



TEALS Summer Training #1

"TEALS 101"

<Location>

<Date>

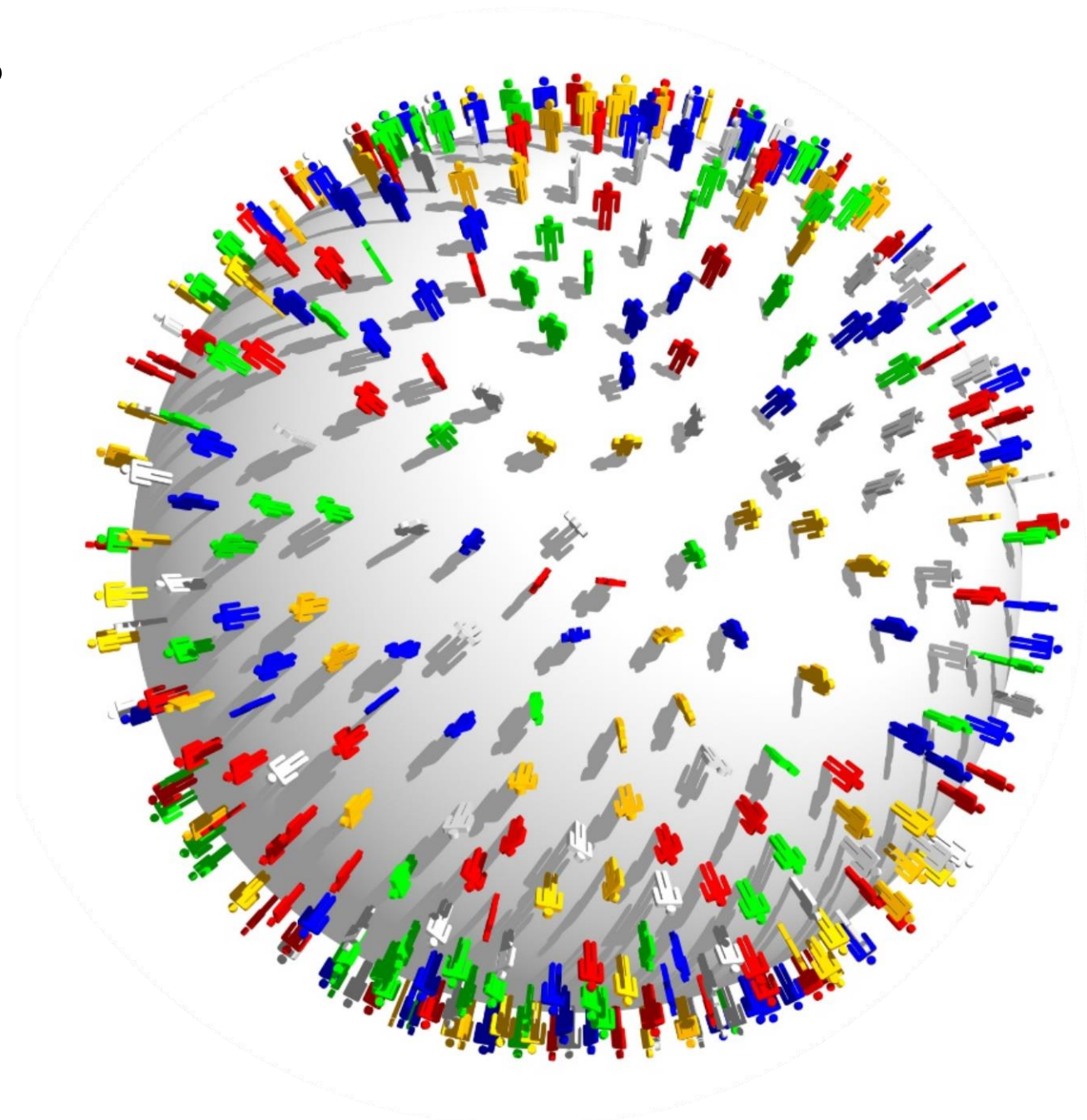
As you arrive, please...

1. Put on a nametag
2. Put your name on an index card and drop into the first box
3. Put your name on a raffle ticket and drop into the second box
4. Pick up an Activity Packet and work on page 3 (Pre-Training Reflection)
5. Please be seated as a teaching team

Know Your Demographics

Go to the Participant Demographic Chart and put a sticker next to your appropriate role or characteristic you identify with

- Coding Education (*Volunteers*)
- Professional Title (*Volunteers*)
- Subject Area (*Teachers*)
- Grade Level (*Teachers*)
- Schooling when growing up
- Community growing up



YOU HAVE 2 MINUTES.

Team Reflections

Discuss...

- How are the cultural and academic backgrounds of your team members **similar** to your students' experience?
- How are the cultural and academic backgrounds of your team members **different** from your students' experience?

Consider...

- The answers each team member added to the charts
- Your school's student demographic data



YOU HAVE 5 MINUTES.





Estimated Impact Together This Year

600

Schools

30

States/BC

21000

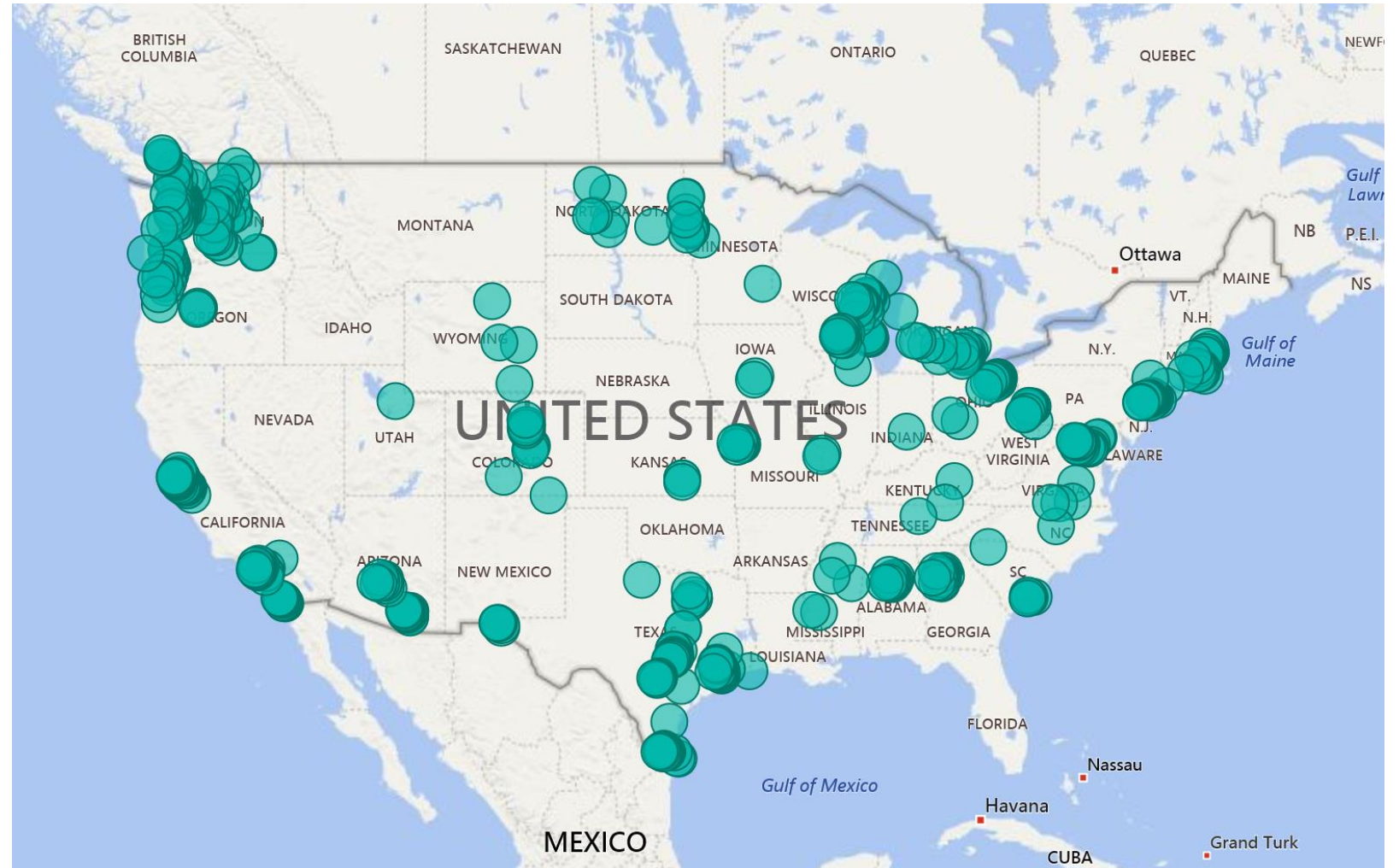
Students

1700

Volunteers

800

Companies

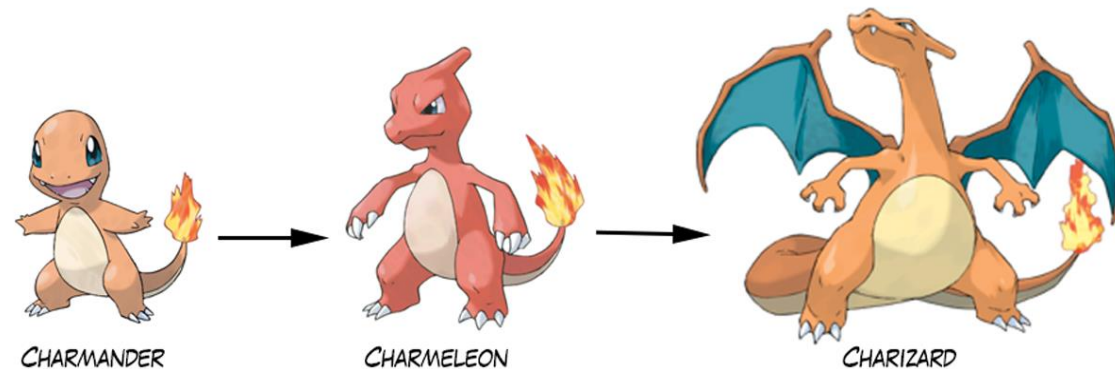


Training Objectives

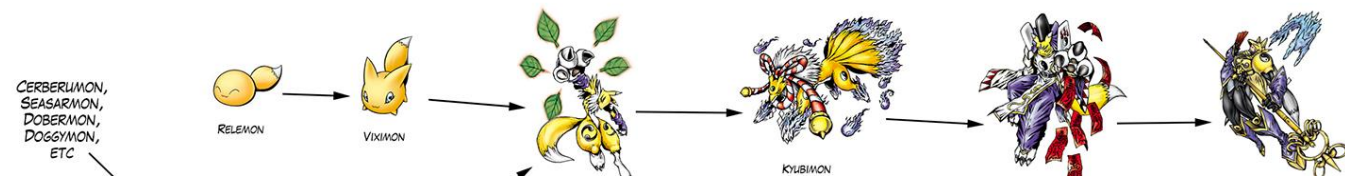
SWBAT (Students [a.k.a. you] Will Be Able To)...

