Programming Strategically

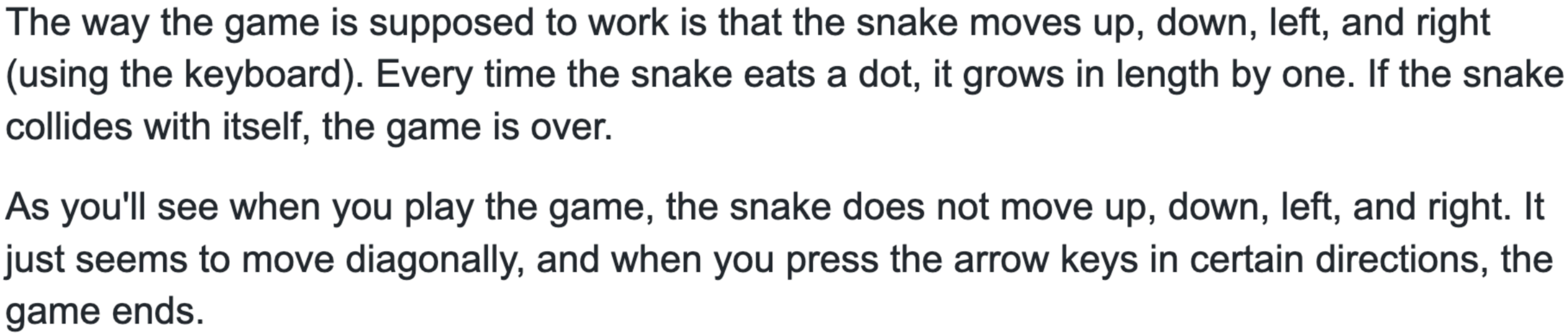
Thomas LaToza

https://cs.gmu.edu/~tlatoza/

@ThomasLaToza **@ThomasLaToza@hci.social**

Developer Experience Design Laboratory

George Mason University



The way the game is supposed to work is that the snake moves up, down, left, and right (using the keyboard). Every time the snake eats a dot, it grows in length by one. If the snake collides with itself, the game is over.

As you'll see when you play the game, the snake does not move up, down, left, and right. It just seems to move diagonally, and when you press the arrow keys in certain directions, the game ends.

```
Find an event immediately before the incorrect behavior
Trace control forwards, observing each statement until something incorrect happens
```

```
Find the statement that generated the incorrect output
Keep following the data used backwards until you find something that's wrong
```

guess and check

backwards search

forwards search

read the docs

check StackOverflow

ask a coworker

draw a whiteboard diagram

guess and check

backwards search

forwards search

read the docs

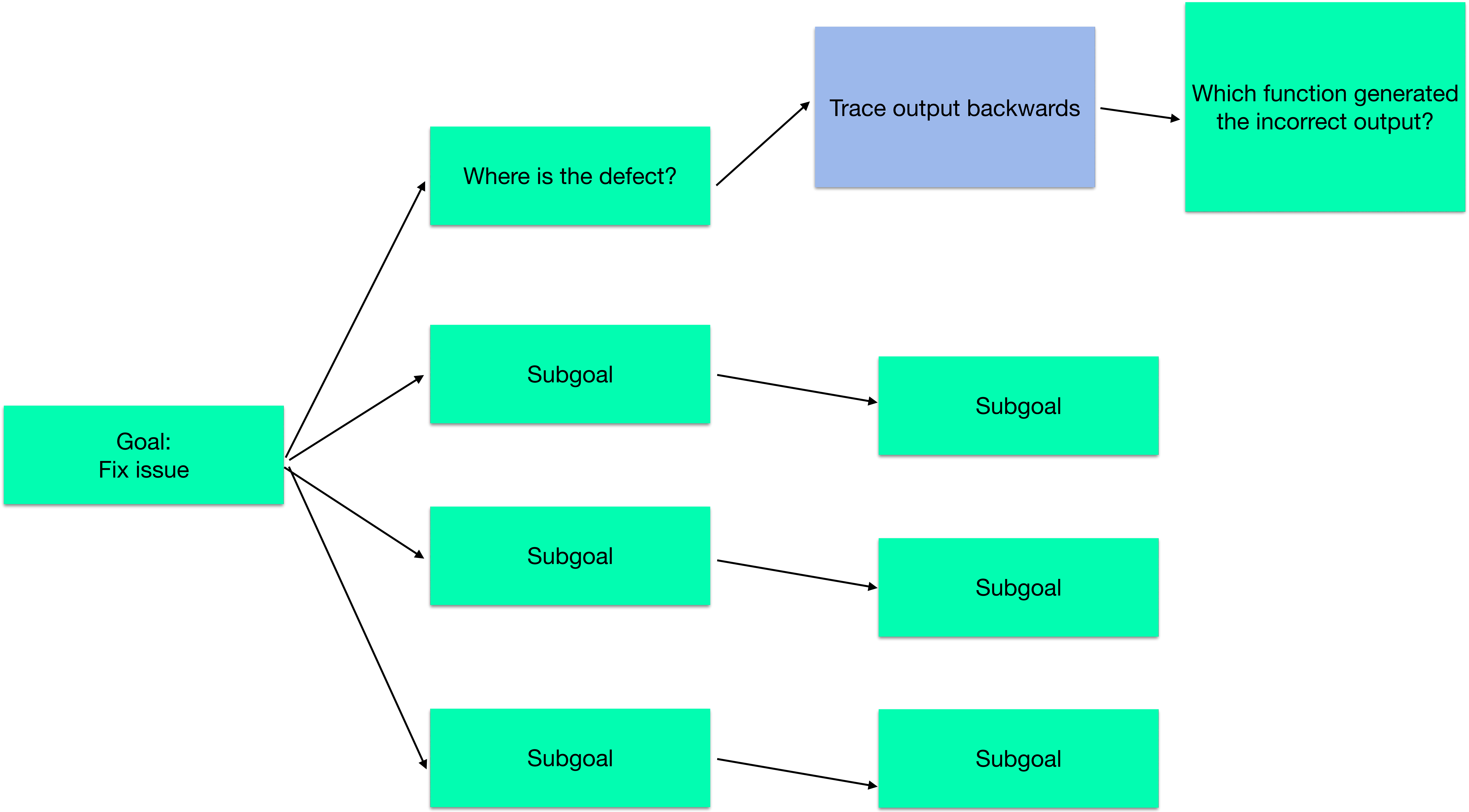
check StackOverflow

ask a coworker

draw a whiteboard diagram

programming strategy a procedure for accomplishing a programming task

Thomas D. LaToza, Maryam Arab, Dastyni Loksa, and Amy J. Ko. (2020). Explicit programming strategies. *Empirical Software Engineering (ESE),* 25, 2416–2449.



Goal:
Fix issue

Where is the defect?

Trace output backwards

Which function generated the incorrect output?

