FFD700	bin hex
		Green:
		bin hex
		Blue:
		bin hex
	Your Street Address Example, 536 Park Place would be the "536"	bin hex
	Your age + your shoe size + height in inches	bin hex
	Number of songs on favorite play list	hex
	Number of colored stickers on all sides of a Rubik's Cube	bin hex
	The current year	hex

D → B									
Decimal #	128	64	32	16	8	4	2	1	Find the Binary #
145	1	0	0	1	0	0	0	1	10010001
134									
85									
99									
248									
111									
88									
13									
200									

[illegible]

Compare Bases to Determine Relation

Convert the bases and make your answer choice > < or =.

Binary	> < or =	Hexadecimal
11011100	1.	75
10101011	2.	AB
101101	3.	2F
1011000	4.	58
11001000	5.	B2
01011110	6.	5E
10110001	7.	B2
1010	8.	14
1100100	9.	62
1111101000	10.	3DD
001111011101	11.	3D3
1010001110	12.	28E
111110100	13.	226
1011000	14.	66
10100111	15.	A7
10	16.	10
00010001	17.	11
100100	18.	42

Alphabet Robot Practice

Directions: (1) Shade the squares of one of your initials (as large as possible); (2) using CSP Exam Reference Sheet Robot Procedure, create pseudo code for a robot to follow the path to draw the initial; (3) refine your pseudo code into an algorithm/statements making it as efficient as possible; (4) write final algorithm in box

Algorithm to Create the Letter _____

Flowchart & Pseudo Code Project

Directions:

1. Analyze one (1) of the scenarios below to create a rough draft flowchart
2. Refine the draft and add pseudocode
3. Create a flowchart diagram to detail the process

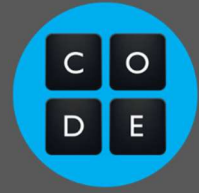
Option A	Diagram the decision-making process of approaching a traffic signal at an intersection. Include the options for red, yellow and green lights in your product.
Option B	Diagram the decision making to select a soda from a vending machine. Include both drink selection and payment using bills or coins.
Option C	Diagram the decision making to respond to one's alarm clock or choose to snooze a while longer.
Option D	Diagram the process to walk from your English class to the theatre including decision choices based on traffic in the hallways.

There are several online, free flowchart making programs such as <https://www.draw.io/> or https://creately.com/app/?templD=gc7qvpsj1&login_type=demo# if you wish to digitize your diagrams (not necessary but can be fun to learn a new skill set).



What is the Internet?

<http://bit.ly/2RHclfm>



The Internet began as a result of...

What was the purpose of ARPANET?

Of the types of communications network that resulted from this experiment, which one worked?

Describe this type of communications network.

Is anyone in charge of the internet (yes or no) Explain.

The Internet is made up of...

The Internet: Wires, Cables, & Wifi

<http://bit.ly/2Rm8NzN>

What type of information does the Internet “ship”?

How can a bit be described?

Why is it called binary code?

We can physically send bits by...

What is bandwidth?

How is bandwidth measured?

What is latency?

How are bits transferred through fiber optic cable?

Sending Data: Pros & Cons

Electric ⚡		
Light -----		
Radio 📶		

The Internet: IP Address & DNS

<http://bit.ly/2QBJLHM>

The Internet is a network of _____.

The Internet is a design philosophy and an architecture expressed as a set of _____.

A protocol is ...

What does a new technology need to know to use the Internet?

All the different devices on the Internet have _____.

What is one of the most important protocols on the Internet?

A computers address is called _____.

Example: 174.129.14.120; Traditional IP Addresses are how long?

How many bits for each part?

Describe IPv4 compared to IPv6.

How many bits are used by IPv6?

What is DNS? Explain what it does.

How are DNS servers connected?

Why is DNS susceptible to cyberattacks?

What is DNS spoofing?

The Internet: Packets, Routing and Reliability

<http://bit.ly/2RJdz9Q>

Is the Internet made up of made of direct, dedicated connections?

Do packets travel along the Internet on a fixed path?

Each packet has the Internet address of _____ and _____.

Special computers on the Internet called _____ act like traffic managers to keep packets moving through networks smoothly.

If one route is congested, individual packets may travel _____ routes.