- List the roles and responsibilities of each member of your teaching team
- Describe your team's plans for communication and working together
- Explain how your teaching team will use each member's strengths and weaknesses to achieve success in your classroom
- Define "pedagogical content knowledge" and list some CS-related examples
- Create a classroom culture where **all** students can be successful

POKEMON EVOLUTION



DIGIMON EVOLUTION



Summer of TEALS



What	When	Format
Training Session #1	TODAY!	Full day, in-person
Team Check-in #1	<Dates>	30 mins – 1-hour, full team and RM
Homework Assignments #1	Due by <Date>	Online
Training Session #2	<Date>	Full day, in-person
Team Check-in #2	<Dates>	30 mins – 1-hour, full team and RM
Homework Assignments #2	Due by <Date>	Online
Training Session #3	<Date>	2 hours
School Visit	By <Date>	At team's discretion
Training Session #3/School Year Kickoff	<Date>	Evening session

Training Expectations



Be an **active learner**, participating in all lessons and activities, taking notes, resisting “digital temptation” and asking for help when you need it.



Complete the **online homework assignments** by the assigned due dates. Communicate with your regional manager if you need help or more time.



Collaborate with your team during and outside of training. Together, complete a classroom plan and attend check-in meetings with your regional manager to prepare for the school year.

Online Resources

TEALS Dashboard

Tealsk12.org > "Log In"

- TEALS to-do's
- Link to curriculum
- Program resources

Canvas

Summer Training Materials

- In-person videos and slides
- Makeup assignments
- Homework assignments
- Extra topics

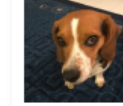
Forums

TEALS online community

- Get help with curriculum
- Lesson-specific discussions
- Student and teacher opportunities

TEALS Dashboard

About You



Test Volunteer
Volunteer

[Dashboard Settings](#)
[Update Login Details](#)
[Update 2018/19 Contact Info](#)

Status

[2018/19 Volunteer Partnership Promise](#) - Complete
[2018/19 TEALS Gear \(T-shirt & Bag\)](#) - Complete

[Teaching Team](#) - TEALS HS 1:INTRO:Section 2 (YR2018/19)

[Class Breakdown \(Demographics\)](#) - Last Updated on October 29, 2018 at 8:55 am
[Class Observations](#)

Calendar

Select Calendar

Event	Start	End	Location	Event Details
PNW - Makeup #5 (Wednesday)	April 2, 2019, 8:00 pm	April 2, 2019, 9:30 pm	Building 34/Quebec	Show Details

Show Past Events

Current timezone: Pacific
[Change Timezone](#)

Subscribed Calendars:

[_TEALSRuralAndDistance \(2018/19\) x](#) [TEALS-WA-PugetSound \(2018/19\) x](#)

Resources

2018/19

[Canvas](#) - Default password: CS4everyone (Please reset after first login.)

[Discussion Forum \(Discourse\)](#)

[AP CS A Curriculum](#)

[AP CS Principles Curriculum \(Code.org\)](#)

[AP CS Principles Curriculum \(BJC\)](#)

[AP CS Principles Curriculum \(BLTW\)](#)

[AP CS Principles Curriculum \(U Teach\)](#)

[AP CS Principles Curriculum \(MobileCSP\)](#)

[Intro CS Curriculum](#)

[Intro CS Semester 2 Curriculum](#)

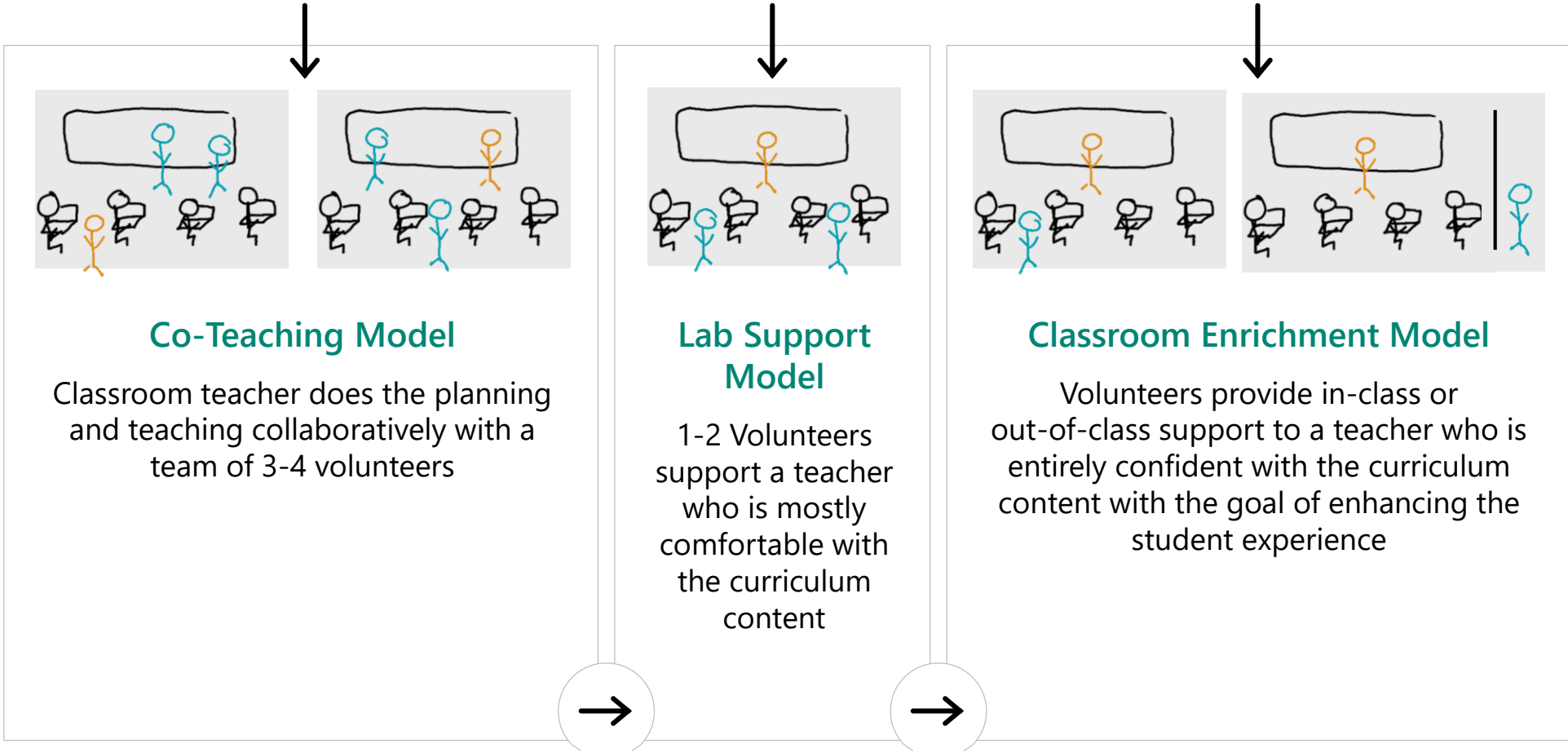
[Additional Curriculum Materials](#) (Username: volunteers@teals.onmicrosoft.com Password: Jasa7117 if problems accessing, use an InPrivate/Incognito browser window.)

[TEALS Pocket Guide](#)

Your Teaching Team



TEALS Support Progression



Classroom Teacher Responsibilities



Lead the teaching team

Give and take feedback

Manage classroom and school issues

Learn computer science!

Volunteer Roles and Responsibilities

	Co-Teaching Model	Lab Support Model	Classroom Enrichment Model
Number of volunteers	3-4 volunteers (1-2 in "teacher" role, 1-2 in "TA" role)	1-2 volunteers	1-2 volunteer
Presence in class	~2x per week per volunteer (2 volunteers per day, 2-4 days per week)	~2x per week per volunteer (1 volunteer per day, 2-4 days per week)	Range from 1-2/wk to occasional (sub coverage, special presentations, or career talks)
Classroom teacher support	Aid the teacher in learning CS content Advise the teacher in planning and leading occasional lessons	Support the teacher in deepening CS content knowledge Advise in planning and leading lessons	Mentor the teacher to check their CS knowledge and learn about real world CS applications
Instruction/ Planning support	Plan and lead lessons ("teacher" role) Support/co-teach lessons ("TA" role)	Support/co-teach lessons	Present occasional special topics
Lab support	Help unblock students as they work on their assignments		In some cases, help unblock students
Grading support	Help grade assignments		Sometimes

Reflections



Introduce yourself to the other members of your teaching team



Share your responses to the **first 3 questions** on the reflection worksheet



YOU HAVE 5 MINUTES.



Reflecting on Reflections

Compare your team's answers to the reflection questions.

Look for:

1

Strengths that are common across most/all team members

2

Worries or weaknesses that are common across most/all team members

3

Strengths from some members that match weaknesses from others

YOU HAVE 5 MINUTES.



Reflecting on Reflections

Group 1 are your
superpowers

Elements of running a TEALS class
your team should be great at

Group 2 are your
danger zones

Areas to which you'll need to
give special focus and where you
may need outside assistance

Group 3 are your
learning opportunities

The team as a whole has everything
it needs to succeed, but will need
to learn to coordinate and share

Working as a Teaching Team



The Classroom Plan

Guided activities to prepare your team for the year

Series of decisions and discussions

Single, shared document per team
In OneDrive/DropBox/Google Drive/etc.

Reference throughout the school year

Blank Classroom Plans:

<http://www.tealsk12.org/ClassroomPlan>

TEALS Program

Computer science in every high school

TEALS Classroom Plan 2019-2020:
Co-Teaching Model

School Name: _____

[Why the Classroom Plan is Important and How Your Team Should Use It](#)

The Classroom Plan serves as a guide to organize your team to teach a year of computer science through the TEALS program. The classroom teacher has done some pre-work on this document and that version should be the starting point for the team. There should be **one copy** of this document per teaching team (classroom teacher + TEALS volunteers), with access available to all the members of the teaching team.

The concepts within the Classroom Plan relate to what you will or have learned in TEALS Summer Training. You should reference the online training materials in Canvas (<http://tealsk12.instructure.com>) if you need a reminder about an idea or reach out to your regional manager.

The Classroom Plan asks you to discuss a question as a team and make a decision about how you will run your class. **The classroom teacher is the final arbiter of what happens in the classroom** – after all, they are legally responsible for the wellbeing of the students. In each section, we provide a recommended default approach (in *gray text*). You should adjust this approach based on local factors of your classroom (examples: age and background of the students, physical layout of the classroom, teacher's comfort level with course content, *etc*). Even if you decide to keep our recommendation, you should discuss why as a group.