Subgoal

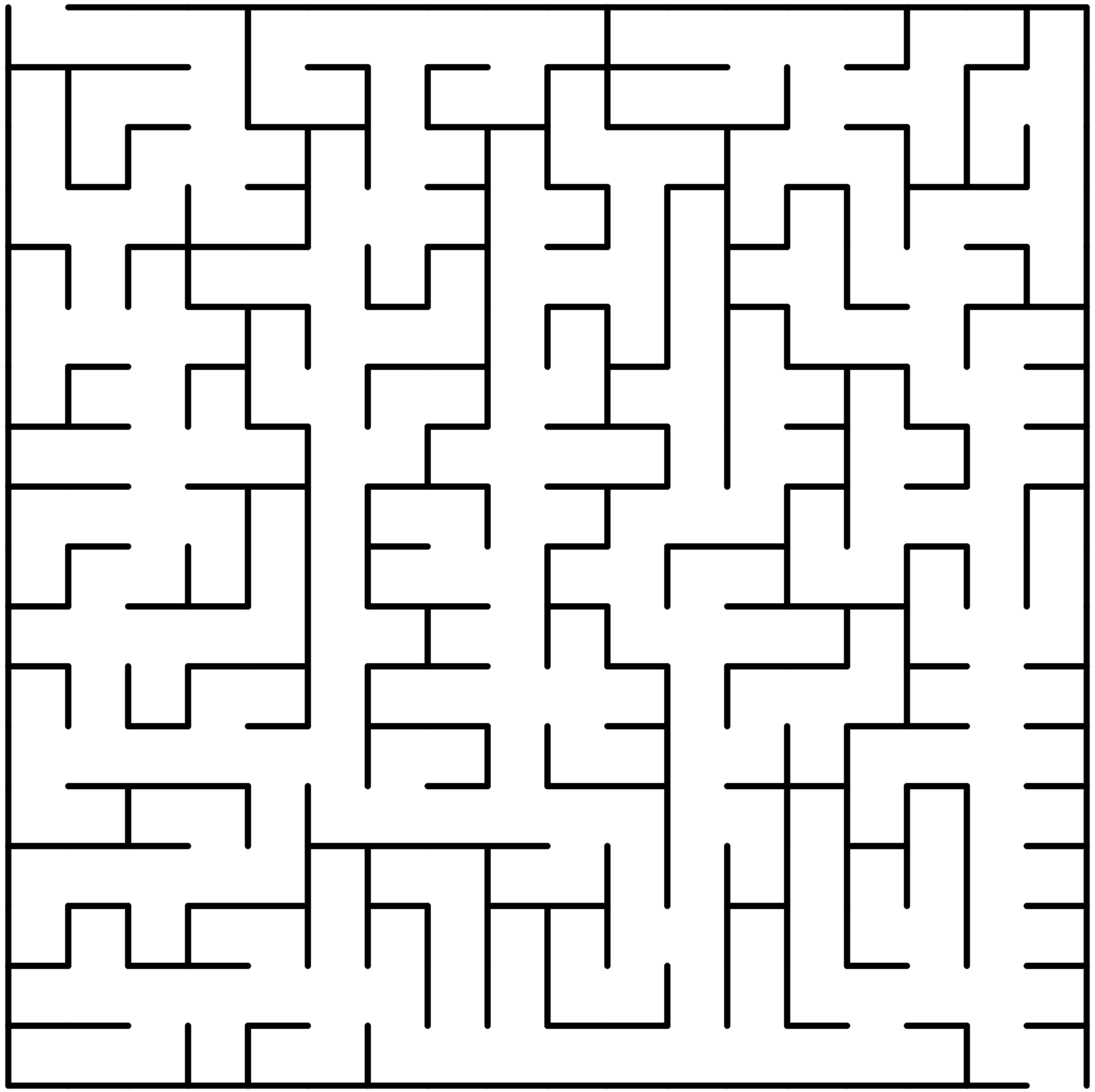
Subgoal

Subgoal

Subgoal

Subgoal

Subgoal



By Nandhp - Generated by the attached program, which was originally written by en:User:Cyp, who attached it to the image description page for an image generated by it on en.wikipedia. The image was licensed under CC-BY-SA-3.0/GFDL., CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=6916582

Herbert A. Simon. (1969). The Sciences of the Artificial. MIT Press.

programming strategy a procedure for accomplishing a programming task

Thomas D. LaToza, Maryam Arab, Dastyni Loksa, and Amy J. Ko. (2020). Explicit programming strategies. *Empirical Software Engineering (ESE),* 25, 2416–2449.

Developers work more systematically and efficiently when given effective explicit programming strategies

"Strategies determine success more than does the programmer's available knowledge"

"Experts seem to acquire a collection of strategies for performing programming tasks."

David J. Gilmore. Expert programming knowledge: A strategic approach. In Psychology of Programming. Elsevier, 223–234.

Amy J. Ko, Thomas D. LaToza, Stephen Hull, Ellen A. Ko, William Kwok, Jane Quichocho, Harshitha Akkaraju, and Rishin Pandit. 2019. Teaching Explicit Programming Strategies to Adolescents. In Technical Symposium on Computer Science Education (SIGCSE '19),469–475.

Thomas D. LaToza, Maryam Arab, Dastyni Loksa, and Amy J. Ko. (2020). Explicit programming strategies. *Empirical Software Engineering (ESE),* 25, 2416–2449.

better CSS
debugging strategy

Q: *Was function **F**’s implementation the ideal design, a hack, or accidental?*

Strategy for answering:

```
1. Begin procedure RetrieveRationaleFromCode
    a. Initialize an empty set of rationales R
    b. For each comment in the function:
        i. If the comment provides information about the rationale for the
           implementation
            1. Add the rationale to R
    c. If R is non-empty
        i. Synthesize the rationales in R into an answer to the question.
        ii. If you successfully synthesized the rationales
            1. Stop, you have an answer.
    d. This strategy failed. Begin procedure RetrieveRationaleFromDevelopers
2. Begin procedure RetrieveRationaleFromDevelopers
    a. Initialize an empty set of developers D
    b. Use version control (e.g., git blame) to identify the developers in the entire
       history of the function who wrote or modified code, adding each developer to D
    c. Use your organization’s default communication channels (e.g., email, IRC,
       Slack), writing a message to everyone in D asking Q
    d. Wait until:
        i. Someone in D responds with the answer, then stop, or
        ii. All in D respond without the answer, or
        iii. You cannot wait any longer.
    e. This strategy failed. Begin procedure InferRationaleFromCode.
3. Begin procedure InferRationaleFromCode
    a. Fully comprehend the behavior of F at the level of computation
    b. Infer the intraprocedural intent of F, understanding how F interacts with all of
       the functions that call it and all of the functions that it calls.
    c. Using the intraprocedural intent of F, infer the possible architectural intents of
       F.
    d. Estimate the likelihood of each possible architectural intents of F. Which intent
       is most likely given the intents of the intraprocedural intent of F and the
       architectural intent of the software?
    e. Select the intent with the highest likelihood, and stop.
    f. If you were unable to infer intents, this strategy failed.
```

Debugging Strategy
Not Secure | programmingstrategies.org/StrategyTracker/Stra...
Apps
Elms payment
NIW
LeetCode
Lexer and Parser
ESEC/FSE 2018
MIT Algorithms |
student-platform [~/Documents/Projects/StrategySharingPlatform/student-platform] - .../src/Components/Strategy/StrategyHandler/Strat...
www
Git:
student-platform
src
Components
Strategy
StrategyHandler
StrategyHandler.js
Layou.module.css
Layout.js
index.html
1: Project
Z: Structure
npm
2: Favorites
☑ Match Case
□ Words
□ Regex
StrategyHandler › retrieveFromLocalstorage() › updates › requiredKnowledge
6: TODO
9: Version Control
TypeScript 3.7.2
Terminal
Event Log
WebStorm 2019.3.5 available: // Update... (yesterday 10:07 PM)
17 chars
670:81
LF
UTF-8
4 spaces
Git: branch1

