Microsoft Mentor

Training Sessions

Welcome back!

Margaret Price
A founder of Inclusive Design;
Strategist, higher education lead

Our sponsors:

CVP, Julia Liuson
CVP, Amanda Silver
VP, Kate Behncken
GM, Jennifer Ritzinger

Audrey St. John, PhD
Associate Professor of Computer Science
Mount Holyoke College

Becky Wai-Ling Packard, PhD
Professor of Psychology and Education
Mount Holyoke College

Heather Pon-Barry, PhD
Associate Professor of Computer Science
Mount Holyoke College

Shani Mensing
Design Mentor and Technical Specialist
Mount Holyoke College

Ground rules

- Be **open-minded**.
- Be willing to be **uncomfortable**.
- Speak from your own **perspective**.
- Be **patient** and allow space for all to learn.
- Facilitation means **interruption**
- What’s said here stays here; what’s **learned** can be shared.

Remember

- Rename yourself
- Group work: intros and pronouns
- Chat usage
- Breakout support

The program

The program

7 weeks

~2 hours
a week

Prep and logistics

Weekly group session

Week 1

Week 2

Week 3

Week 4

Week 5

Week 6

Week 7

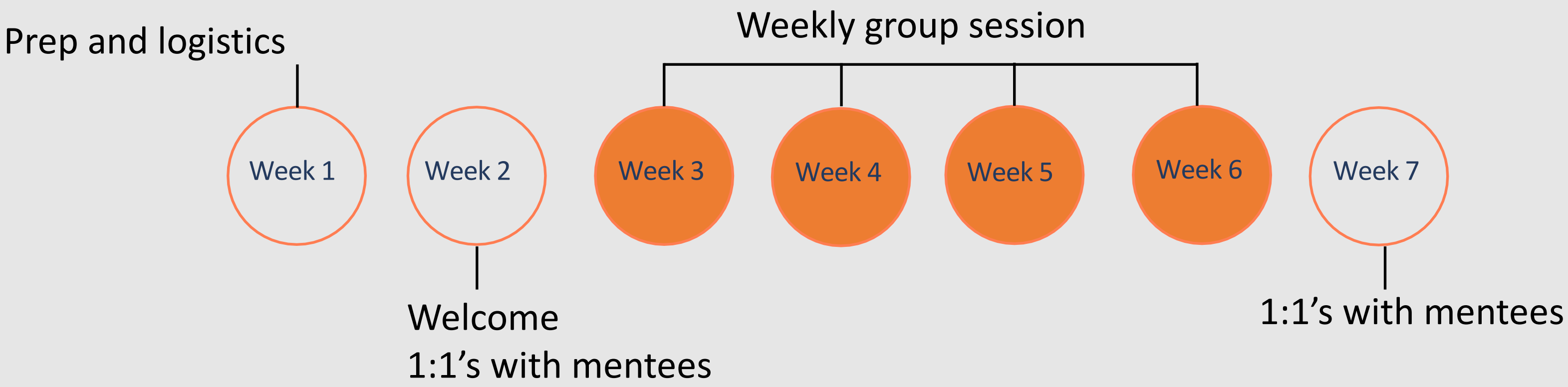
Welcome
1:1's with mentees

1:1's with mentees

The program

7 weeks

~2 hours a week



Session structure

Lead in

Learning topic 1 (ex: self-efficacy)
mentor story, video, breakout, report back

CS Challenge
mentor lead-in, breakout, report back

Learning topic 2 (ex: growth mindset)
mentor story, video, round robin

Open conversation

Weekly group session

- Session 2 — Can I hack it?
- Session 3 — How do I find my way?
- Session 4 — Can I be strategic?
- Session 5 — Can I pivot?

Session structure

Lead in

Learning topic 1 (ex: self-efficacy)
mentor story, video, breakout, report back

CS Challenge
mentor lead-in, breakout, report back

Learning topic 2 (ex: growth mindset)
mentor story, video, round robin

Open conversation

Weekly group session

- Session 2 — Can I hack it?
- Session 3 — How do I find my way?
- Session 4 — Can I be strategic?
- Session 5 — Can I pivot?

Session structure

Lead in

Learning topic 1 (ex: self-efficacy)
mentor story, video, breakout, report back

CS Challenge
mentor lead-in, breakout, report back

Learning topic 2 (ex: growth mindset)
mentor story, video, round robin

Open conversation

Weekly group session

- Session 2 — Can I hack it?
- Session 3 — How do I find my way?
- Session 4 — Can I be strategic?
- Session 5 — Can I pivot?