Throughout the school year, if something isn't working in your class, revisit this plan and try modifying some of the decisions you made over the summer. This document is also really helpful if volunteers are added to your team later on.

[The Co-Teaching Model](#)

The TEALS Co-Teaching Model is designed to help teachers with minimal background in computer science learn the CS course content and become comfortable as the leader of the selected class. The [TEALS Implementation Guide](#) contains an appendix that outlines a learning model for classroom teachers in this role.

As you work through the classroom plan, always keep an eye towards the goal of supporting the classroom teacher's learning and comfort.

Fitting Together

- You will be presented with a series of scenarios
- With your teaching team, decide what role each of you could play in that situation
- Think about your superpowers and learning opportunities



Situation #1 – Planning

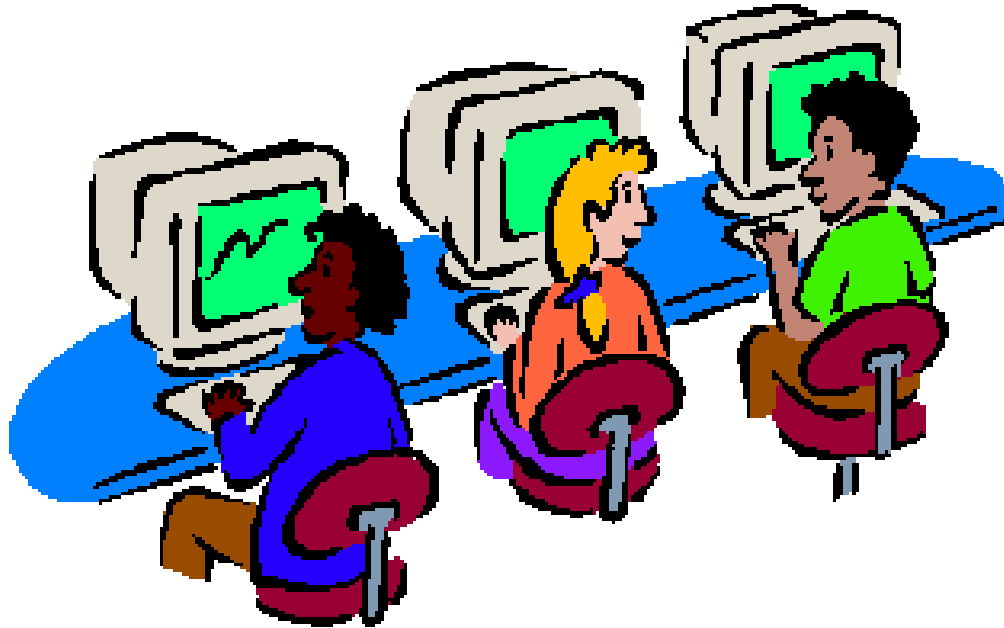
The team is planning next week's lessons covering conditionals and Boolean logic. You have the lesson plan and supporting materials from your curriculum.



What role will each member of your team take on during the planning process?

Situation #2 – Lab

It is lab time, and students are working. Some are making progress, some appear to be struggling, and a few are not on task. Several students were also absent yesterday on a field trip.



What role will each member of your team take during this lab?

Situation #3 – Instruction

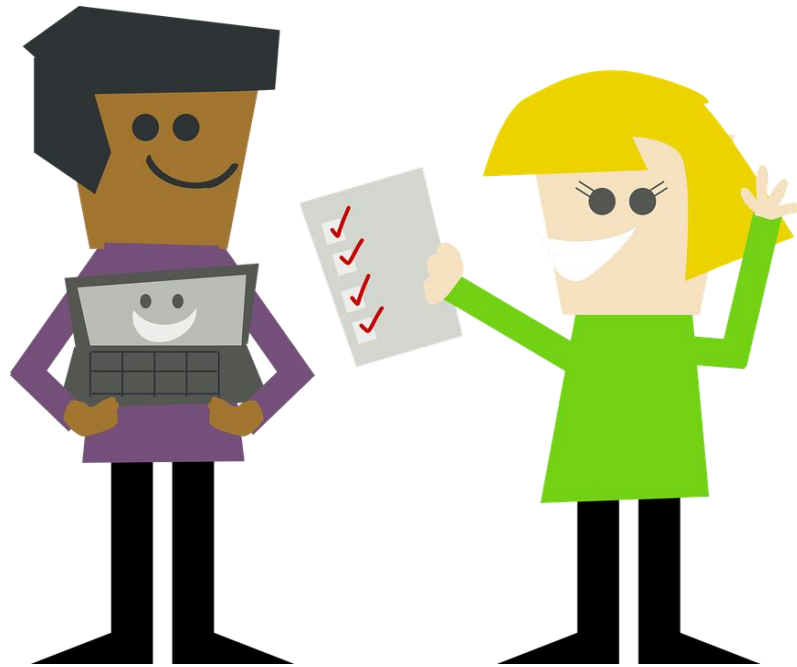
Loops are being introduced for the first time, and one member of the team is in front of the class leading instruction.



What role will each member of your team take on during instruction?

Situation #4 – Grading

Students have just submitted their first large project and it is time to start grading. You have the project spec and rubric from your curriculum.



What role will each member of your team take on during grading?

Memory Game

Get Ready

1

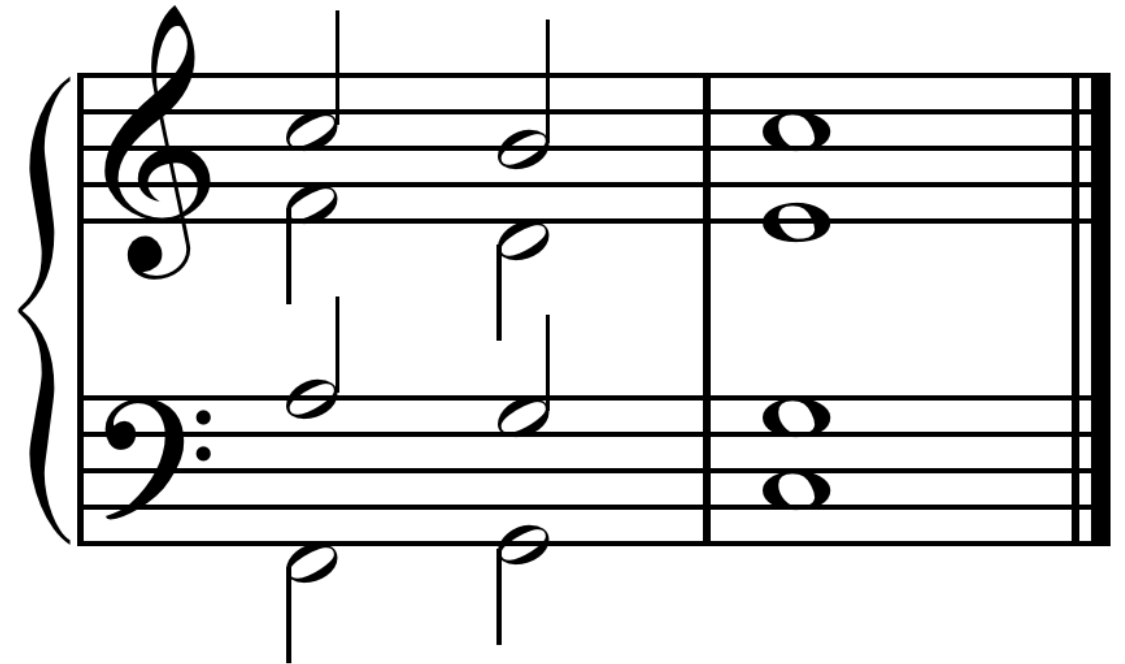
10 seconds to look at a picture

2

Then, 1 minute to reproduce the picture in your notebook

3

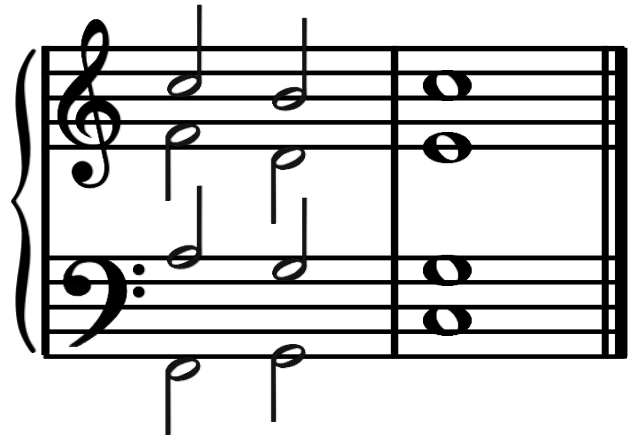
Afterwards, compare drawings with your neighbors



Are they similar or different? Why?

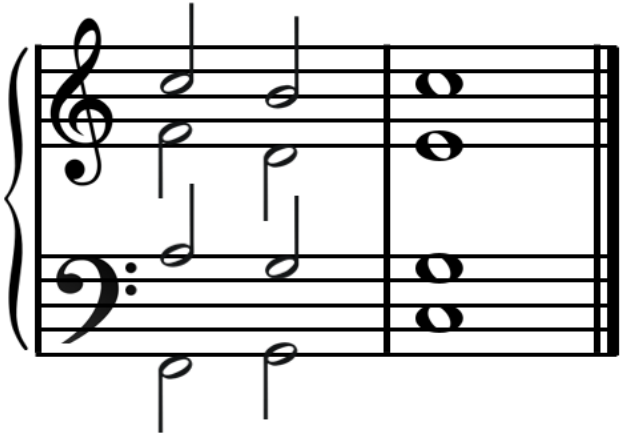
Chunking Expert

Expert

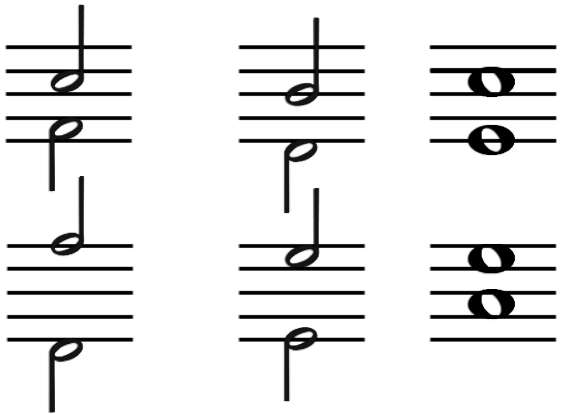
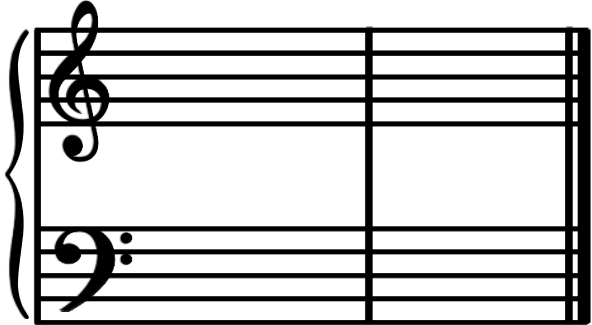