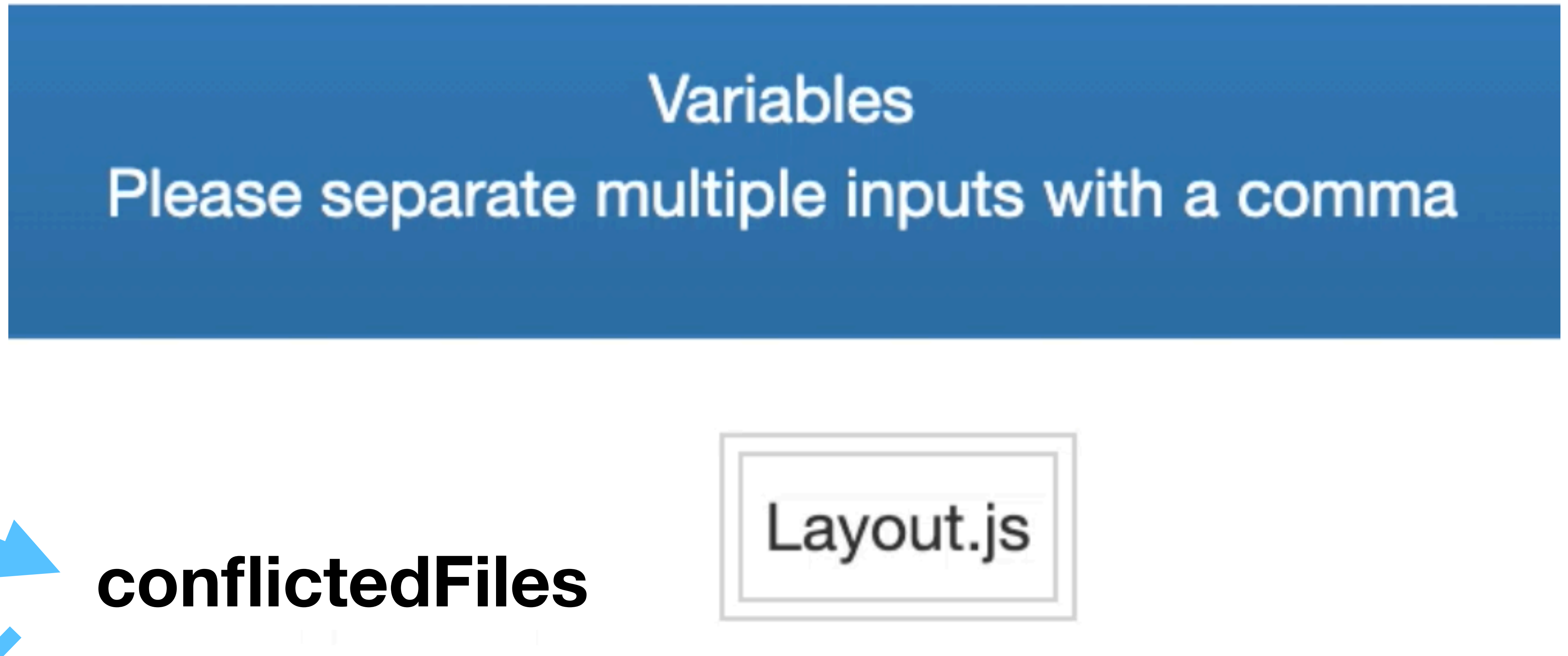
Please select your Strategy

```
                this.setState( state: {
                    strategyId: response.data,
                    status: "saved"
                })
                localStorage.clear();
            }).catch(error => {
            console.log(error);
        });
        }
    }

    updateKnowledge = (array) => {
        let uniqueItems = new Set();
        array.forEach((item) => {
            uniqueItems.add(item.toLowerCase())
        });

        this.setState( state: {
            requiredKnowledge: Array.from(uniqueItems)
        })
        localStorage.setItem("new_requiredKnowledge", JSON.stringify(Array.from(uniqueItems)));
    }

    updateTools = async (array) => {
        let uniqueItems = new Set();
        let formattedAllTools = {}

        Object.values(this.state.allTools).forEach((item) => {
            formattedAllTools[item.toLowerCase()] = item;
        })

        for (let i = 0; i < array.length; i++) {
            let item = array[i]
            let key = item.toLowerCase()

            if (formattedAllTools[key]) {
                uniqueItems.add(formattedAllTools[key])
            } else {
                let success = true
                try {
                    await axios.post( url: "/dataManagement/technologies",  data: {name: item})
```

```
STRATEGY :: strategy IDENTIFIER (IDENTIFIER+) STATEMENTS

STATEMENTS :: STATEMENT+

STATEMENT :: * (ACTION l CALL l CONDITIONAL l FOREACH l ASSIGNMENT l RETURN )+

ACTION :: (word l IDENTIFIER)+ .

CALL :: do identifier ( IDENTIFIER* )

CONDITIONAL :: if QUERY STATEMENTS

FOREACH :: for each IDENTIFIER in identifier STATEMENTS

UNTIL :: until QUERY STATEMENTS

ASSIGNMENT :: set IDENTIFIER to QUERY

RETURN :: return QUERY

QUERY :: (word l IDENTIFIER l CALL)+

IDENTIFIER :: ' identifier '
```

ASSIGNMENT :: set IDENTIFIER to QUERY

SET 'conflictedFiles' TO the project files that have a conflict



Variables

Please separate multiple inputs with a comma

conflictedFiles: Layout.js

FOREACH :: for each IDENTIFIER in identifier STATEMENTS

FOR EACH 'file' IN 'conflictedFiles'

```
# If you've spent a lot of time debugging unfamiliar code, the way that you probably debug is
# to first look at the failure, then look at the code to understand how it's architected, and
# then look for possible reasons for why the program failed. Once you have a guess, you
# probably then check it with things like breakpoints and logging. This strategy often works
# if you can have a lot of prior experience with debugging and inspecting program state. But
# if you don't have that experience, or you happen to guess wrong, this approach can lead to
# a lot of dead ends.
#
# The strategy you're about to use is different. Instead of guessing and checking, this
# strategy involves systematically working backwards from the code that directly caused the
# failed output to all of the code that caused that failed output to occur. As you work
# backwards, you'll check each statement for defects. If you work backwards like this,
# following the chain of causality from failure to cause, you will almost certainly find the
# bug.

STRATEGY debug()
 # This first step will give you enough familiarity to find lines in the program that create
 # the program's output. Read the names of all of the functions and variables in the program
 # Some programs produce command line output with print statements.
 # Is the faulty output you're investigating printed to a command line?
 IF the faulty output is logged to a command line
  # To find print statements, try searching for keywords related to 'log' or 'print'
  SET outputLines TO the line numbers of calls to console logging functions
 # Graphical output includes things like colored lines and rectangles
 IF the faulty output is graphical output
  # To find these lines, try searching for keywords related to graphical output, like '
  # draw' or 'fill'. Focus on lines that directly render something, not on higher-level
  # functions that indirectly call rendering functions.
  SET outputLines TO the line numbers of function calls that directly render graphics to
      the screen
 # Now that you have some lines that could have directly produced the faulty output, you're
 # going to check each line, see if it executed, and then find the cause of it executing. If
 # you're lucky, you only have one output line to check.
 FOR EACH 'line' IN 'outputLines'
  # First, let's make sure the line executed. You want to be sure that this is actually the
```

Strategy: Design task

		Self-guided	Guided
Template	Found and used example code as a template for implementation.	4/14 (29%)	0/14 (0%)
Decompose	Analyzed functional requirements for sub-problems, implementing each independently	9/14 (64%)	0/14 (0%)
TDD	Translated functional requirements into test cases, identifying sub-problems from test case requirements.	2/14 (14%)	11/14 (79%)

Strategy: Debugging task

		Self-guided	Guided
Guess & check	Participants found suspicious lines of code, modifying them and checking the effects of their modification.	4/14 (29%)	0/14 (0%)
Forward search	Participants identified where the program began processing input, following its execution forward	9/14 (64%)	0/14 (0%)
Backward search	Participants identified faulty output and worked backwards through control and data flow dependencies	2/14 (14%)	11/14 (79%)

Thomas D. LaToza, Maryam Arab, Dastyni Loksa, and Amy J. Ko. (2020). Explicit programming strategies. *Empirical Software Engineering (ESE),* 25, 2416–2449.