Weekly group session

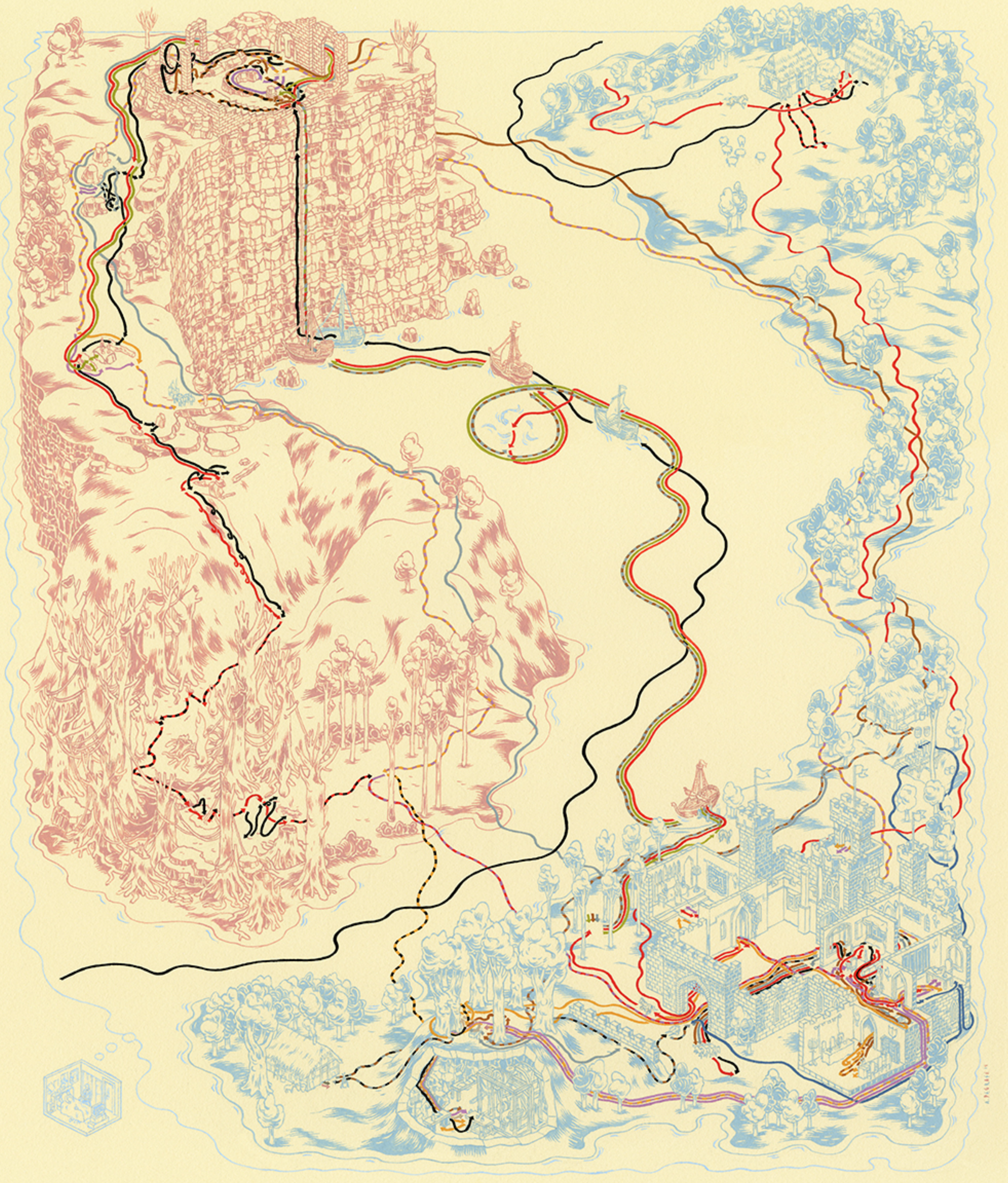
Session 2

Session 3

Session 4

Session 5

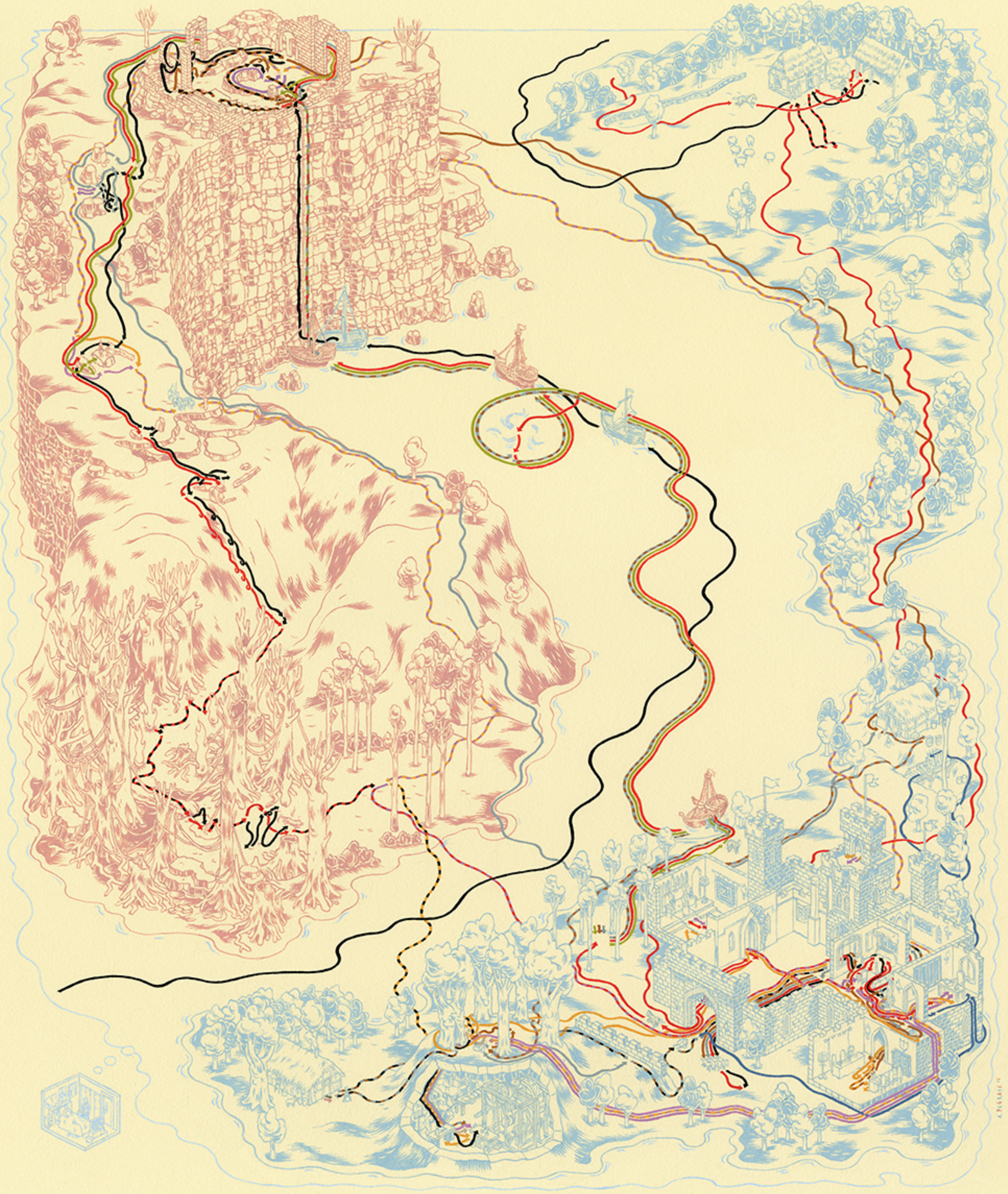
CS Challenge



Session 2: the challenge

“Please, I need help with my latest archaeological discovery! I’ve mapped the site, but need to create a detailed blueprint of each building. If I sketch the walls for a building, can you tell me where to place LiDAR scanners? Time is of the essence, so make sure you’re optimizing their placement!”