Chunking Intermediate

Expert

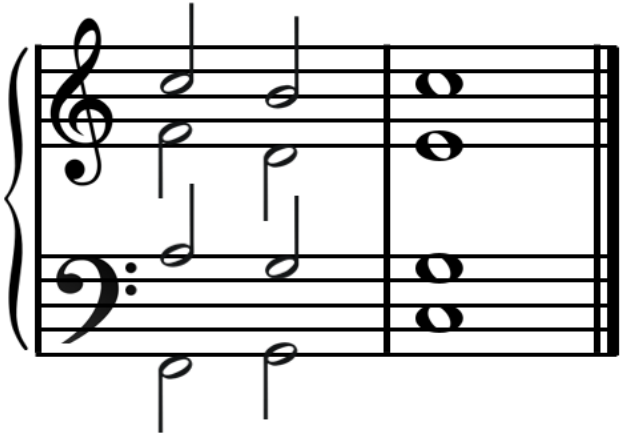


Intermediate

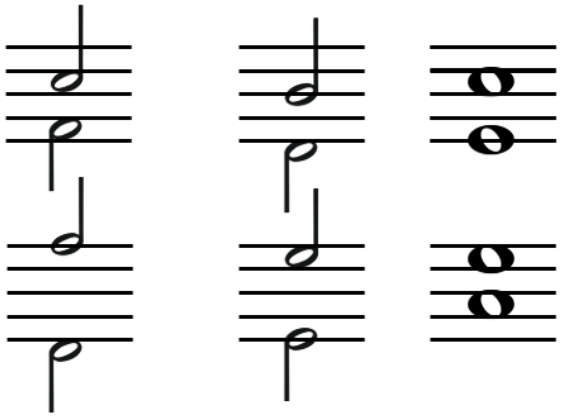
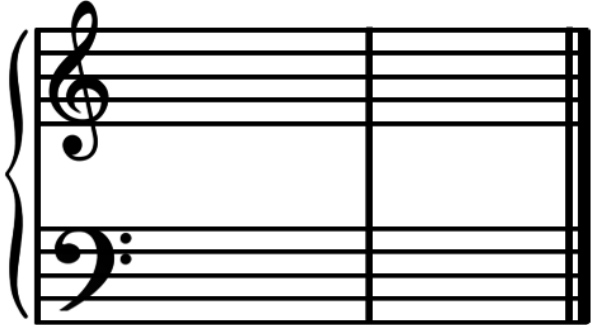


Chunking Novice

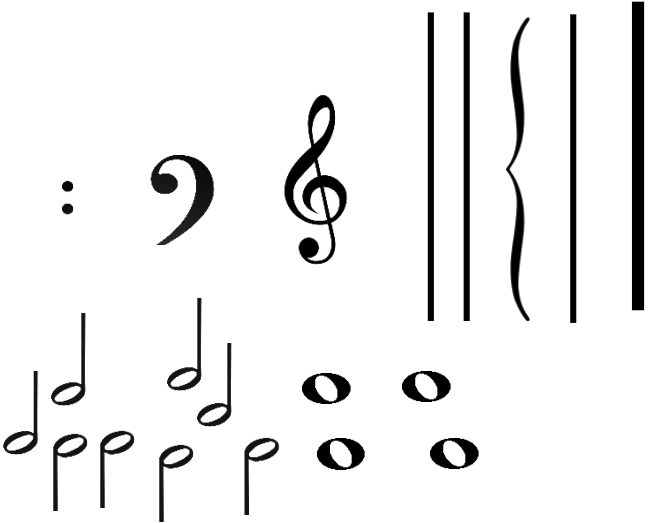
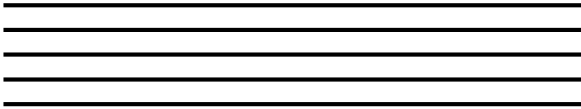
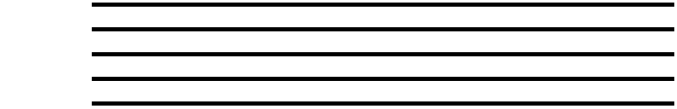
Expert



Intermediate

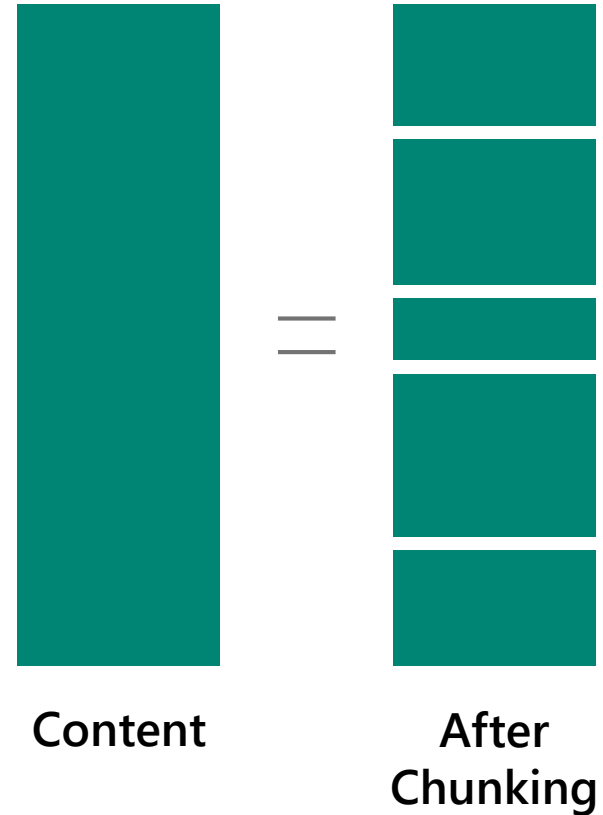


Novice



Chunking

- Brains use pattern-matching to chop information into chunks
- Experts' chunks tend to be bigger/more abstracted than beginners'



Expert Bias

“Perceiving something as easier or simpler than it is due to one’s own experience or knowledge of the subject”

- EXAMPLES**
- Chunking
 - Jargon
 - Time expectations
 - Giving few examples

Classroom Teachers

Help catch and mitigate expert bias



Team Communication

Plan on *at least* these two forms of communication:

Daily Handoff

Asynchronous

Brief

Searchable



Weekly Sync

Real-time

Bigger picture



Classroom Plan Check-in #1



Select 2 roles from the “Classroom Roles” section of the Classroom Plan and discuss how the different roles position the teacher as leader of the teaching team.



Remember to keep your Classroom Plan document somewhere everyone can access it!



YOU HAVE 8 MINUTES.

Sample Teaching Rotations

Monday	Tuesday	Wednesday	Thursday	Friday
Alyssa (classroom teacher)	Alyssa (classroom teacher)	Alyssa (classroom teacher)	Alyssa (classroom teacher)	Alyssa (classroom teacher)
Miya	Tho	Tho		Miya
Aidan	Aidan	Shaun	Shaun	

Monday	Tuesday	Wednesday	Thursday	Friday
Jerome (classroom teacher)	Jerome (classroom teacher)	Jerome (classroom teacher)	Jerome (classroom teacher)	Jerome (classroom teacher)
Alina	Alina		Suraj	Suraj

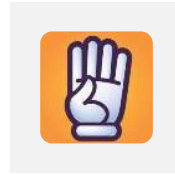
Monday	Tuesday	Wednesday	Thursday	Friday
David (classroom teacher)	David (classroom teacher)	David (classroom teacher)	David (classroom teacher)	David (classroom teacher)
	Cherie			Cherie

Keeping in Touch

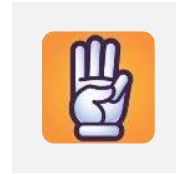
For each item on the following slide, think about how often it should be tracked and/or discussed?



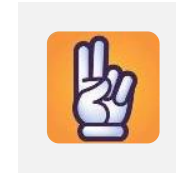
In class, as it happens



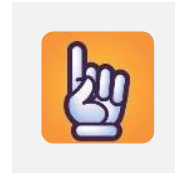
In the daily handoff communication(s)



In the weekly team sync, every week



In the weekly team sync, but not every week



Ad-hoc/as appropriate

Keeping in Touch

A "We got through the lab today, but didn't have time to go over the solutions"

B "It's taken us at least two days to finish the last few lesson plans"

C "No one scored lower than 80% on the last test; the students seem to understand this"

D "Alex hasn't turned in any homework in a while"

E "I'm still getting a lot of questions about variables"

F "We'll have to do the test on Thursday because there's an all-school assembly on Friday morning"

G "Your lesson on loops went pretty quickly, and you used some terms without defining"

H "You said that the body of the if statement is executed if the condition is false, but it's actually the other way around"

I "The CS class is being moved to 3rd period next semester"



As it happens



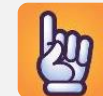
Daily handoff



Weekly sync
(always)



Weekly sync
(sometimes)



As appropriate

Classroom Plan Check-in #2



Work through the “Daily Handoff Plan” and “Weekly Team Sync” sections of your Classroom Plan.



Start filling out these sections of your Classroom Plan as a group.



Remember to keep your Classroom Plan document somewhere everyone can access it!



YOU HAVE 8 MINUTES.

Computer Science Pedagogy



Content Knowledge

“what to teach”

Syntax

Programming Languages

Data representations

Algorithms

Abstraction

Tools

Real-world applications

...