Design task

1.30 times more likely to make more progress

$p < 0.023^*$

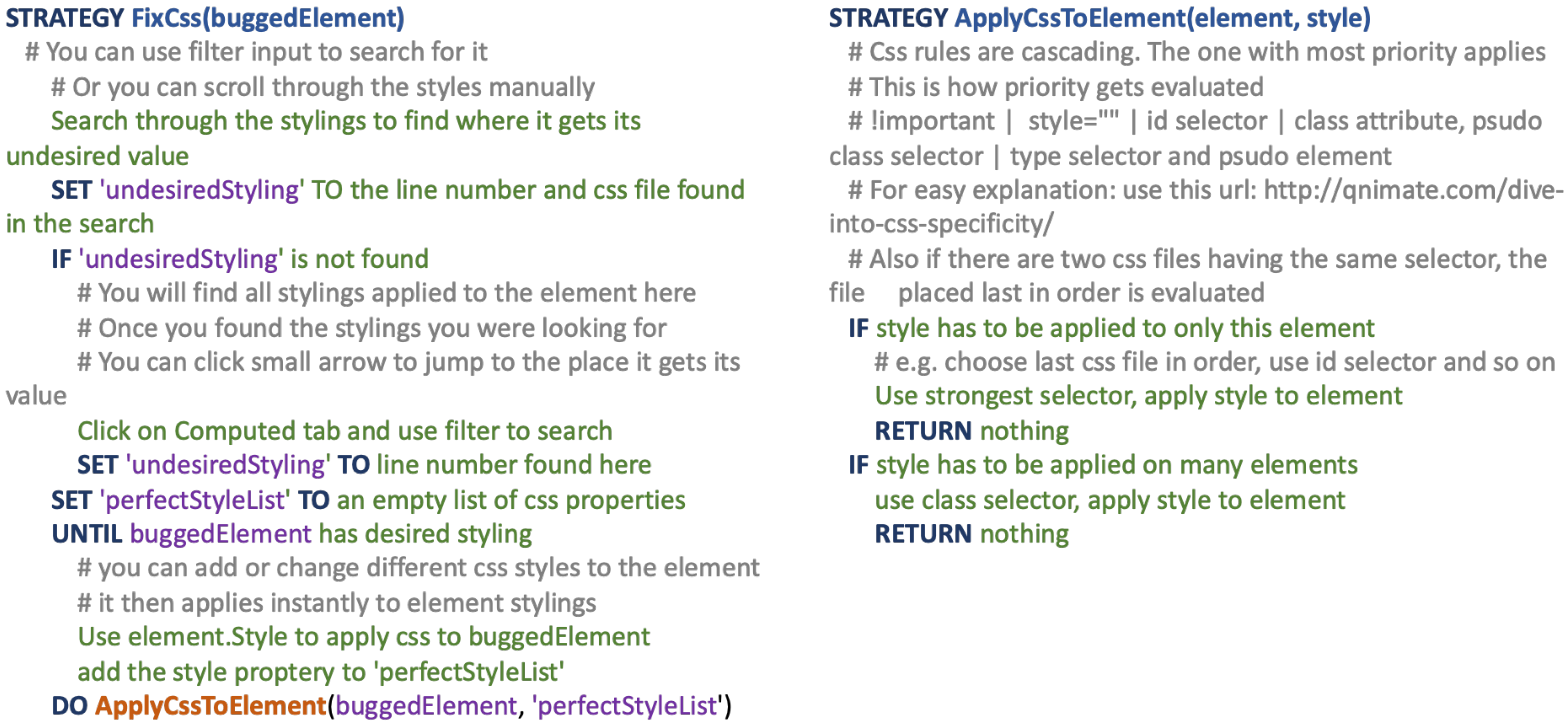
Debugging task

1.96 times more likely to make more progress

$p < 0.004^*$

Thomas D. LaToza, Maryam Arab, Dastyni Loksa, and Amy J. Ko. (2020). Explicit programming strategies. *Empirical Software Engineering (ESE),* 25, 2416–2449.

are programming strategies tacit?



```
STRATEGY FixCss(buggedElement)
  # You can use filter input to search for it
    # Or you can scroll through the styles manually
    Search through the stylings to find where it gets its undesired value
    SET 'undesiredStyling' TO the line number and css file found in the search
    IF 'undesiredStyling' is not found
      # You will find all stylings applied to the element here
      # Once you found the stylings you were looking for
      # You can click small arrow to jump to the place it gets its value
      Click on Computed tab and use filter to search
      SET 'undesiredStyling' TO line number found here
    SET 'perfectStyleList' TO an empty list of css properties
    UNTIL buggedElement has desired styling
      # you can add or change different css styles to the element
      # it then applies instantly to element stylings
      Use element.Style to apply css to buggedElement
      add the style proptery to 'perfectStyleList'
    DO ApplyCssToElement(buggedElement, 'perfectStyleList')
```

```
STRATEGY ApplyCssToElement(element, style)
  # Css rules are cascading. The one with most priority applies
  # This is how priority gets evaluated
  # !important |  style="" | id selector | class attribute, psudo class selector | type selector and psudo element
  # For easy explanation: use this url: http://qnimate.com/dive-into-css-specificity/
  # Also if there are two css files having the same selector, the file    placed last in order is evaluated
  IF style has to be applied to only this element
    # e.g. choose last css file in order, use id selector and so on
    Use strongest selector, apply style to element
    RETURN nothing
  IF style has to be applied on many elements
    use class selector, apply style to element
    RETURN nothing
```

Maryam Arab, Thomas D. LaToza, Jenny Liang, Amy J. Ko. (2022). An exploratory study of sharing strategic programming knowledge. Conference on Human Factors in Computing Systems (CHI), 1-15.

- Strategy-related
  - Generality
  - Ambiguity
  - Imprecise steps
  - Required tool use

> *“ I used chrome but still I was not able to find the NET section to find the CSS component. It took me a long time to find the component.*

- Mismatch between the level of knowledge assumed by the strategy and possessed by the user

Maryam Arab, Thomas D. LaToza, Jenny Liang, Amy J. Ko. (2022). An exploratory study of sharing strategic programming knowledge. Conference on Human Factors in Computing Systems (CHI), 1-15.

code interacting with framework

	search online forum	create diagrams
likelihood (odds ratio)	3.84 (3.84x more likely)	0.51 (0.51x less likely)

Cassandra Bailey. The Impact of Affect, Scenario and Task Characteristics on Developer Decision-Making. (2020). Masters Thesis, George Mason University.

feeling stressed / nervous (LVHA)

	add print statements	read surrounding code
likelihood (odds ratio)	2.42 (2.42x more likely)	0.17 (0.17x less likely)

Cassandra Bailey. The Impact of Affect, Scenario and Task Characteristics on Developer Decision-Making. (2020). Masters Thesis, George Mason University.

feeling sad / depressed (LVLA)

	experiment with edits
likelihood (odds ratio)	0.09 (0.09x less likely)

Cassandra Bailey. The Impact of Affect, Scenario and Task Characteristics on Developer Decision-Making. (2020). Masters Thesis, George Mason University.

feeling excited / enthusiastic (HVHA)

ask for help from a colleague

likelihood
(odds ratio)

2.13

(2.13x more likely)

Cassandra Bailey. The Impact of Affect, Scenario and Task Characteristics on Developer Decision-Making. (2020). Masters Thesis, George Mason University.