As part of the _____, every _____ keeps track of multiple paths for sending packets, and it chooses the cheapest available path for each piece of data based on destination IP address for the packet.

Often the best route for data to travel isn't necessarily the _____.

Having options for paths makes the network _____.

This is the basis for a key principle of the Internet is _____.

_____ manages the sending and receiving of all your data as packets.

The Internet: HTTP and HTML

<http://bit.ly/2QxsHCQ>

We can grow and scale the Internet without _____ for anybody using it.

URL, which stands for _____.

HTTP stands for _____.

If a web page has a lot of different images it may load slower. Why?

Safe websites prevent this, by asking your web browser to communicate on a secure channel using something called _____ and its successor Transport Layer Security.

SSL and TLS are active when you see the little _____ that appears in your browser address bar, next to the HTTPS.

When a website asks your browser to engage in a secure connection, it first provides a _____.

If a website tries to start a secure connection without a properly issued digital certificate, your browser will _____.

_____ and _____ manage the sending and receiving of HTML, media files, or anything on the web.

Internet packets themselves are made up of _____, sequences of 1s and 0s that are physically sent through electric wires, fiber optic cables, and wireless networks.

The Internet: Encryption & Public Keys

<http://bit.ly/2RFlrt9>

The Internet is an _____ and _____ system. Users exchange _____ . A method to keep the data safe is _____ .

Encryption is _____ or _____

the message to _____ the original text.

Decryption _____ the message to make it readable.

_____ is the first known to have used encryption. He used it for _____ purposes.

A cipher is an _____ that shifts a certain number of _____ down the alphabet. The _____ is the number of shifts in the algorithm shift.

In the modern world, we use _____ bit keys to encrypt.

Symmetric encryption: The sender and receiver _____ a private key.

Asymmetric encryption uses a _____ key and a _____ key.



Public key cryptography is the _____ of all _____ messages on the open Internet including the security protocols _____ and _____. When you see _____ on your browser's address bar, you know public key encryption is being used.



The Internet: Cybersecurity & Crime

<http://bit.ly/2SL1s9R>

Cybercrime causes problems _____, _____, and in national _____.

Cybercriminals exploit _____ in _____ and _____ taking advantage of unintentional _____ of software users.

Viruses: An _____ program that once installed _____.

Viruses can be disguised as _____. A backdoor is a _____ that can be used by hackers to gain entry to a system.

A _____ is known as a digital army.

DDoS: A distributed _____ of _____ attack is when a website is _____ with too many _____. It is distributed when the site is _____ at once.

Phishing: When _____ emails are sent to _____ people to respond with private _____.

90% of the time a system is hacked it is because of a _____ mistake made by a _____.



Cybersecurity101

<http://bit.ly/2Qy3EiT>



The point of the Internet originally was for super computers to _____ to _____.

Networks came when computers could _____ information _____ and _____ to each other.

Viruses are a form of malware.

Malware is _____ (look up – not defined on video). What are three forms of malware common today: _____, _____, _____

List 3 ways to protect your online accounts and data:

- 1) _____
- 2) _____
- 3) _____

Learn more and play game at

<http://www.pbs.org/wgbh/nova/labs/lab/cyber/>

The Internet: How Search Works

<http://bit.ly/2VD8rDn>

Search engines are asked to answer _____ questions and terribly _____ questions.

The search engine is _____ going to the WWW to search the request in real time.

The Internet is a _____ of pages connected to each other by _____.



Search engines are _____ running a _____ called a “_____” that _____ through pages _____ information about them and _____ the hyperlinks.

Information found by the “_____” is added to the _____ which is a special database.

A _____ algorithm is used to help determine which pages might be the most useful.

The _____ algorithm is named for _____ and is used to rank pages according to the quality of information based on numbers of links to that page.

Search programs are always _____ to improve their _____ so they will return better _____.

Search engines use _____ which is a type of artificial intelligence.



How Google Search Works

<http://bit.ly/2skbSBs>

#1 When you do a Google Search you are not actually searching the web.....you are searching _____ of the web.

Matt uses the word “fetch” to refer to what action by the spiders?

The Page Rank Algorithm looks at what two things?

Each page has an overall _____ to help give search results to users of Google.

Google NEVER takes \$\$ to _____, _____, or _____.

Each search result includes what information?