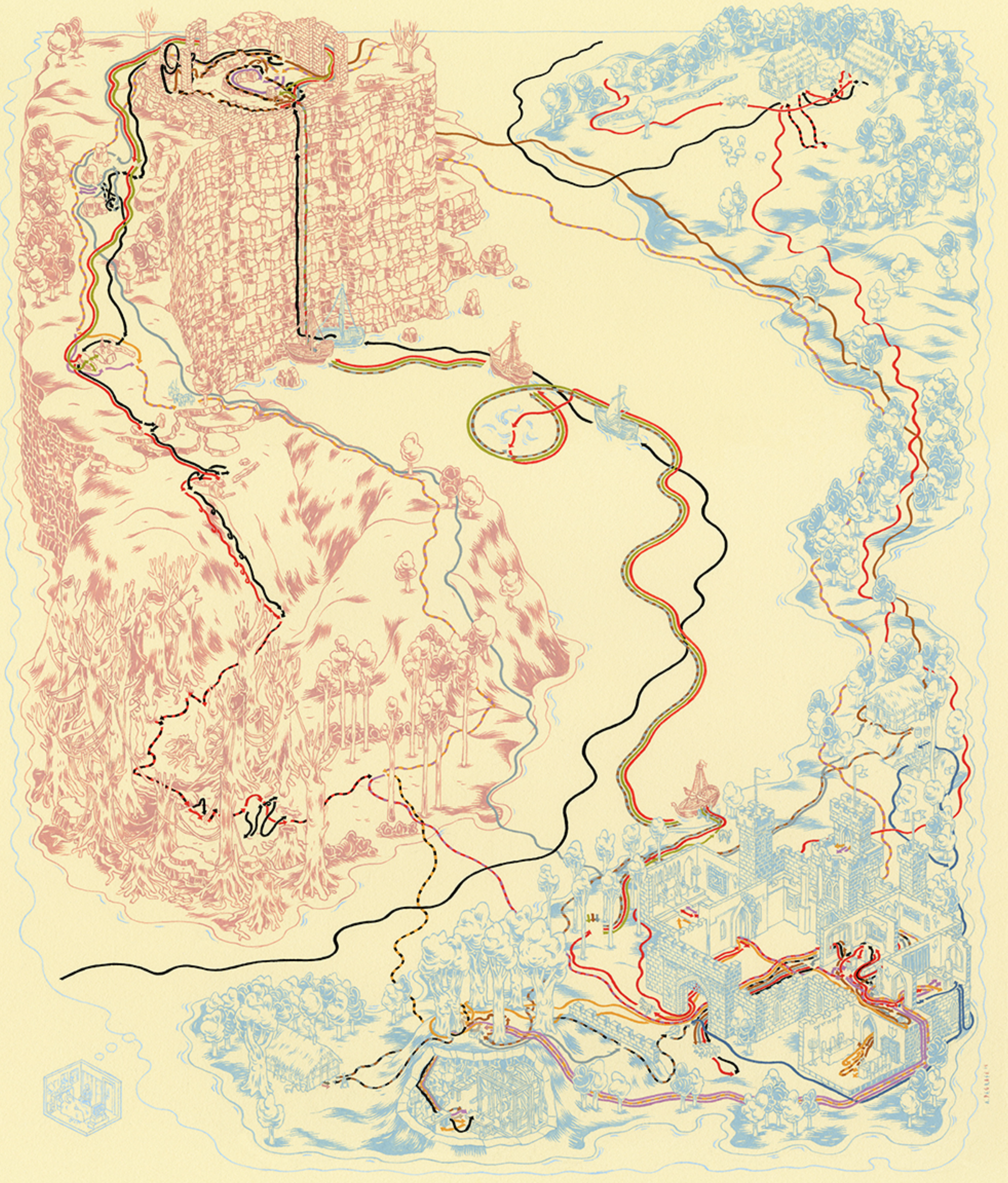
Signed,
an Amazing Archaeologist



Session 2: word problem discomfort

“Please, I need help with my latest archaeological discovery! I’ve mapped the site, but need to create a detailed blueprint of each building. If I sketch the walls for a building, can you tell me where to place LiDAR scanners? Time is of the essence, so make sure you’re optimizing their placement!”

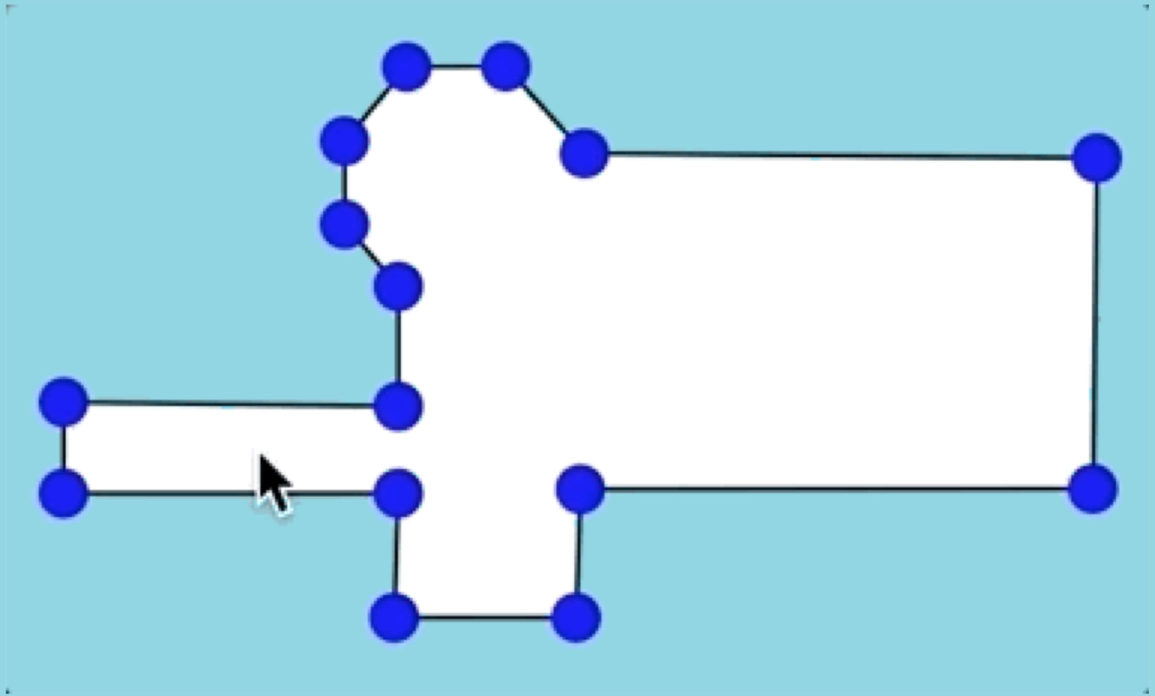
Signed,
an Amazing Archaeologist



Session 2: word problem discomfort

"Please, I need help with my latest archaeological discovery! I've mapped the site, but need to create a detailed **blueprint** of each building. If I **sketch** the walls for a building, can you tell me where to place **LiDAR scanners**? Time is of the essence, so make sure you're **optimizing** their placement!"

Signed,
an Amazing Archaeologist



Session 2: word problem discomfort

"Please, I need help with my latest archaeological discovery! I've mapped the site, but need to create a detailed **blueprint** of each building. If I **sketch** the walls for a building, can you tell me where to place **LiDAR scanners**? Time is of the essence, so make sure you're **optimizing** their placement!"

Signed,
an Amazing Archaeologist

Session structure

Lead in

Learning topic 1 (ex: self-efficacy)
mentor story, video, breakout, report back

CS Challenge
mentor lead-in, breakout, report back

Learning topic 2 (ex: growth mindset)
mentor story, video, round robin

Open conversation

Session 2: code discomfort

```python
# given a polygon as a list of points,
# where each is a list of (x,y) coordinates,
# print the polygon to the command line
def printPolygon( polygon ):
    # create a variable to hold the result
    # starting with the empty string
    printPoly = ""

    # for each point in the polygon
    for (x,y) in polygon:
        # make a string variable for (x,y)
        printPt = "(" + str(x) + "," + str(y) + ")"

        # "glue" it onto the big string
        printPoly = printPoly + printPt + " "
    # print out the final result
    print( printPoly )

# make a small list for a triangle
myPolygon = ((0,0),(1,1),(1,0))

# print the polygon coordinates
printPolygon( myPolygon )
```

```python
# given a polygon as a list of points,
# where each is a list of (x,y) coordinates,
# print the polygon to the command line
def printPolygon( polygon )
    # create a variable to hold the result
    # starting with the empty string
    printPoly = "

    # for each point in the polygon
    for (x,y) in polygon:
        # make a string variable for (x,y)
        printPt == "(" + str(x) "," + str(y) + ")"

        # "glue" it onto the big string
    printPoly = printPoly + printPt + " "
    # print out the final result
    print( printPoly )

# make a small list for a triangle
myPolygon = ((0,0),(1,1,(1,0))

# print the polygon coordinates
printPolygon( mypolygon )
```