Takeaways

be more effective with

metacognition be aware of your problem solving process

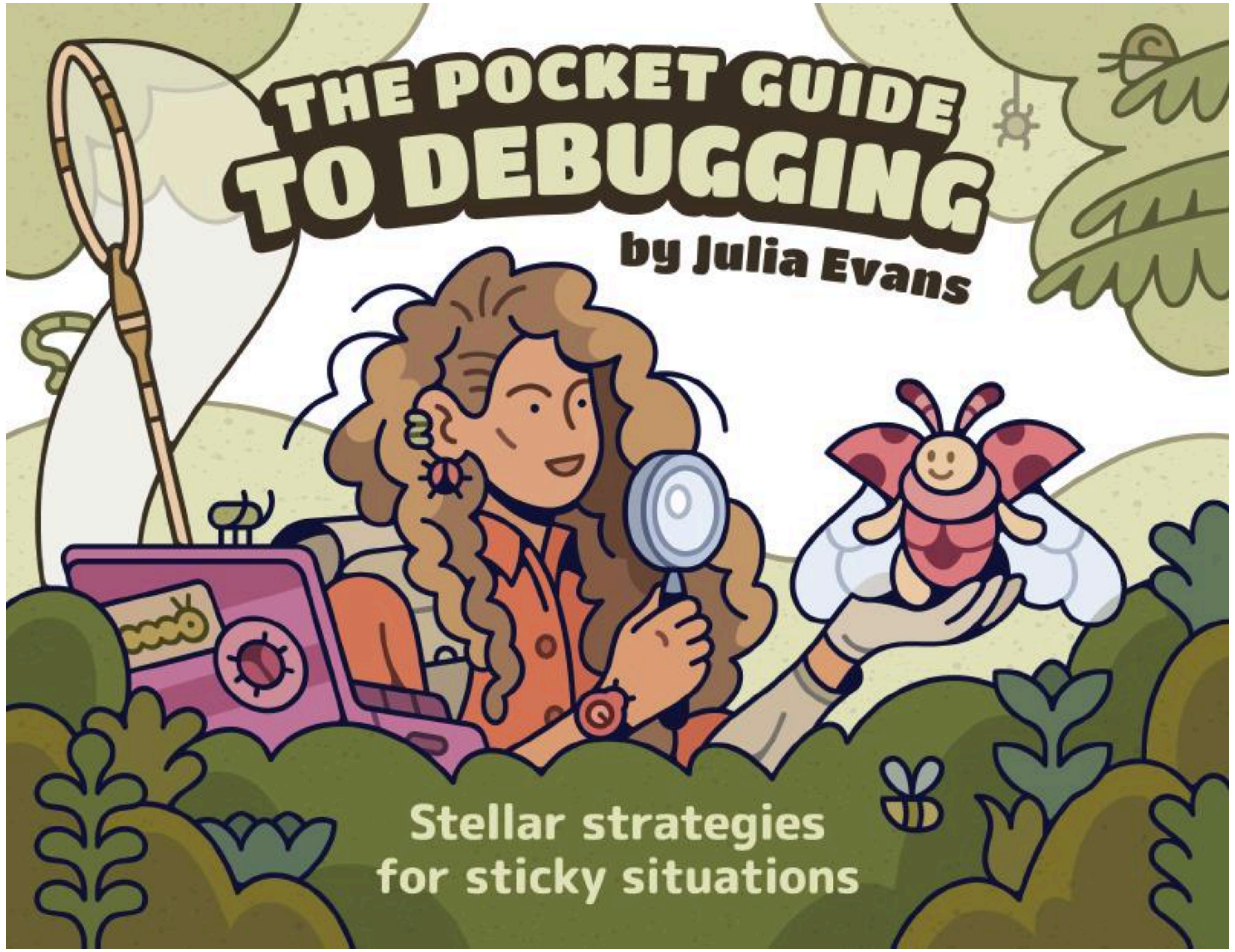
be more effective with

self-regulation monitor progress and use of time

(Robillard et al. 2004; Falkner et al. 2014)

be more effective with

better strategies


THE POCKET GUIDE TO DEBUGGING
by Julia Evans
Stellar strategies for sticky situations