Pedagogical Knowledge

“How to teach”

Student mindsets

Classroom management

Theory of learning

Differentiated instruction

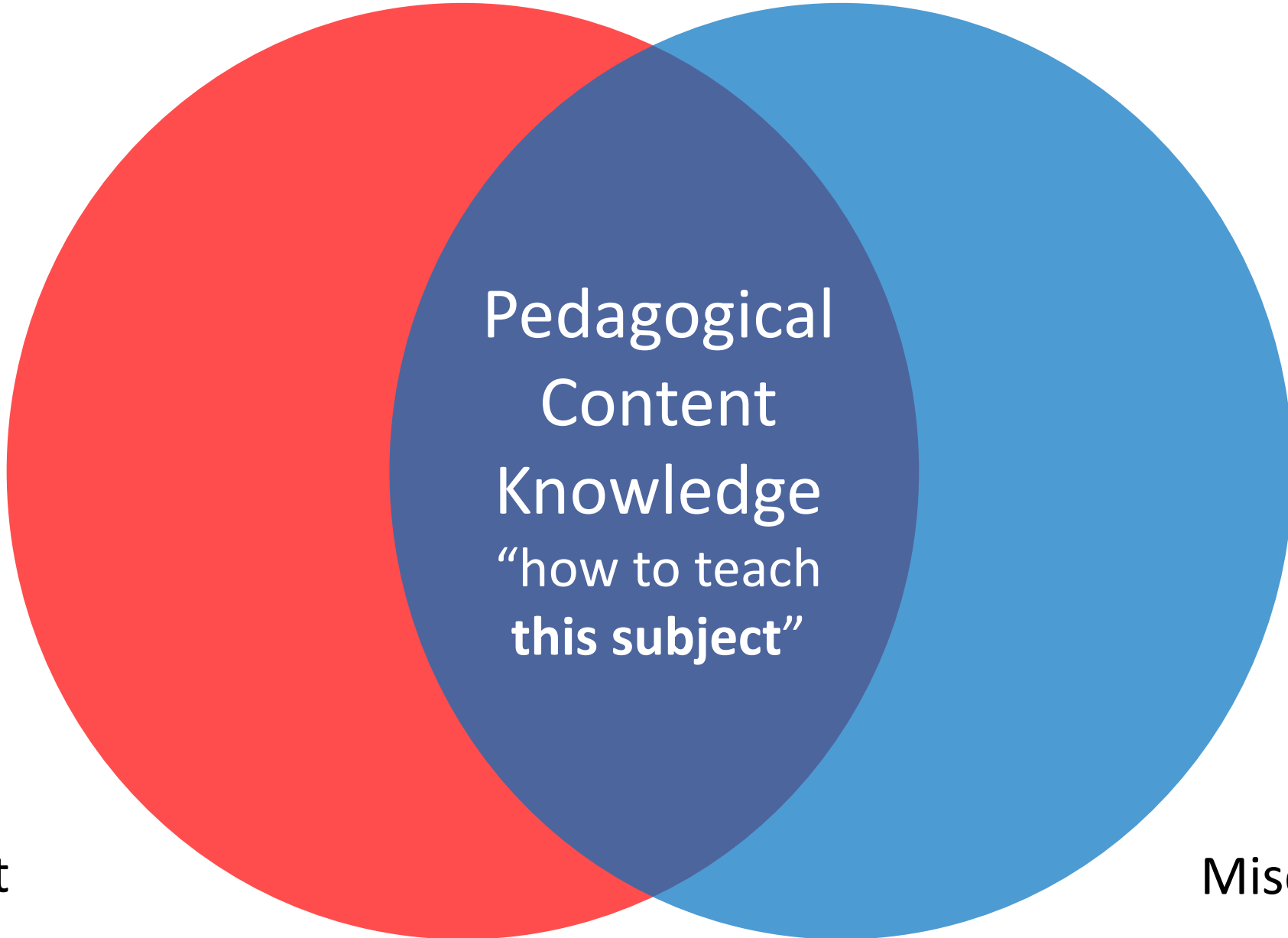
Lesson planning

Questioning techniques

...

Teaching
Methods

Curriculum
Scope &
Sequence



Forms of
Assessment

Student
Misconceptions

Our PCK Pillars

The Notional
Machine

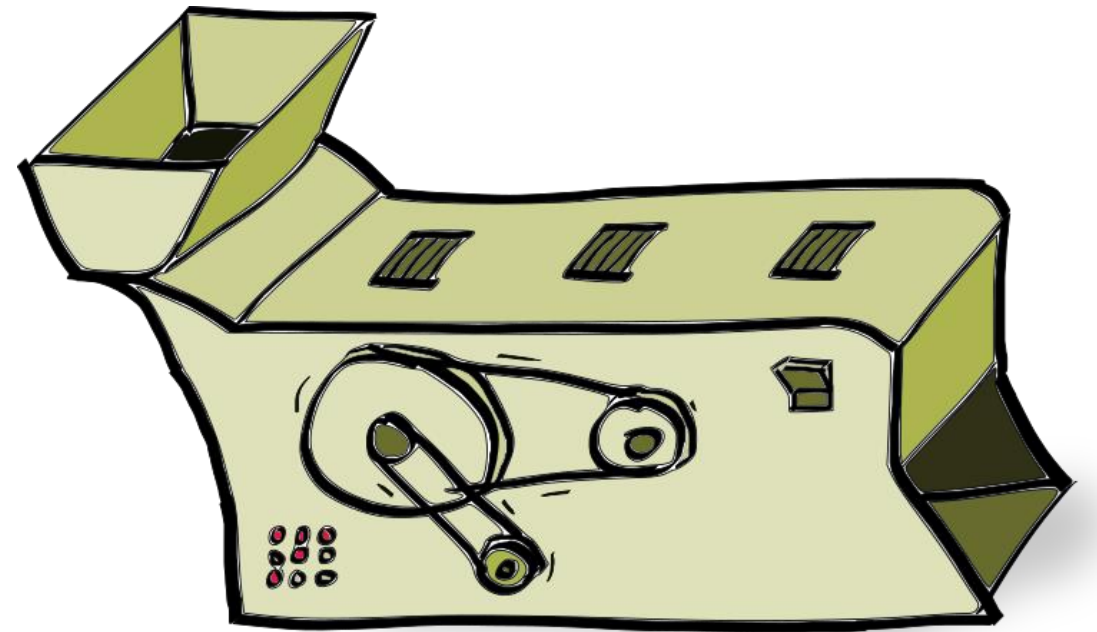
Problem
Solving

Hierarchy of
Skills

Inclusive
Teaching

Pillar 1: The “Notional Machine”

Students need to build a mental model of how the computer works and how it evaluates code.



Pillar 1: The “Notional Machine”

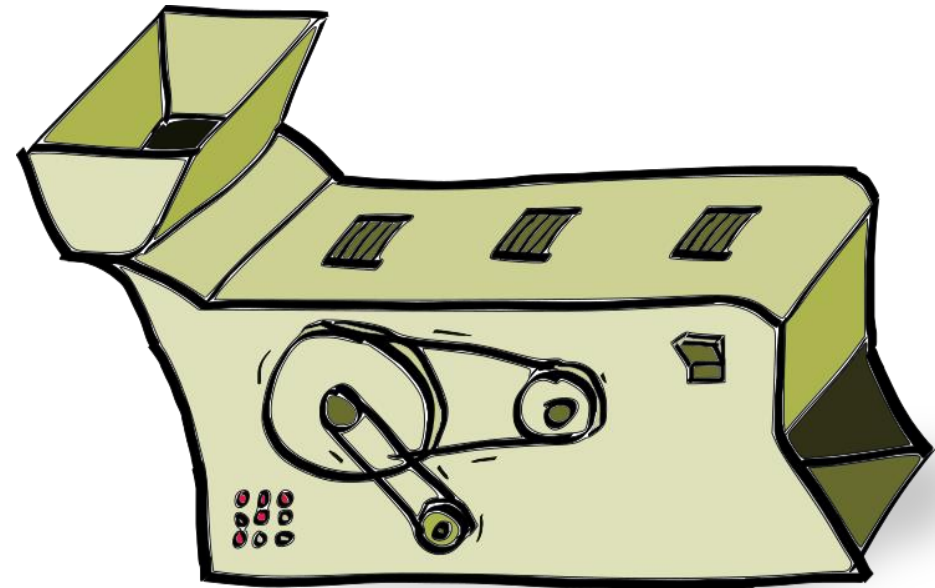
Tracing code

Memory diagrams

Worked examples

Data-structure diagrams

Analogies and examples

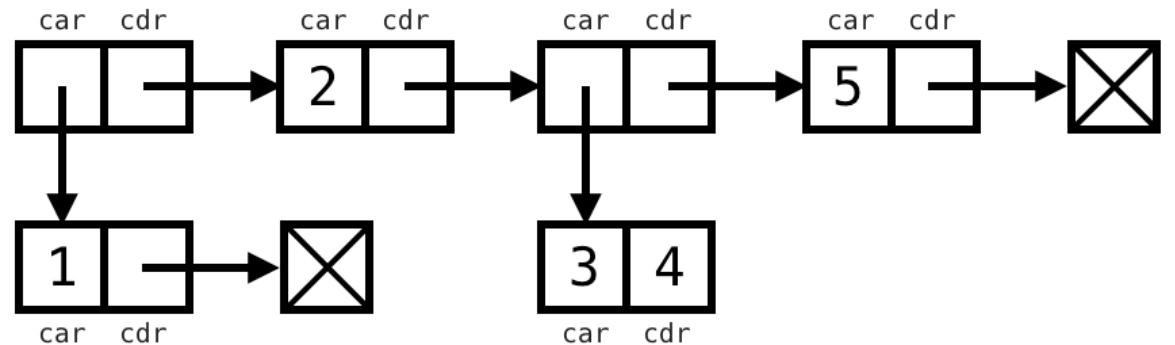


Memory diagrams

Processes for visualizing the state of the computer as it executes code

Typically used while tracing through a code example

Mimics a "debugger"



Ithaca-Style Memory Diagrams

```
1> s = "hello"  
2> x = 4  
3> y = 6  
4> x = 8  
5> total = x + y + x  
6> print(total)
```

Activity: Memory diagrams

Work through creating a memory diagram for the code to the right

Volunteers lead the activity

Teachers ask questions and look for expert bias

```
1> name = "Denise"  
2> age = 15  
3> if age < 16:  
4>     canDrive = false  
5> if age >= 16:  
6>     canDrive = true  
7> output = name + " is "  
8> if not canDrive:  
9>     output = output + "not "  
10> output = output + "eligible to drive."  
11> print(output)
```



YOU HAVE 6 MINUTES.