1. _____

2. _____

3. _____

4. _____

5. And related searches

6. And sometimes ads to the right and top of the page

CSP CED LO/EK Concepts and Vocabulary Complete using the [CSPv2017 CED](#).

Also see Quizlet for CSP vocabulary sets and resources made/shared by others

Creativity	1.1.1	Creative development process	
	1.2.1	Computational artifact	
	1.2.2C	Computing tools	
	1.2.4	Collaborate/ collaboration	
	1.2.5	Analyze	
	1.3.1	Creative expression (digital)	
Abstraction	2.1.1	Abstraction	
	2.1.1A	Digital data	
	2.1.1B	Bits	
	2.1.1D	Number bases	
		Binary numbers (base-2)	
	2.1.1F	Hexadecimal (base-16)	
	2.1.2	Binary sequences	
	2.1.2B	Programming languages	
	2.1.2C	Real numbers (floating-point)	
	2.2.2A	Software	
	2.2.3	Levels of abstractions	
	2.2.3B	High-level languages	
	2.2.3C	Low-level languages	
	2.2.3D	Abstraction hierarchy	
	2.2.3E	Binary data	
	2.2.3F	Boolean function	
	2.2.3F	Logic gate	
	2.2.3G	Chip (as an abstraction)	
	2.2.3H-I	Hardware	
	2.2.3K	Lower-level abstractions	
	2.2.3K	Higher-level abstractions	
	2.3.1	Models (see 2.3.1A-C)	
	2.3.1D	Simulations	
	2.3.2	Hypotheses	

	3.1.1	Data vs. Information	
	3.1.1A	Iterative	
	3.1.1B	Filter	
	3.1.1C	Clustering	
	3.1.1C	Data classification	
	3.1.1E	Patterns	
	3.1.2A	Data-driven problems	
	3.1.2E	Online collaborative tools	
	3.1.3	Visualization(s) <i>see also</i> 3.1.3B	
	3.2.1	Extract(ing)	
Data and Information	3.2.1A	Large data set(s) <i>see also</i> 3.2.2	
	3.2.1B	Trend(s) <i>see also</i> 7.1.1G	
	3.2.1C	Computing tools (fusion tables, queries)	
	3.2.1D	Search tools	
	3.2.1E	Filter systems (filter tools)	
	3.2.1F	Spreadsheet/ database software	
	3.2.1G-I	Metadata	
	3.2.2E	Scalability	
	3.3.1A	Digital data representation	
	3.3.1B	Secure transmission concerns	
	3.3.1C	Lossy data compression	
	3.3.1D	Lossless data compression	
	3.3.1G	Data file formats	
	3.3.1F	Privacy concerns	
	3.3.1F	Security concerns	
	3.3.1H	Storage media	
Algorithms	4.1.1	Algorithm	
	4.1.1B	Sequencing	
	4.1.1C	Boolean condition	
	4.1.1C	Selection	
	4.1.1D	Iteration (repetition)	
	4.1.2A	Pseudo code	
	4.1.2B	Natural language	

	4.2.1	Runtime	
	4.2.1B	Reasonable time	
	4.2.2	Solvable problem	
	4.2.2	Unsolvable problem	
	4.2.2A	Heuristic solution(s)	
	4.2.3	Undecidable problem	
	4.2.3B	Decidable problem	
	4.2.4	Empirical analysis	
	4.2.4H	Linear search	
Programming	5.1.1	Program(ming)	
	5.1.2A	Iterative process	
	5.1.2C	Incremental development	
	5.1.2D-F	Program documentation	
	5.1.2G	Program development	
	5.1.3	Collaborative development	
	5.2.1B	Sequential execution	
	5.2.1C	Program instructions	
	5.2.1D-E	Program execution	
	5.2.1F-H	Process(es)	
	5.2.1I	Executable programs	
	5.3.1A-D	Procedure(s)	
	5.3.1E-G	Parameter(s)	
	5.3.1H	Data abstraction	
	5.3.1I	Strings; string operation	
	5.3.1I	Substring	
	5.3.1I	Concatenation	
	5.3.1J	Integers	
	5.3.1J	Floating-point numbers <i>see also 2.1.2C</i>	
	5.3.1K-L	Lists; List operations	
	5.3.1M-O	Application Program Interfaces (APIs); Libraries	
	5.4.1C	Self-identifying variables	
	5.4.1E	Debugging	
	5.4.1M	Functionality	