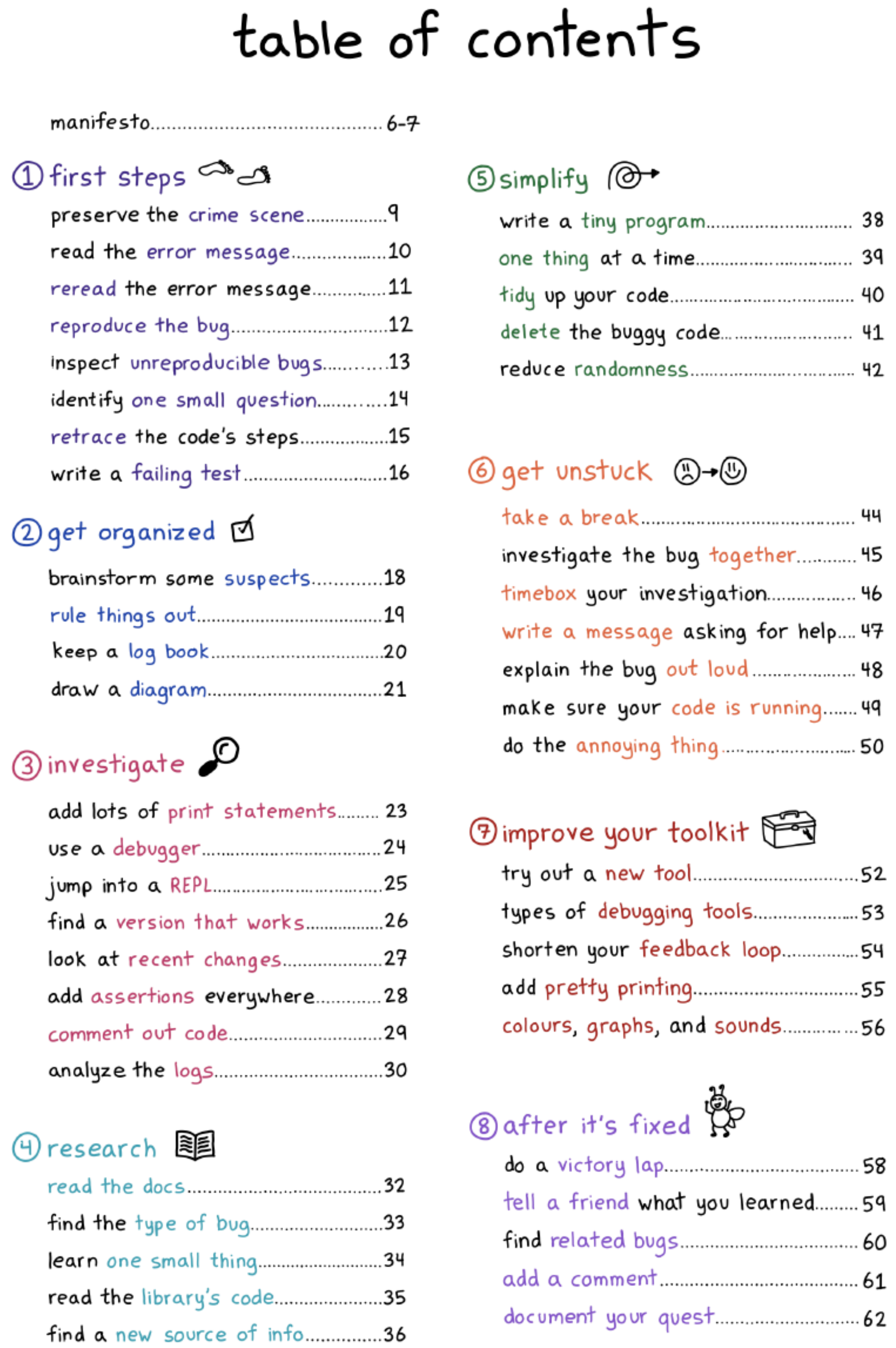
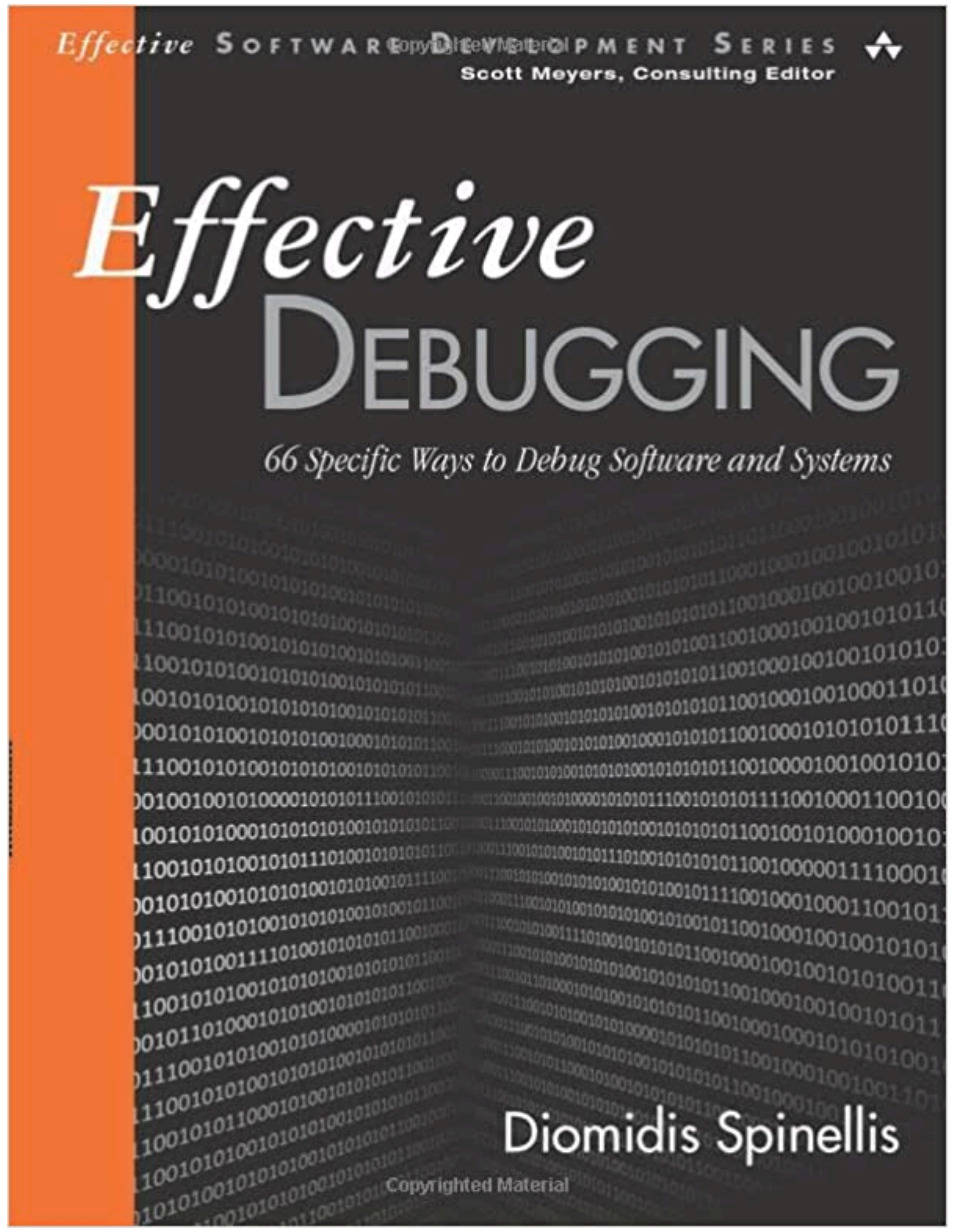
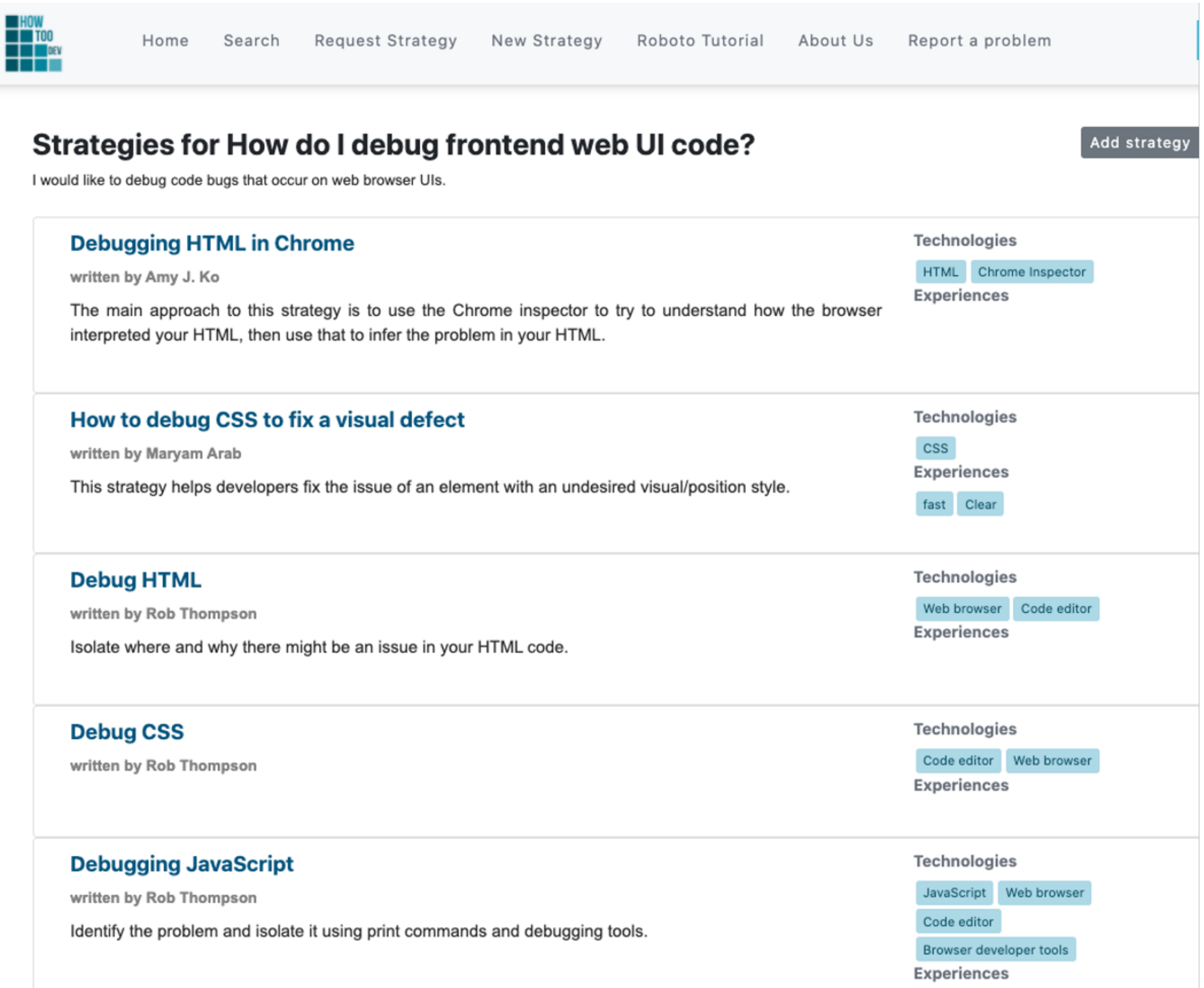


table of contents


Effective Software Development Series
Copyrighted Material
Scott Meyers, Consulting Editor
Effective Debugging
66 Specific Ways to Debug Software and Systems
Diomidis Spinellis
Copyrighted Material

be more effective with

sharing strategies



Strategies for How do I debug frontend web UI code?