spot the differences

```java
    /**
     * given a polygon as a 2D array of coordinates
     * print the polygon to the command line
     */
    public static void printPolygon( int[][] polygon ) {
        // create a variable to hold the result
        // starting with the empty string
        String printPoly = "";

        // for each point in the polygon
        for (int i = 0; i < polygon.length; i++ ) {
            int x = polygon[i][0]; // assume x comes first
            int y = polygon[i][1]; // and y comes second

            // make a string variable for (x,y)
            String printPt = "(" + Integer.toString(x) + ","
                             + Integer.toString(y) + ")";

            // "glue" it onto the big string
            printPoly = printPoly + printPt + " ";
        }
        // print out the final result
        System.out.println( printPoly );
    }

    public static void main( String[] args ) {
        // make a small list for a triangle
        int[][] myPolygon = {{0,0},{1,1},{1,0}};

        // print the polygon coordinates
        printPolygon( myPolygon );
    }
}
```

```java
    /**
     * given a polygon as a 2D array of coordinates
     * print the polygon to the command line
     */
    public static void printPolygon( int[] polygon ) {
        // create a variable to hold the result
        // starting with the empty string
        String printPoly = "";

        // for each point in the polygon
        for (int i = 0; i < polygon.length; i+ ) {
            int x = polygon[i][0]; // assume x comes first
            int y = polygon[i][1] // and y comes second

            // make a string variable for (x,y)
            printPt = "(" + Integer.toString(x) + ","
                             + Integer.toString(y) + ")";

            // "glue" it onto the big string
            printPoly = printPoly + printPt + " ";

        // print out the final result
        System.out.println( printPoly );
    }

    public static void main( string[] args ) {
        // make a small list for a triangle
        int[][] myPolygon = {(0,0},{1,1},{1,0}};

        // print the polygon coordinates
        printPolygon( myPolygon );
    }
}
```

Session structure

Session 3: tech tools discomfort

Lead in

Learning topic 1 (ex: self effi

mentor story, vide

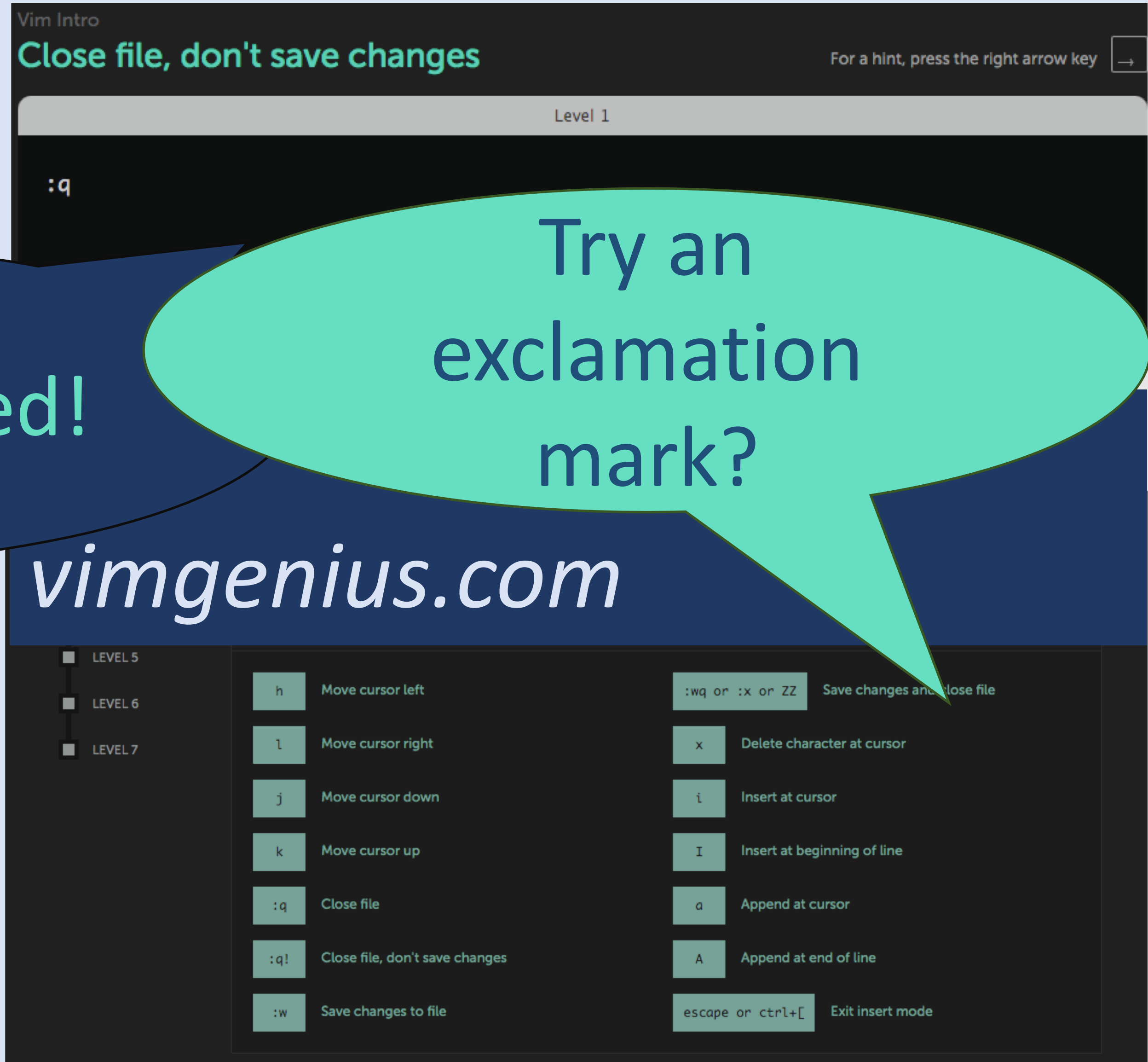
CS Challenge

mentor lead-in, break

Help! I’m trapped!

Try an exclamation mark?

vimgenius.com



Learning topic 2 (ex: growth mindset)

mentor story, video, round robin

Open conversation

Session 4

Can I be strategic?

Session structure

Lead in

Learning topic 1 (ex: self-efficacy)
mentor story, video, breakout, report back

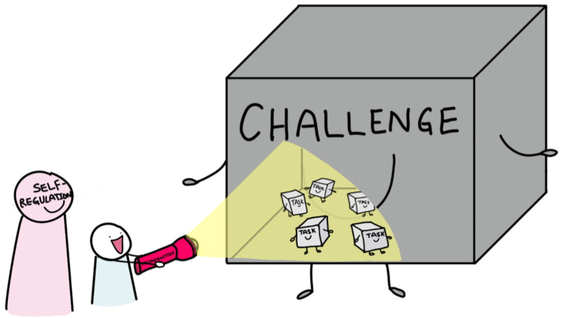
CS Challenge
mentor lead-in, breakout, report back

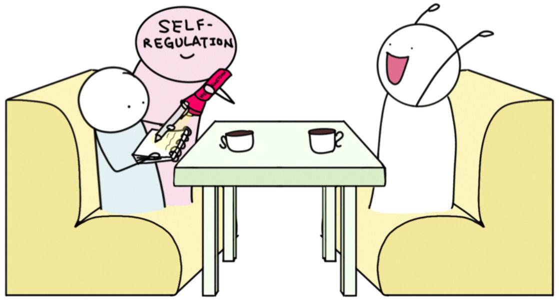
Learning topic 2 (ex: growth mindset)
mentor story, video, round robin