	5.4.1J	Program justification	
	5.4.1M	Functionality	
	5.5.1B	Integers	
	5.5.1C	Real numbers (floating-point)	
	5.3.1J	Arithmetic operators	
	5.5.1E	Logical concepts	
	5.5.1E	Boolean algebra	
		Syntax	
The Internet	6.1.1A	The Internet	
	6.1.1B	End-to-end architecture	
	6.1.1C	Devices	
	6.1.1C	Network(s)	
	6.1.1E	Internet Protocol (IP)	
	6.1.1G	Domain Name System (DNS) see also 6.3.1B	
	6.1.1G	IP address(es)	
	6.1.1H	IPv6	
	6.1.1I	HTTP/HTTPS	
	6.1.1I	SMTP	
	6.1.1I	IETF	
	6.2.1A	Hierarchy	
	6.2.1A	Redundancy	
	6.2.1B	Domain name syntax	
	6.2.1D	Routing	
	6.2.2A	Scalability	
	6.2.2B	Redundancy of routing	
	6.2.2D	Protocols (include TCP/IP)	
	6.2.2D	Interfaces	
	6.2.2E	Open Standards	
	6.2.2F	Packet switching	
	6.2.2G	TCP/IP	
	6.2.2H	Browser (s)	
	6.2.2H	Web server	
	6.2.2H	SSL/TLS	
	6.2.2J	Bandwidth	

	6.2.2K	Latency	
	6.3.1A	Trust model	
	6.3.1C	Cybersecurity	
	6.3.1D	Cyber warfare; cybercrime	
	6.3.1E	DDoS	
	6.3.1F	Phishing	
	6.3.1F	Viruses	
	6.3.1G	Antivirus software	
The Internet	6.3.1G	Firewall	
	6.3.1H-I	Cryptography	
	6.3.1K	Symmetric encryption	
	6.3.1L	Public key encryption	
	6.3.1M	Certificate authorities	
	6.3.1M	Digital certificate	
Global Impact	7.1.1	Computing innovation(s)	
	7.1.1A	Email; SMS; chat	
	7.1.1B	Video conferencing; video chat	
	7.1.1C	Social media <i>see also</i> 7.1.1H	
	7.1.1D	Cloud computing	
	7.1.1E	Dissemination <i>see also</i> 7.1.1H	
	7.1.1F	Public data	
	7.1.1I	GPS (global positioning system)	
	7.1.1J	Sensor networks	
	7.1.1K	“Smart” technologies	
	7.1.1M	Internet vs. WWW	
	7.1.1N	e-commerce	
	7.1.1O	Productivity	
	7.1.2A	Distributed solutions	
	7.1.2B	“Citizen science”	
	7.1.2C	Human computation	
	7.1.2D	Digital collaboration	
	7.1.2F	Crowdsourcing	
	7.1.2G	Mobile computing	

	7.2.1A	Machine learning	
	7.2.1A	Data mining	
	7.2.1B	Scientific computing	
	7.2.1D	Open Access	
Global Impact	7.2.1D	Creative Commons (CC) license	
	7.2.1F	Moore's Law	
	7.3.1B	Commercial access	
	7.3.1B	Download	
	7.3.1B	Streaming	
	7.3.1C	Peer-to-peer networks	
	7.3.1D	Authenticated access	
	7.3.1D	Anonymous access	
	7.3.1E	Censorship (of digital info)	
	7.3.1F	Open source software	
	7.3.1F	Licensing of software	
	7.3.1I	Anonymity	
	7.3.1I	Proxy servers	
	7.3.1L	Curation of information	
	7.3.1M	Targeted advertising	
	7.3.1N	Intellectual property	
	7.3.1O	Copyright	
	7.3.1P	Digital Millennium Copyright Act	
	7.4.1A	Innovation(s)	
	7.4.1B	Wireless	
	7.4.1D	"Digital divide"	
	7.4.1D	Socioeconomic	
	7.4.1E	Infrastructure	
	7.4.1E	Commercial	
	7.5.1A	Online databases/ Virtual libraries	
	7.5.1A	Primary source	
	7.5.1A	Secondary source	
	7.5.1C	Plagiarism	
	7.5.2A	Credibility of source(s)	

Abstraction All Around Us

Abstractions aren't just used in coding. Think about your daily life, identify and describe examples of procedural and data abstractions.