I would like to debug code bugs that occur on web browser UIs.

Debugging HTML in Chrome

written by Amy J. Ko

The main approach to this strategy is to use the Chrome inspector to try to understand how the browser interpreted your HTML, then use that to infer the problem in your HTML.

Technologies: HTML, Chrome Inspector

Experiences

How to debug CSS to fix a visual defect

written by Maryam Arab

This strategy helps developers fix the issue of an element with an undesired visual/position style.

Technologies: CSS

Experiences: fast, Clear

Debug HTML

written by Rob Thompson

Isolate where and why there might be an issue in your HTML code.

Technologies: Web browser, Code editor

Experiences

Debug CSS

written by Rob Thompson

Technologies: Code editor, Web browser

Experiences

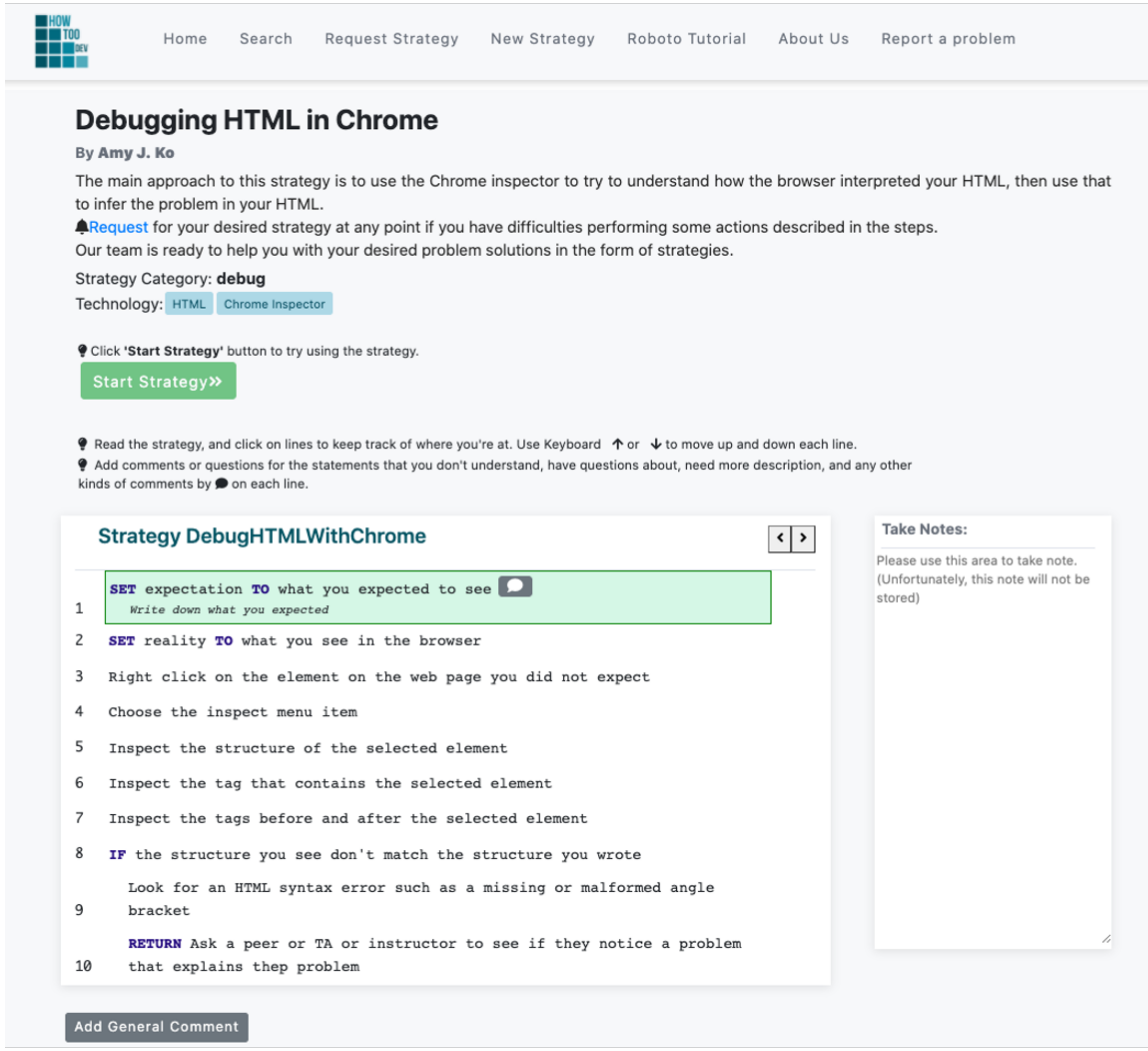
Debugging JavaScript

written by Rob Thompson

Identify the problem and isolate it using print commands and debugging tools.

Technologies: JavaScript, Web browser, Code editor, Browser developer tools

Experiences



Debugging HTML in Chrome

By Amy J. Ko

The main approach to this strategy is to use the Chrome inspector to try to understand how the browser interpreted your HTML, then use that to infer the problem in your HTML.

🔔Request for your desired strategy at any point if you have difficulties performing some actions described in the steps.

Our team is ready to help you with your desired problem solutions in the form of strategies.

Strategy Category: **debug**

Technology: HTML, Chrome Inspector

Click **'Start Strategy'** button to try using the strategy.

Read the strategy, and click on lines to keep track of where you're at. Use Keyboard ↑ or ↓ to move up and down each line.

Add comments or questions for the statements that you don't understand, have questions about, need more description, and any other kinds of comments by 💬 on each line.

Strategy DebugHTMLWithChrome

```
SET expectation TO what you expected to see
  Write down what you expected
SET reality TO what you see in the browser
Right click on the element on the web page you did not expect
Choose the inspect menu item
Inspect the structure of the selected element
Inspect the tag that contains the selected element
Inspect the tags before and after the selected element
IF the structure you see don't match the structure you wrote
  Look for an HTML syntax error such as a missing or malformed angle bracket
  RETURN Ask a peer or TA or instructor to see if they notice a problem that explains thep problem
```

Take Notes:

Please use this area to take note. (Unfortunately, this note will not be stored)

be aware of impact of how you feel

feeling stressed / nervous (LVHA)

feeling sad / depressed (LVLA)

feeling excited / enthusiastic (HVHA)

participate in a programming strategies mentoring session

email tlatoza@gmu.edu

Programming Strategically

Thomas LaToza

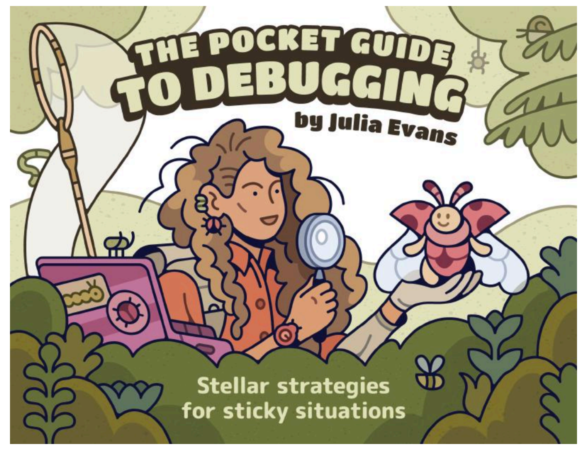
tlatoza@gmu.edu
https://cs.gmu.edu/~tlatoza/