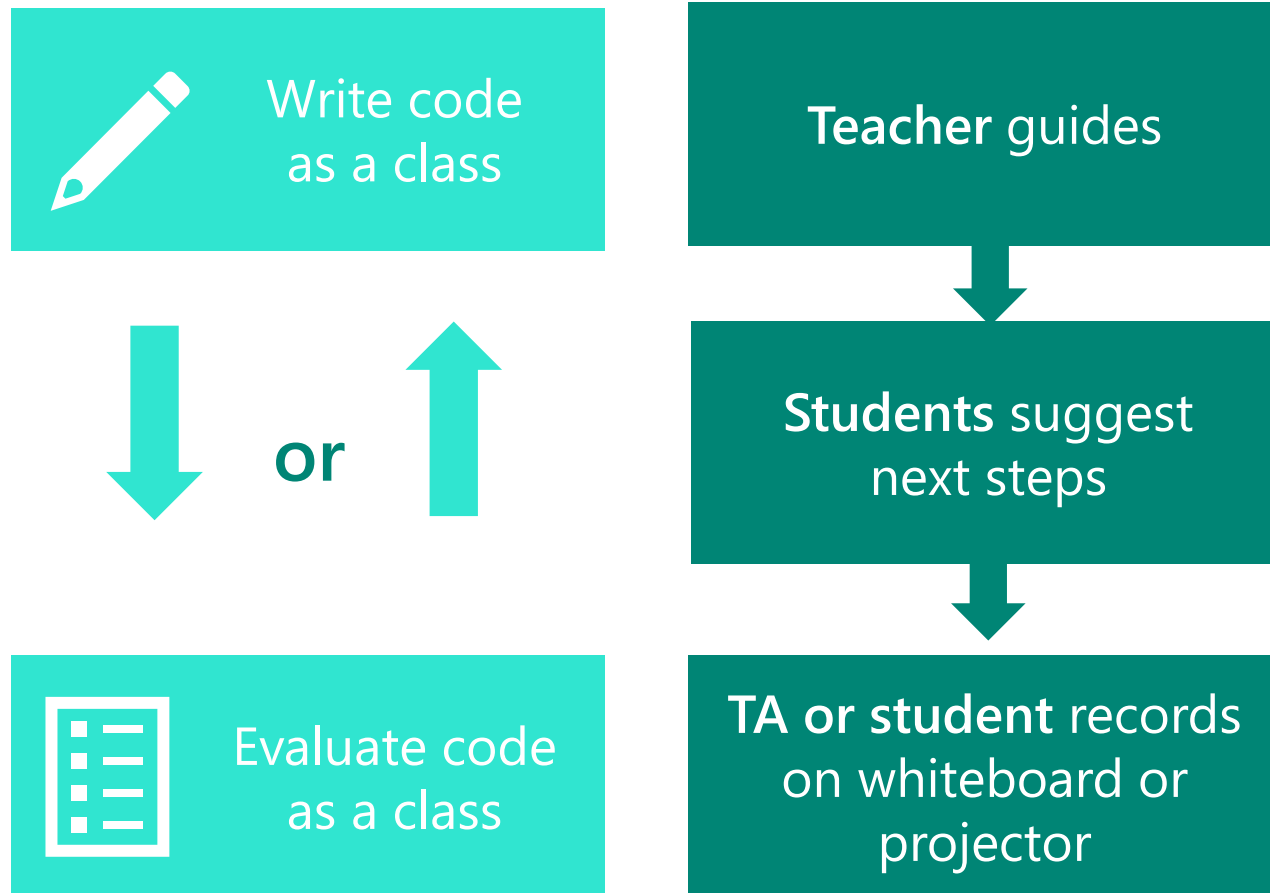
Activity: Memory diagrams

output	"Denise is not eligible to drive"
canDrive	"Denise is " false
age	15
name	"Denise"
Identifiers	Values

```
1> name = "Denise"
2> age = 15
3> if age < 16:
4>     canDrive = false
5> if age >= 16:
6>     canDrive = true
7> output = name + " is "
8> if not canDrive:
9>     output = output + "not "
10> output = output + "eligible to drive."
11> print(output)
```

output
Denise is not eligible
to drive.

Worked Examples

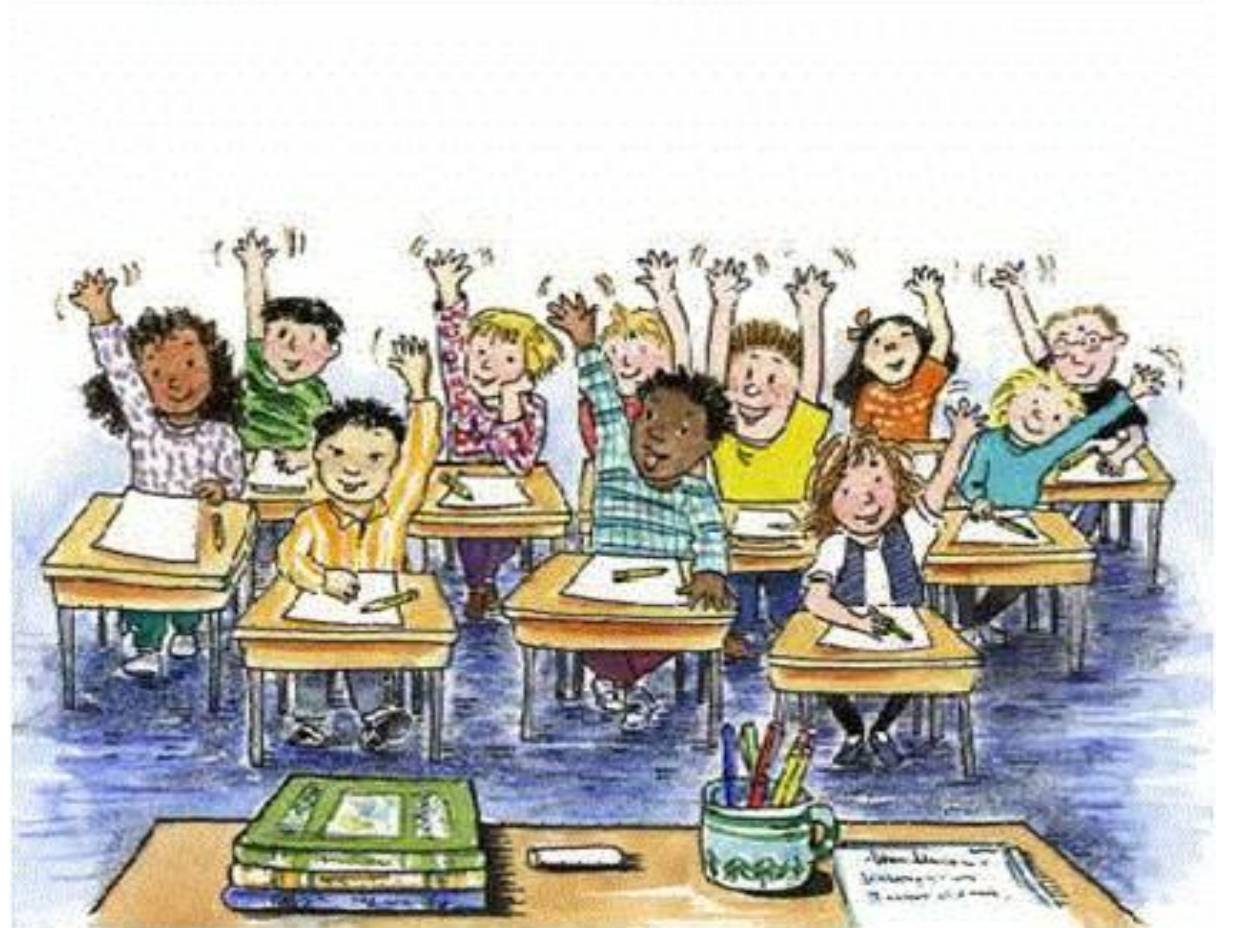


"We do" then "you do"

- Have a clear transition ("Now it's your turn." "OK, see if you can..." etc.)
- Students can reimplement, extend the example, or do a variation
- Be careful of chunking; provide hints (including pseudocode)

Worked Example Preparation

- Have a clear start and end point
- Decide what is in and out of scope
- Word questions carefully
- Focus on learning objectives
- Reference patterns and idioms
- Consider multiple possible approaches
- Know what comes next



Pillar 2: Solving Problems

Problem Solving is an integral part of what students need to learn and do in computer science.



Pillar 2: Solving problems

- Four steps to solve any CS Problem
- Subgoal Labeling
- Notebooks
- Debugging techniques
 - Print Statement
 - Isolation
 - Rubber Duck
 - Debugger tool
- Socratic Method



Subgoal Labeling

- Improves learning outcomes and retention
(Margelieux, Guzdial, Catrambone 2012)
- Models breaking a problem into parts
- Give a short, clear label to each part
- Incorporate the labels into and/all of:
 - Lab/project instructions
 - Starter code template (“code skeleton”)
 - Sample solutions



Subgoal Labeling in Lab Instructions

WITHOUT LABELS

1. Click on "My Blocks" to see the blocks for components you created.
2. Click on "clap"
3. Drag out a when clap.Touched block
4. Click on "clapSound" and
5. Drag out call clapSound.Play
6. Connect it after when clap.Touched

WITH LABELS

Handle Events from My Blocks

1. Click on "My Blocks" to see the blocks for components you created.
2. Click on "clap"
3. Drag out a when clap.Touched block

Set Output from My Blocks

4. Click on "clapSound" and
5. Drag out call clapSound.Play
6. Connect it after when clap.Touched

Subgoal Labeling in Code

WITHOUT LABELS

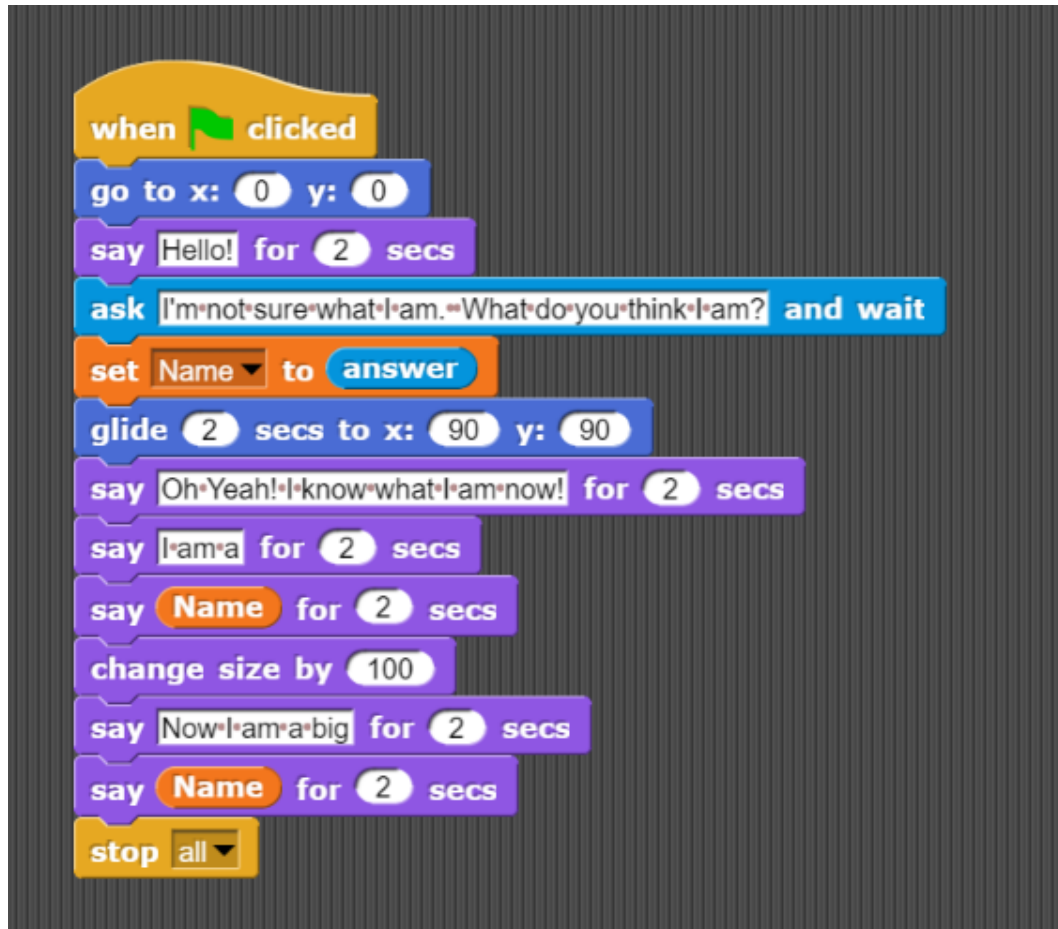
```
def countEmAll(someword):  
    count = 0  
    for letters in someword:  
        count = count + 1  
    return count
```