Procedural Abstraction

You are told "go clean your room". What does this phrase abstract?

The fire alarm goes off at school. What procedure does this sound evoke?

Data Abstraction

QR Codes:

Bar Codes:

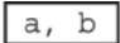
URLs:

Packet Trace

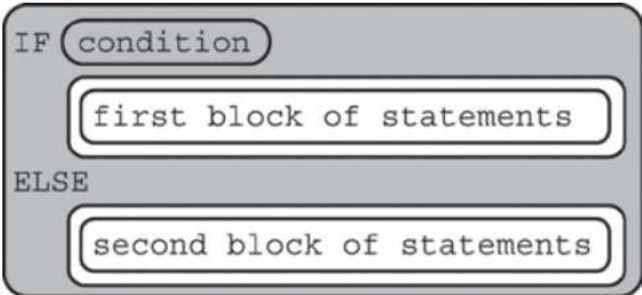
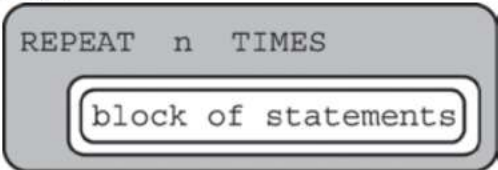
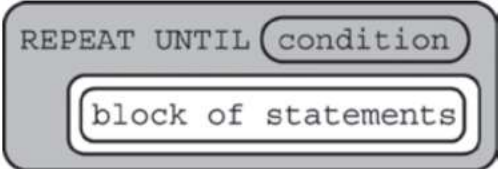
Directions: Watch [There and Back: A Packet's Trail](#) and [Warriors of the Net](#), detail below the **role and actions of each software and hardware component described** in a packet's route.

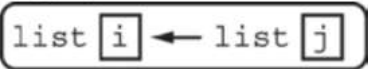
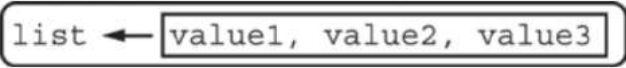
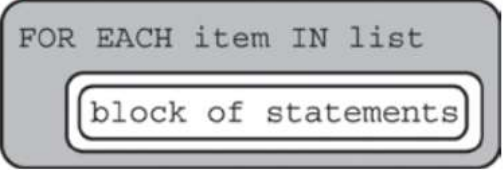
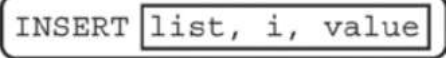
User Request	Browser	
	Link	
	Mail room (TCP/IP Stack)	
	Label	Sender - Receiver - Type -
	Address	
	LAN	
	Router	
	Corporate Intranet	
	Router Switch	
	Network Interface	
	Proxy server	
	Firewall	
	Router	
	The Internet	
	Satellite	
	Telephone	
	Transoceanic Cable	
Destination Location	Firewall	
	Server	
	Web server	
	Web server application	

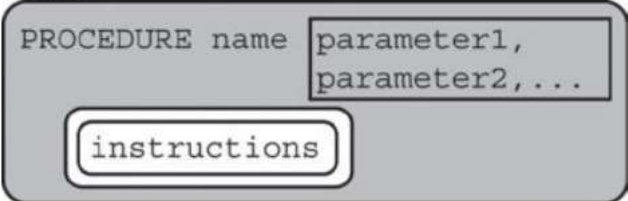
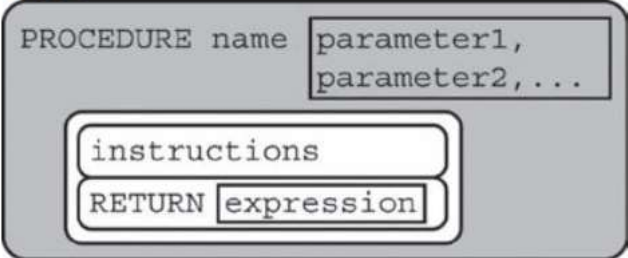
AP CSP Exam Reference Guide Categories