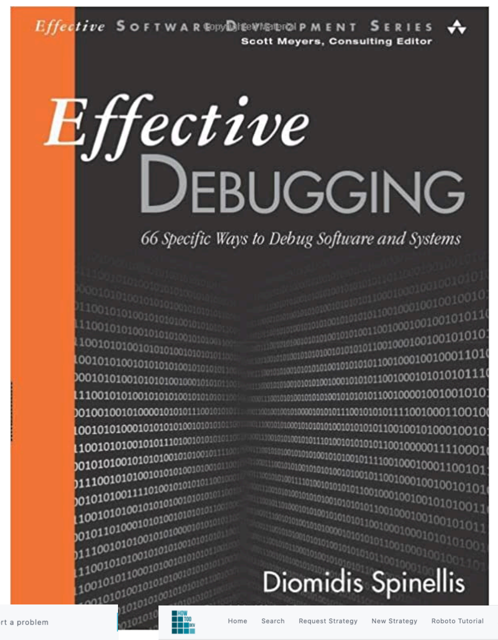
@ThomasLaToza **@ThomasLaToza@hci.social**

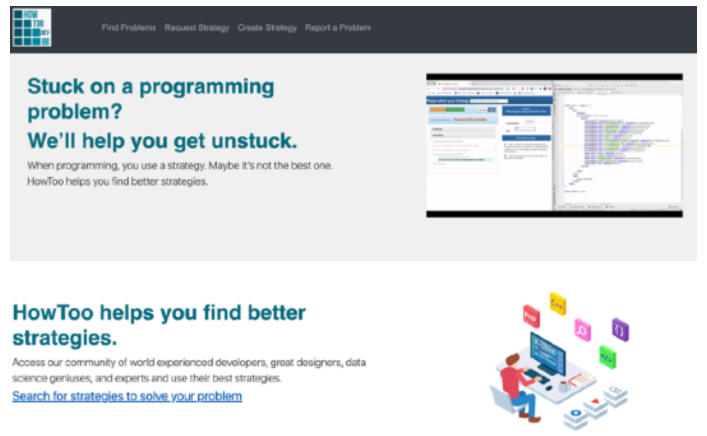
metacognition be aware of your problem solving process

self-regulation monitor progress and use of time

better strategies


THE POCKET GUIDE TO DEBUGGING
by Julia Evans
Stellar strategies for sticky situations


Effective Software Development Series
Scott Meyers, Consulting Editor
Effective Debugging
66 Specific Ways to Debug Software and Systems
Diomidis Spinellis



Stuck on a programming problem?
We'll help you get unstuck.

When programming, you use a strategy. Maybe it's not the best one. HowToo helps you find better strategies.

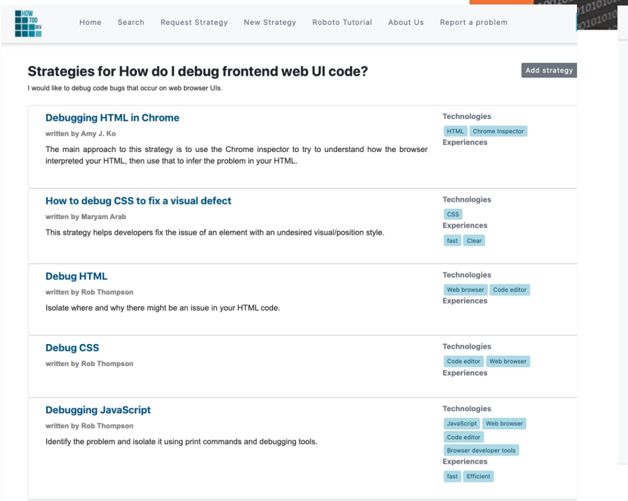
HowToo helps you find better strategies.

Access our community of world experienced developers, great designers, data science geniuses, and experts and use their best strategies.

Search for strategies to solve your problem

https://howtoo.herokuapp.com/

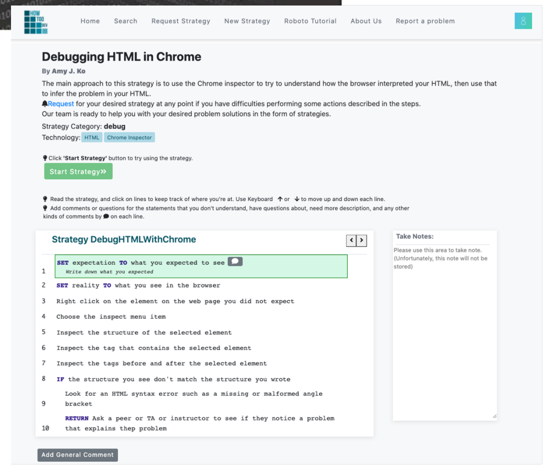
sharing strategies



Strategies for How do I debug frontend web UI code?

I would like to debug code bugs that occur on web browser UIs.

Debugging HTML in Chrome written by Amy J. Ko The main approach to this strategy is to use the Chrome inspector to try to understand how the browser interpreted your HTML, then use that to infer the problem in your HTML.	Technologies HTML Chrome Inspector Experiences
How to debug CSS to fix a visual defect written by Maryam Arab This strategy helps developers fix the issue of an element with an undesired visual/position style.	Technologies CSS Experiences fast Clear
Debug HTML written by Rob Thompson Isolate where and why there might be an issue in your HTML code.	Technologies Web browser Code editor Experiences
Debug CSS written by Rob Thompson	Technologies Code editor Web browser Experiences
Debugging JavaScript written by Rob Thompson Identify the problem and isolate it using print commands and debugging tools.	Technologies JavaScript Web browser Code editor Browser developer tools Experiences fast Efficient



Debugging HTML in Chrome

By Amy J. Ko

The main approach to this strategy is to use the Chrome inspector to try to understand how the browser interpreted your HTML, then use that to infer the problem in your HTML.

Request for your desired strategy at any point if you have difficulties performing some actions described in the steps.
Our team is ready to help you with your desired problem solutions in the form of strategies.

Strategy Category: **debug**

Technology: HTML Chrome Inspector

Click **'Start Strategy'** button to try using the strategy.

Read the strategy, and click on lines to keep track of where you're at. Use Keyboard ↑ or ↓ to move up and down each line.

Add comments or questions for the statements that you don't understand, have questions about, need more description, and any other kinds of comments by 💬 on each line.

Strategy DebugHTMLWithChrome

```
SET expectation TO what you expected to see
  Write down what you expected
SET reality TO what you see in the browser
Right click on the element on the web page you did not expect
Choose the inspect menu item
Inspect the structure of the selected element
Inspect the tag that contains the selected element
Inspect the tags before and after the selected element
IF the structure you see don't match the structure you wrote
  Look for an HTML syntax error such as a missing or malformed angle
  bracket
  RETURN Ask a peer or TA or instructor to see if they notice a problem
  that explains thep problem
```

Take Notes:

Please use this area to take note. (Unfortunately, this note will not be stored)

affect be aware of impact of how you feel