WITH LABELS

```
def countEmAll(someword):  
    # Start an accumulator  
    count = 0  
    # Go through the pieces  
    for letters in someword:  
        # Do something with each piece  
        count = count + 1  
    # Return the result  
    return count
```

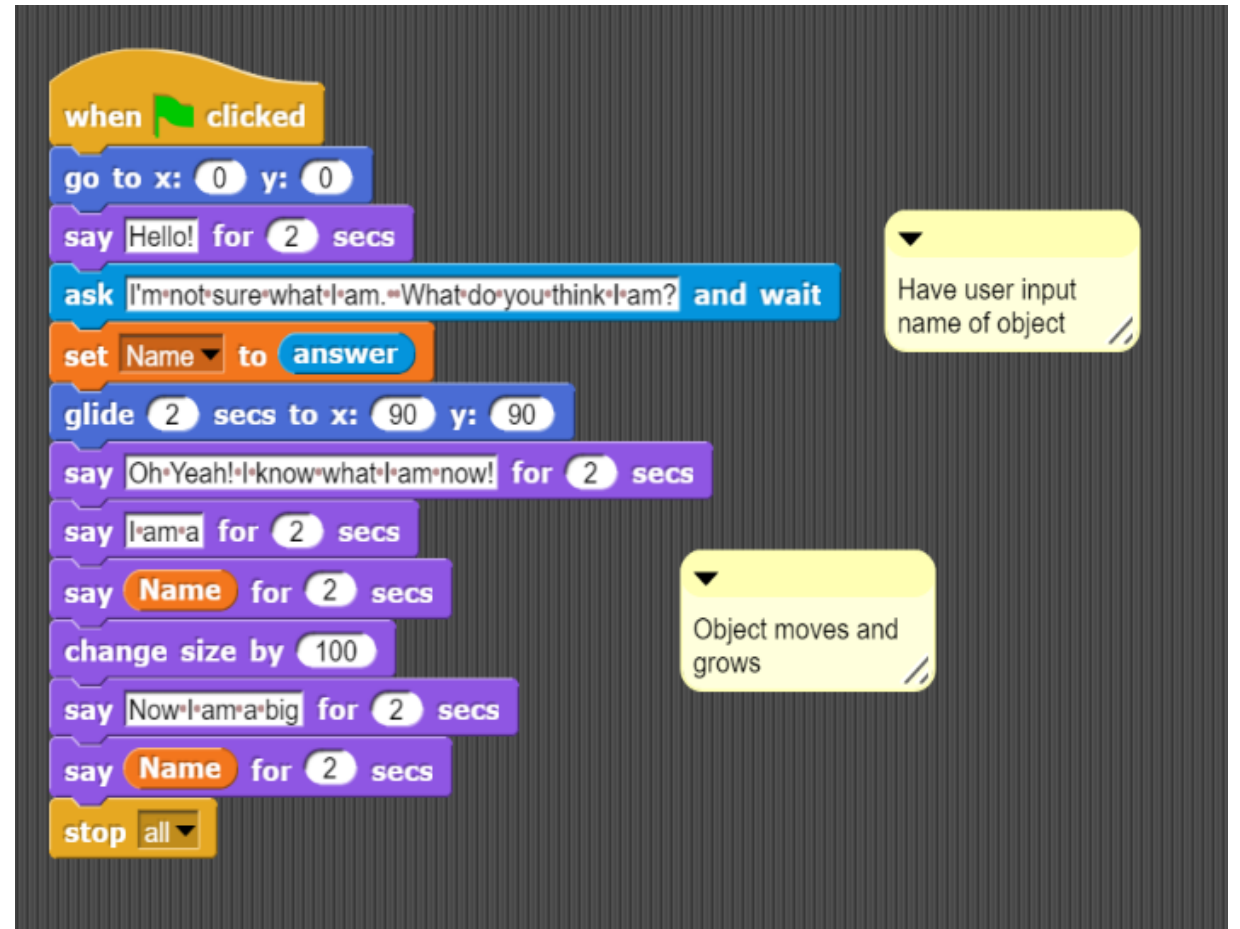
Subgoal Labeling in SNAP!

SNAP! With no comments



```
when clicked
  go to x: 0 y: 0
  say Hello! for 2 secs
  ask I'm not sure what I am. What do you think I am? and wait
  set Name to answer
  glide 2 secs to x: 90 y: 90
  say Oh Yeah! I know what I am now! for 2 secs
  say I am a for 2 secs
  say Name for 2 secs
  change size by 100
  say Now I am a big for 2 secs
  say Name for 2 secs
  stop all
```

SNAP! With comments



```
when clicked
  go to x: 0 y: 0
  say Hello! for 2 secs
  ask I'm not sure what I am. What do you think I am? and wait
  set Name to answer
  glide 2 secs to x: 90 y: 90
  say Oh Yeah! I know what I am now! for 2 secs
  say I am a for 2 secs
  say Name for 2 secs
  change size by 100
  say Now I am a big for 2 secs
  say Name for 2 secs
  stop all
```

Have user input name of object

Object moves and grows

Debugging with Students

Debugging is one of the most common tasks performed by computer scientists. Think about...



What tools and techniques do you use when solving problems?

How is helping someone else solve a problem different than solving one for yourself?



How is helping a *student* solve a problem different than helping a colleague solve a problem?

Why might it be important to *not* tell a student the solution?

Debugging with Students

When debugging with students

**Narrate
the process**

Students can't
read your mind

**Explain your
conclusions**

What's obvious to you
may not be obvious
to students

**Unpack
the problem**

Students tend to work
in smaller chunks

LOOK OUT FOR EXPERT BIAS!!

Rubber Duck Debugging

Describing a problem out loud often helps lead to a solution

- Even without a response

Encourage students to talk out their problem

Writing as they talk can help even more!



Activity: Rubber Duck Debugging



Talk through your process of solving the problem on the next slide

(As though you were talking to a rubber duck)



As you talk, write down key insights in your notebook



Debrief what you learned with your teaching team



You have 8 minutes

Activity: Rubber Duck Debugging

Problem: Write a program that asks the user for a number n, and prints out a grid of '*' symbols with 5 '*'s in each row, and n rows.



Student's Solution

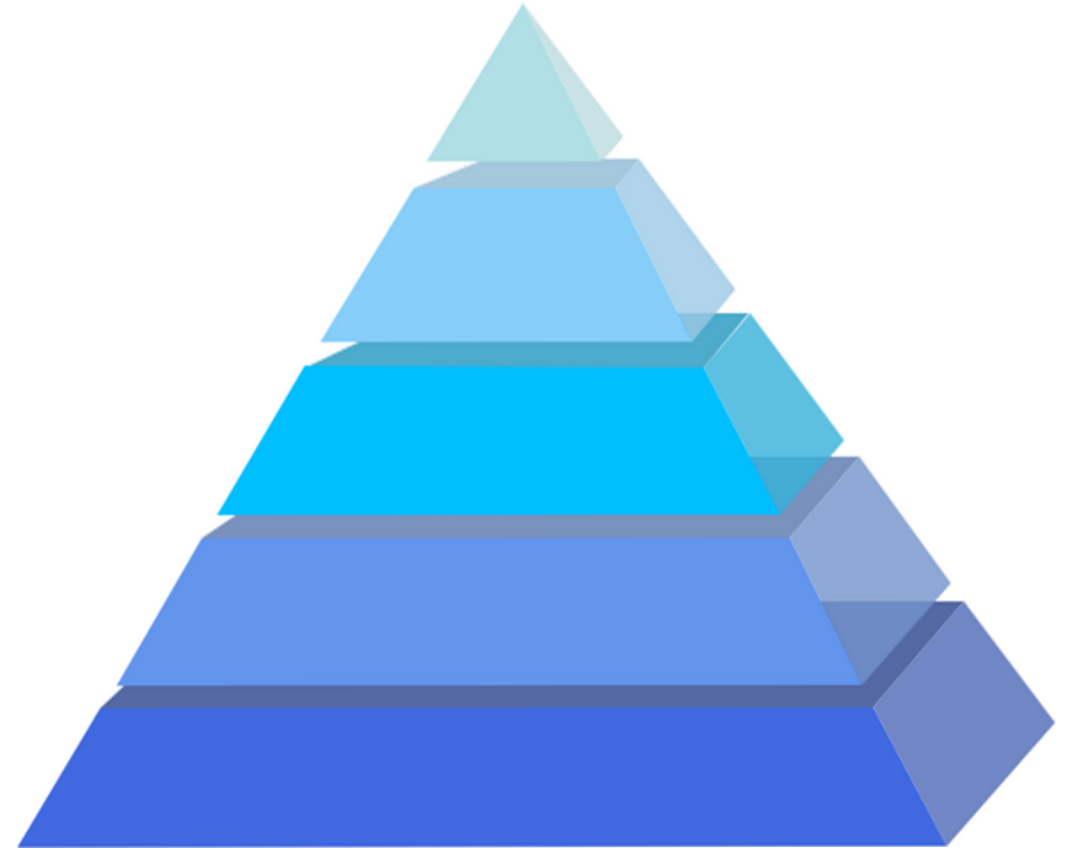
```
num_rows = int(input("How many rows? "))
i = 1
while(i < num_rows):
    print("*****")
    i = i + 1
```

Example Output

```
How many rows?
> 4
*****
*****
*****
```

Pillar 3: Hierarchy of Skills

Learning to write programs is a many-layered skill. Lessons and assessments should progress through the hierarchies of skills and knowledge.