Instruction	Explanation
Assignment, Display, and Input	
Text: $a \leftarrow \text{expression}$ Block: 	Evaluates <code>expression</code> and assigns the result to the variable <code>a</code> .
Text: <code>DISPLAY (expression)</code> Block: 	Displays the value of <code>expression</code> , followed by a space.
Text: <code>INPUT ()</code> Block: <code>INPUT</code>	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 <code>+</code> , <code>-</code> , <code>*</code> , and <code>/</code> are used to perform arithmetic on <code>a</code> and <code>b</code> . For example, $3 / 2$ evaluates to <code>1.5</code> .
Text and Block: $a \text{ MOD } b$	Evaluates to the remainder when <code>a</code> is divided by <code>b</code> . Assume that <code>a</code> and <code>b</code> are positive integers. For example, $17 \text{ MOD } 5$ evaluates to <code>2</code> .
Text: <code>RANDOM (a, b)</code> Block: <code>RANDOM</code> 	Evaluates to a random integer from <code>a</code> to <code>b</code> , including <code>a</code> and <code>b</code> . For example, <code>RANDOM (1, 3)</code> could evaluate to <code>1</code> , <code>2</code> , or <code>3</code> .
Relational and Boolean Operators	
Text and Block: $a = b$ $a \neq b$ $a > b$ $a < b$ $a \geq b$ $a \leq b$	The relational operators <code>=</code> , <code>≠</code> , <code>></code> , <code><</code> , <code>≥</code> , and <code>≤</code> are used to test the relationship between two variables, expressions, or values. For example, $a = b$ evaluates to <code>true</code> if <code>a</code> and <code>b</code> are equal; otherwise, it evaluates to <code>false</code> .

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: <pre>IF (condition) { <first block of statements> } ELSE { <second block of statements> }</pre> Block: 	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: <pre>REPEAT n TIMES { <block of statements> }</pre> Block: 	The code in block of statements is executed n times.
Text: <pre>REPEAT UNTIL (condition) { <block of statements> }</pre> 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: <code>list[i]</code> Block: 	Refers to the element of <code>list</code> at index <code>i</code> . The first element of <code>list</code> is at index 1.
Text: <code>list[i] ← list[j]</code> Block: 	Assigns the value of <code>list[j]</code> to <code>list[i]</code> .
Text: <code>list ← [value1, value2, value3]</code> Block: 	Assigns <code>value1</code> , <code>value2</code> , and <code>value3</code> to <code>list[1]</code> , <code>list[2]</code> , and <code>list[3]</code> , respectively.
Text: <pre>FOR EACH item IN list { <block of statements> }</pre> Block: 	The variable <code>item</code> is assigned the value of each element of <code>list</code> sequentially, in order from the first element to the last element. The code in <code>block of statements</code> is executed once for each assignment of <code>item</code> .
Text: <code>INSERT (list, i, value)</code> Block: 	Any values in <code>list</code> at indices greater than or equal to <code>i</code> are shifted to the right. The length of <code>list</code> is increased by 1, and <code>value</code> is placed at index <code>i</code> in <code>list</code> .
Text: <code>APPEND (list, value)</code> Block: 	The length of <code>list</code> is increased by 1, and <code>value</code> is placed at the end of <code>list</code> .

Instruction	Explanation
List Operations (continued)	
Text: REMOVE (list, i) Block: 	Removes the item at index <code>i</code> in <code>list</code> and shifts to the left any values at indices greater than <code>i</code> . The length of <code>list</code> is decreased by 1.
Text: LENGTH (list) Block: 	Evaluates to the number of elements in <code>list</code> .
Procedures	
Text: PROCEDURE name (parameter1, parameter2, ...) { <instructions> } Block: 	A procedure, <code>name</code> , takes zero or more parameters. The procedure contains programming instructions.
Text: PROCEDURE name (parameter1, parameter2, ...) { <instructions> RETURN (expression) } Block: 	A procedure, <code>name</code> , takes zero or more parameters. The procedure contains programming instructions and returns the value of <code>expression</code> . The <code>RETURN</code> 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 <code>true</code> if there is an open square one square in the direction relative to where the robot is facing; otherwise evaluates to <code>false</code> . The value of <code>direction</code> can be <code>left</code> , <code>right</code> , <code>forward</code> , or <code>backward</code> .