Open conversation

Weekly group session

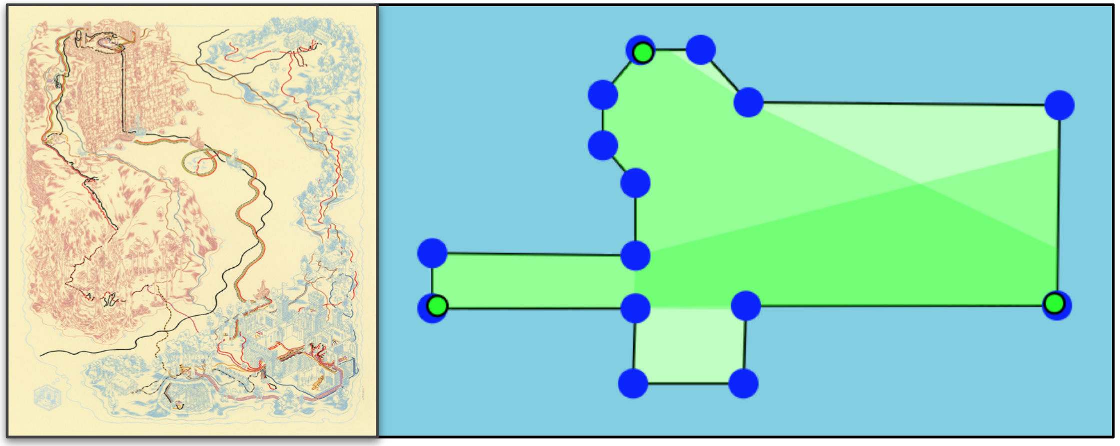
Session 2	Can I hack it?
Session 3	How do I find my way?
Session 4	Can I be strategic?
Session 5	Can I pivot?





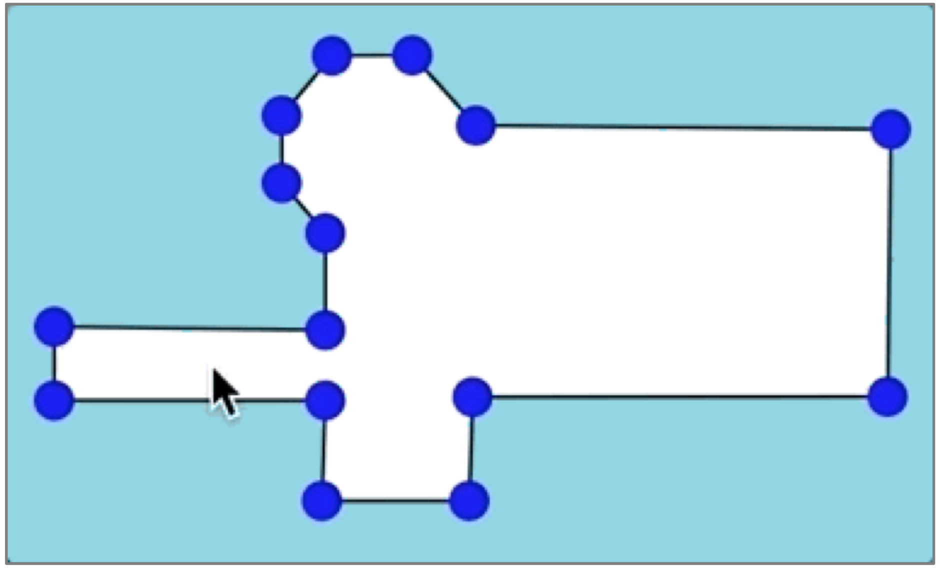
Strategizing Solutions

The Art Gallery Problem

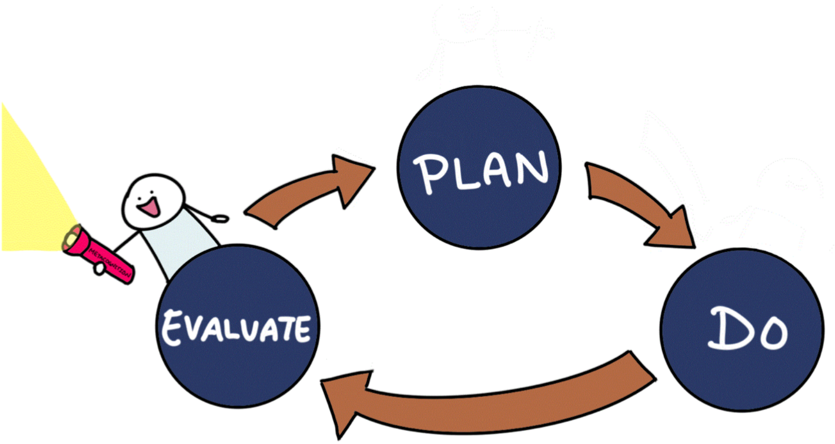


The Art Gallery Problem

- Challenge: guards cover the interior
- Input: simple polygon
- Output: (optimized) placement of guards



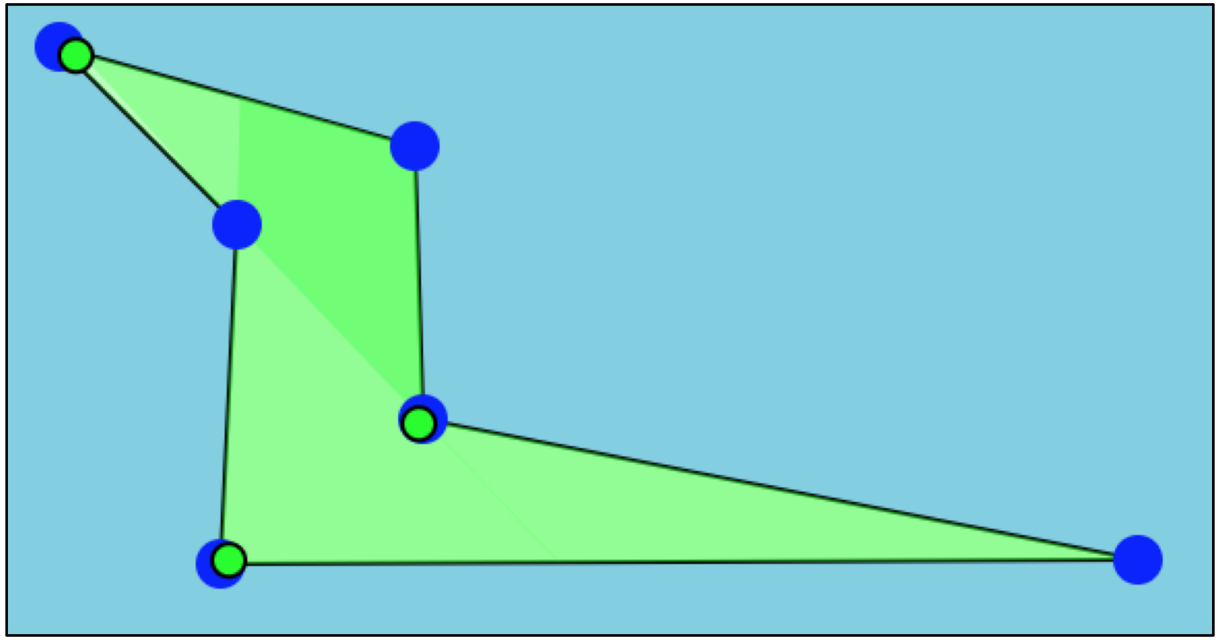
Are you up for the challenge?



Plan #1: every-other

Place n/2 guards

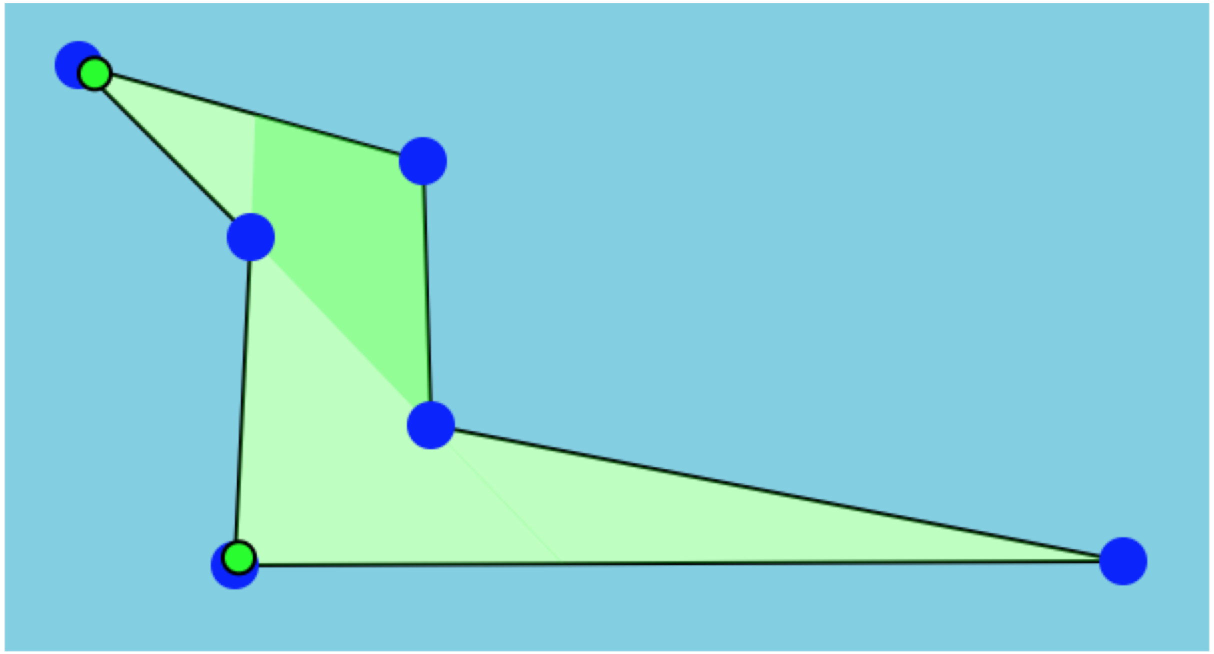
- Every other point



Plan #2: top-and-bottom

Place 2 guards

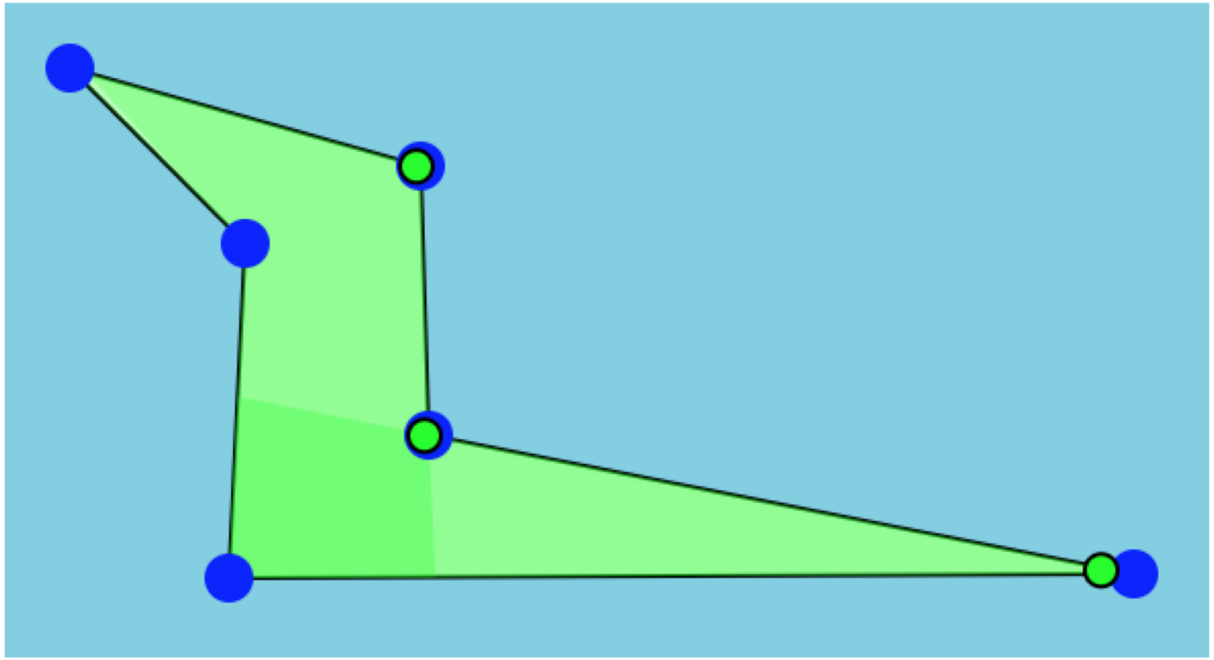
- One at the top
- One at the bottom



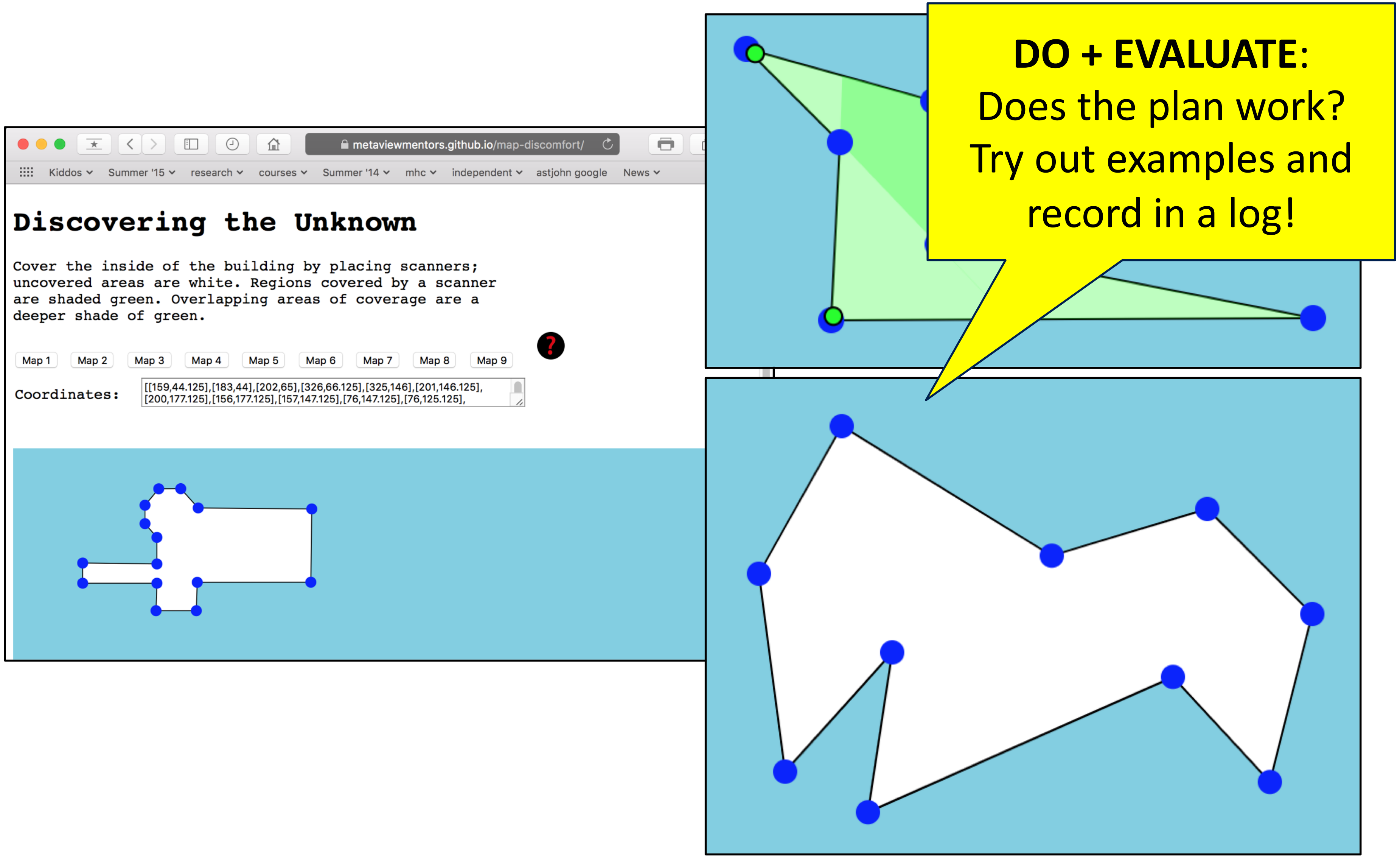
Plan #3: random-half

Place n/2 guards

- Pick random starting point
- Place n/2 guards from there



Pick a plan and try it!



Session structure

Lead in

Learning topic 1 (ex: self-efficacy)
mentor story, video, breakout, report back

CS Challenge
mentor lead-in, breakout, report back

Learning topic 2 (ex: growth mindset)
mentor story, video, round robin

Open conversation

Weekly group session

- Session 2 — Can I hack it?
- Session 3 — How do I find my way?
- Session 4 — Can I be strategic?
- Session 5 — Can I pivot?

Breakout

What if I don’t know the answer?

Breakout in small groups

Introductions (remember: pronouns)

Pick a facilitator

Facilitator: read the scenario out loud

Discuss (hear all voices)

Time-permitting: repeat

Remember your role(s)

Mentors are coaches.

Mentors are responsible role models.

Mentors are committed to learning.

Lesson

How can mentors support student learning?

Mentors help students learn by modeling strategies and encourage students to practice.

Students learn by watching others.

Students learn by practicing on their own and with others.

Bandura, A. (1986). *Social foundations of thought and action: A social cognitive theory*. Prentice-Hall, Inc.

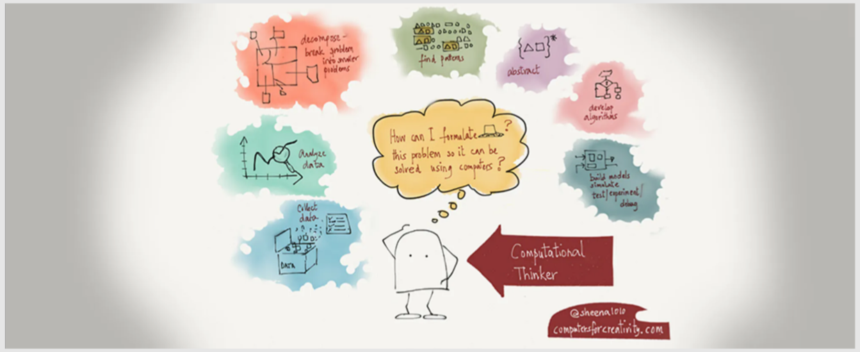
In this program, you will model computational thinking and problem-solving.

“A computational thinker is one who… **analyzes**…to understand the problem.

Then decomposes (**breaks it down**) into simpler problems.

Instead of solving only that problem, you look for **patterns**, remove details and abstract so you can solve all problems of that type.”

- Jeannette Wing



Jeannette Wing (2006, 2016) https://www.microsoft.com/en-us/research/blog/computational-thinking-10-years-later/

Sheena Vaidyanathan (2016) https://www.edsurge.com/news/2016-08-06-what-s-the-difference-between-coding-and-computational-thinking

Mentors can encourage students to build on small successes, which grows their self-efficacy.

Mentors can notice where students are making progress, and point out that mistakes = learning.

Kinnunen, P., & Simon, B. (2012). My program is ok – Am I? Computing freshmen's experiences of doing programming assignments. *Computer Science Education, 22*(1), 1–28.

Mentoring can come from the mentor and from the mentoring cohort (peer group).

Schunk, D. H. (1991). Self-efficacy and academic motivation. *Educational Psychologist, 26*(3-4), 207–231.

Students benefit from feedback that is timely and specific. Mentors can fill in the gaps.

Beyer, S. (2014). Why are women underrepresented in computer science? Gender differences in stereotypes, self-efficacy, values, and interests and predictors of future CS course-taking and grades. *Computer Science Education, 24*, 153–192.

Shute, V. J. (2008). Focus on formative feedback. *Review of Educational Research, 78*(1), 153–189.

Zimmerman, B. J. (1989). A social cognitive view of self-regulated academic learning. *Journal of Educational Psychology, 81*(3), 329–339.

Students miss out on valuable feedback during their intro course(s).

Courses may be large or lecture-based.

An early + positive CS experience predicts future enrollment.

Mentors can help provide valuable support and feedback + direct them to other sources.

Students benefit from feedback that is timely and specific. Mentors can fill in the gaps by encouraging strategies (not by giving answers).

- Problem-solving **strategies** are not always explicitly taught
- Students may not plan or check for their understanding.
- In Session 3: Can I be strategic? students try out a **plan** and are encouraged to **log** what they tried

Xie, Nelson & Ko (2018]

Loksa & Ko (2016)

Not all feedback is provided in a constructive way. Mentors can help unpack that feedback.

Students hear:

You aren’t cut out for this.

Mentors say:

“I am giving you this feedback because I take you seriously.”

Students hear:

This is what you did wrong.

Mentors say:

“Are you willing try some new strategies? I noticed this was present and this was absent, and you can make this better.”

Cohen, G. L., Steele, C. M., & Ross, L. D. (1999). The mentor's dilemma: Providing critical feedback across the racial divide. *Personality and Social Psychology Bulletin, 25*(10), 1302–1318.

Know which conversation you are having. Acting as a **mirror** can help.

Rather than jump into problem-solving...

"Here is what you should do differently."

Create space to find out.

"Tell me more about your process."

"Was it the interaction or the material?"

Clarifying your mentoring role

- Not a tutor, teacher
- You can offer ideas and suggestions of **how** to think about the problem

  “I’d be happy to think about that with you.”

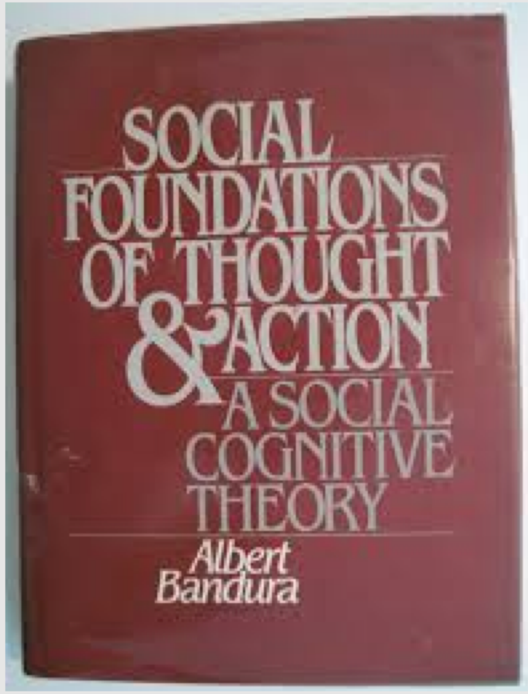
  “Let’s explore who on the campus can help.”
- You may need to report/elevate some things you hear
  - You cannot keep all things confidential
    - Students will be advised of this too.

  “Thanks for sharing that with me. I’m so sorry that is happening.”

  “I’d like to bring this up with my mentoring team and get their perspective.”

Bibliography

Bandura, A. (1986). *Social foundations of thought and action: A social cognitive theory*. Prentice-Hall, Inc.



Schunk, D. H. (1991). Self-efficacy and academic motivation. *Educational Psychologist, 26*(3-4), 207–231.
https://doi.org/10.1207/s15326985ep2603&4_2

Abstract
Discusses academic motivation in terms of self-efficacy, an individual's judgments of his/her own capabilities to perform given actions. An overview of self-efficacy theory is presented, and self-efficacy is contrasted with related constructs (perceived control, outcome expectations, perceived value of outcomes, attributions, and self-concept). Studies of the effects of person variables (goal setting and information processing) and situation variables (models, attributional feedback, and rewards) on self-efficacy and motivation are reviewed. Substantive issues that require more research include measures of self-efficacy, maintenance and generalization, classroom goals, and teaching processes.

Kinnunen, P., & Simon, B. (2012). My program is ok – Am I? Computing freshmen's experiences of doing programming assignments. *Computer Science Education, 22*(1), 1–28.

Abstract

This article provides insight into how computing majors experience the process of doing programming assignments in their first programming course. This grounded theory study sheds light on the various processes and contexts through which students constantly assess their self-efficacy as a programmer. The data consists of a series of four interviews conducted with a purposeful sample of nine computer science majors in a research intensive state university in the United States. Use of the constant comparative method elicited two forms of results. First, we identified six stages of doing a programming assignment. Analysis captures the dimensional variation in students' experiences with programming assignments on a detailed level. We identified a core category resulting from students' reflected emotions in conjunction with self-efficacy assessment. We provide a descriptive model of how computer science majors build their self-efficacy perceptions, reported via four narratives. Our key findings are that some students reflect negative views of their efficacy, even after having a positive programming experience and that in other situations, students having negative programming experiences still have a positive outlook on their efficacy. We consider these findings in light of possible languages and support structures for introductory programming courses.

Beyer, S. (2014). Why are women underrepresented in computer science? Gender differences in stereotypes, self-efficacy, values, and interests and predictors of future CS course-taking and grades. *Computer Science Education, 24*, 153–192.

Abstract

This study addresses why women are underrepresented in Computer Science (CS). Data from 1319 American first-year college students (872 female and 447 male) indicate that gender differences in computer self-efficacy, stereotypes, interests, values, interpersonal orientation, and personality exist. If students had had a positive experience in their first CS course, they had a stronger intention to take another CS course. A subset of 128 students (68 females and 60 males) took a CS course up to one year later. Students who were interested in CS, had high computer self-efficacy, were low in family orientation, low in conscientiousness, and low in openness to experiences were more likely to take CS courses. Furthermore, individuals who were highly conscientious and low in relational-interdependent self-construal earned the highest CS grades. Efforts to improve women's representation in CS should bear these results in mind.

Shute, V. J. (2008). Focus on formative feedback. *Review of Educational Research, 78*(1), 153–189. https://doi.org/10.3102/0034654307313795

Abstract
This article reviews the corpus of research on feedback, with a focus on formative feedback—defined as information communicated to the learner that is intended to modify his or her thinking or behavior to improve learning. According to researchers, formative feedback should be nonevaluative, supportive, timely, and specific. Formative feedback is usually presented as information to a learner in response to some action on the learner's part. It comes in a variety of types (e.g., verification of response accuracy, explanation of the correct answer, hints, worked examples) and can be administered at various times during the learning process (e.g., immediately following an answer, after some time has elapsed). Finally, several variables have been shown to interact with formative feedback's success at promoting learning (e.g., individual characteristics of the learner and aspects of the task). All of these issues are discussed. This review concludes with guidelines for generating formative feedback. (PsycINFO Database Record (c) 2016 APA, all rights reserved)

https://psycnet.apa.org/record/2011-14717-005

Zimmerman, B. J. (1989). A social cognitive view of self-regulated academic learning. *Journal of Educational Psychology, 81*(3), 329–339. https://doi.org/10.1037/0022-0663.81.3.329

Abstract
Researchers interested in academic self-regulated learning have begun to study processes that students use to initiate and direct their efforts to acquire knowledge and skill. The social cognitive conception of self-regulated learning presented here involves a triadic analysis of component processes and an assumption of reciprocal causality among personal, behavioral, and environmental triadic influences. This theoretical account also posits a central role for the construct of academic self-efficacy beliefs and three self-regulatory processes: self-observation, self-judgment, and self-reactions. Research support for this social cognitive formulation is discussed, as is its usefulness for improving student learning and academic achievement.

https://psycnet.apa.org/record/1990-06085-001

Cohen, G. L., Steele, C. M., & Ross, L. D. (1999). The mentor's dilemma: Providing critical feedback across the racial divide. *Personality and Social Psychology Bulletin, 25*(10), 1302–1318. https://doi.org/10.1177/0146167299258011

Abstract
Two studies examined the response of 125 Black and 121 White students to critical feedback presented either alone or buffered with additional information to ameliorate its negative effects. Black students who received unbuffered critical feedback responded less favorably than White students both in ratings of the evaluator's bias and in measures of task motivation. By contrast, when the feedback was accompanied both by an invocation of high standards and by an assurance of the student's capacity to reach those standards, Black students responded as positively as White students and both groups reported enhanced identification with relevant skills and careers. This "wise," two-faceted intervention proved more effective than buffering criticism either with performance praise (Study 1) or with an invocation of high standards alone (Study 2). The role of stigma in mediating responses to critical feedback, and the implications of our results for mentoring and other teacher–student interactions, are explored.

https://psycnet.apa.org/record/1999-11737-011

CS Problem Solving

An Explicit Strategy to Scaffold Novice Program Tracing

Benjamin Xie, Greg L. Nelson, & Amy J. Ko (2018)

https://faculty.washington.edu/ajko/papers/Xie2018TracingStrategies.pdf

The Role of Self-Regulation in Programming Problem Solving Process and Success

Dastyni Loksa & Amy J. Ko   (2016)

https://dl.acm.org/doi/pdf/10.1145/2960310.2960334

Computational Thinking

Jeannette Wing (2006, 2016) https://www.microsoft.com/en-us/research/blog/computational-thinking-10-years-later/

Sheena Vaidyanathan (2016) https://www.edsurge.com/news/2016-08-06-what-s-the-difference-between-coding-and-computational-thinking