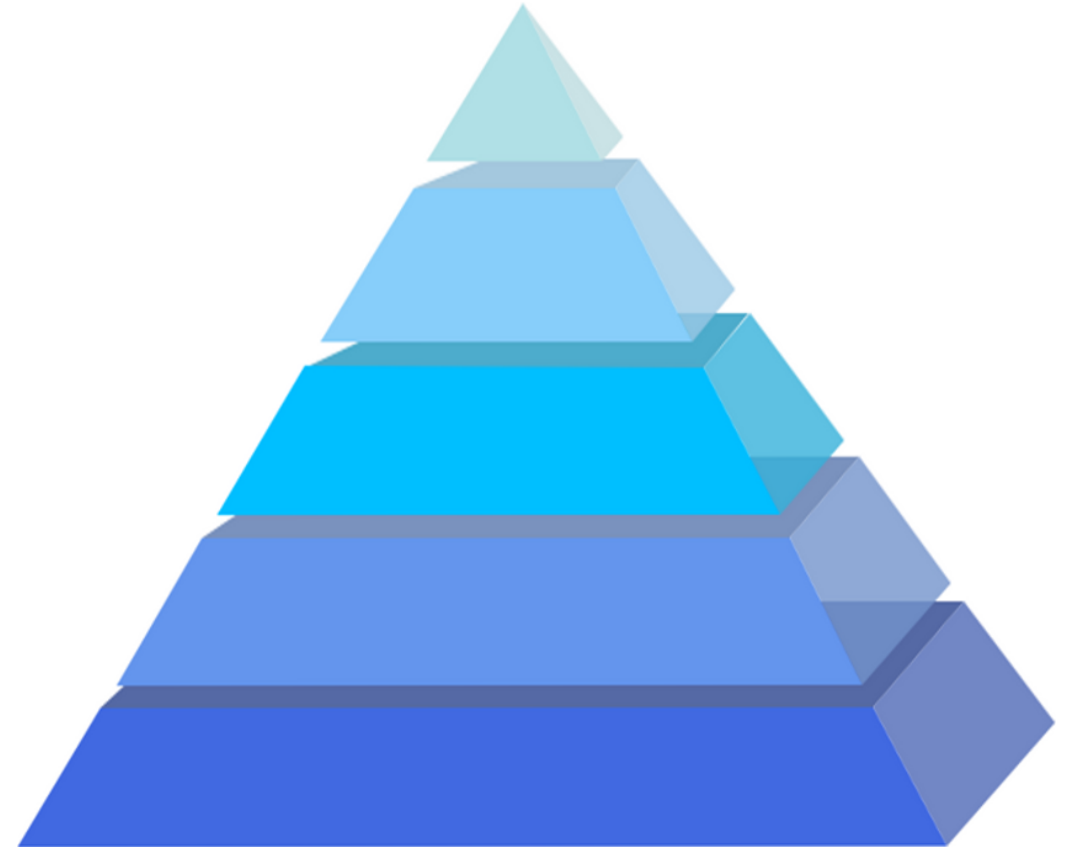
Pillar 3: Hierarchy of skills

Bloom's Taxonomy

Levels of abstraction in defining CS concepts

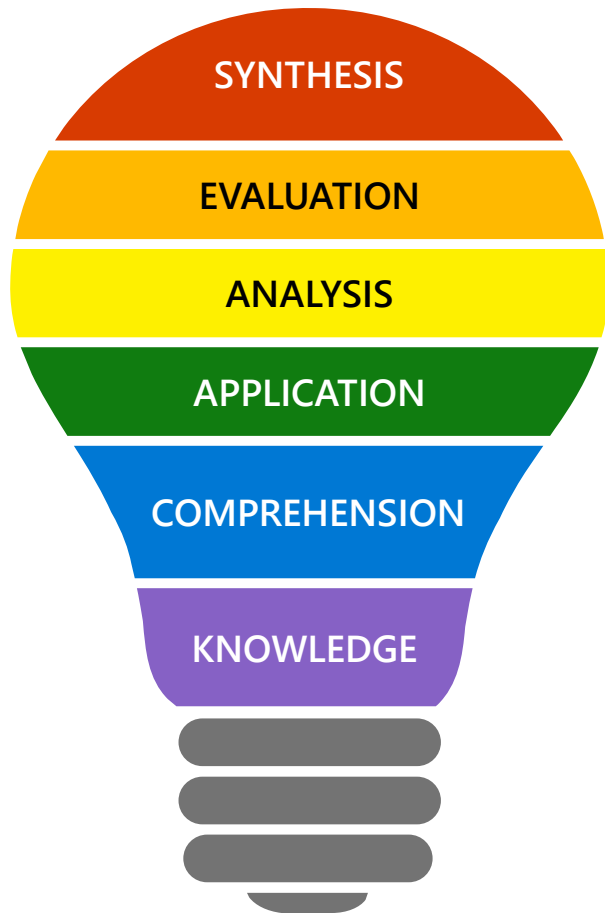
- Definitions
- Syntax
- Patterns
- Generalization

Assessment tasks that map to these hierarchies



Bloom's Taxonomy in CS

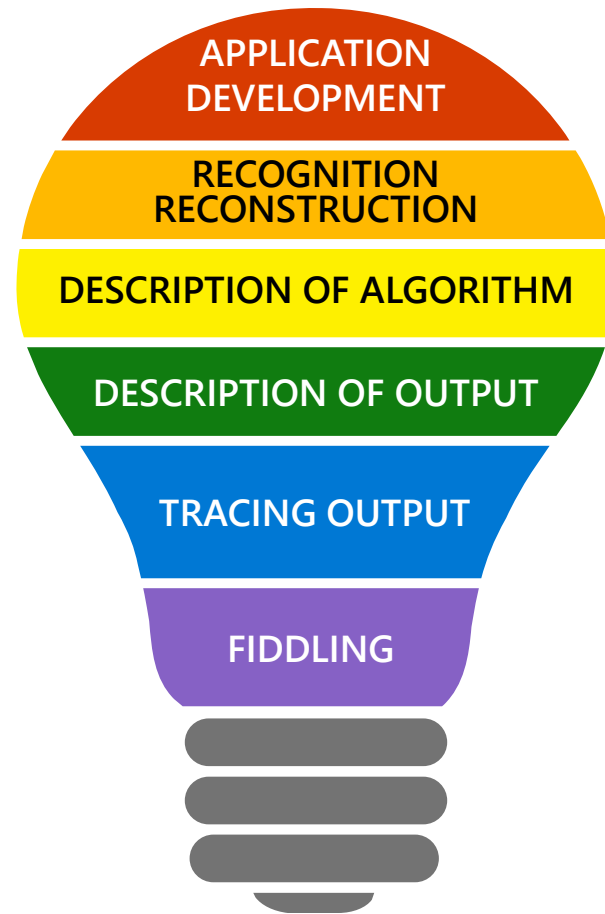
General – Bloom's Taxonomy



Higher Order Thinking Skills

Lower Order Thinking Skills

CS-Specific



Higher Order Thinking Skills

Lower Order Thinking Skills

Literal Translation

Translate each of the following pseudocode statements into Python code:

1. Initialize variable "max" to 0
2. Iterate through each value in the list "numList"
3. Print the variable "max"

Summary

In words, describe what this code does:

```
max = numList[0]
foreach n in numList:
    if n > max:
        max = n
print(max)
```

Code Tracing

What value will be printed?

```
numList = [32, 100, 31, 5]
max = numList[0]
foreach n in numList:
    if n > max:
        max = n
print(max)
```

Code Completion

Insert the missing expression so that this program prints the maximum value from a list stored in the variable numList:

```
max = numList[0]
foreach n in numList:
    if {MISSING CODE}:
        max = n
print(max)
```

Parsons Problem

Reorder the following lines of code so that this program prints the maximum value from a list stored in the variable numList:

1. max = n
2. print(max)
3. foreach n in numList:
4. max = numList[0]
5. if n > max:

Synthesis

Write a program that prints the maximum value from a list stored in the variable numList:

Pillar 4: Inclusive Teaching

Teaching practices based on an understanding social, economical, cultural, etc., aspects of a student's life to increase both the participation and achievement of students, especially from underrepresented groups.



Pillar 4: Inclusive Teaching Includes



Universal Design

Awareness of context

Connecting with students

Wise Feedback

Curriculum adaptation

Culturally Responsive Teaching

Becoming a Culturally Responsive Educator

Culturally responsive teaching (CRT):
A way of teaching that includes students' cultural references in all aspects of learning to increase the participation and achievement of students from underrepresented groups.

Identify your biases

Motivate students to learn



Increase student understanding of subject

Wise Feedback

Emotions affect learning

Effective feedback:

- Is unique to the student's learning situation
- Balances faith in student potential with honesty about current performance
- Reassures students that they will not be stereotyped or doubted as less capable



Three elements of “Wise Feedback”



High Standards

- Affirm holding high standards
- Mistakes are a sign of the high demands of the task/class, not of (perceived) low capability



Personal Assurance

- Assure the student that he or she is capable and can improve with effort
- Reference past successes or progress



Actionable Steps

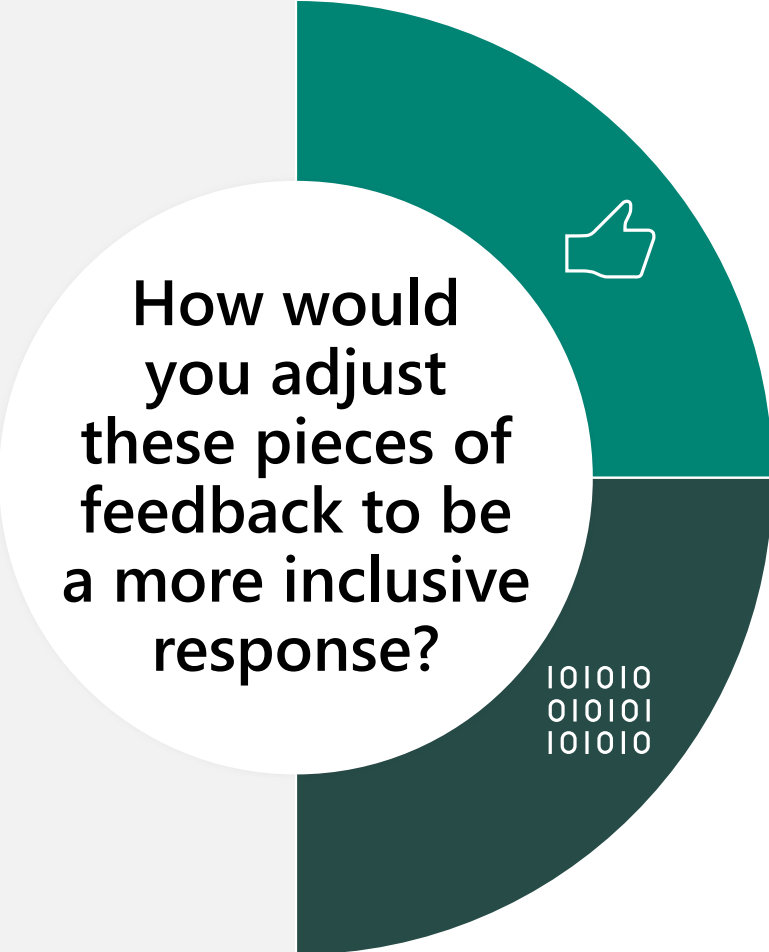
- Give specific, actionable steps to work on
- Instructive rather than evaluative
- Corrective rather than vague

Critiqued Examples of Feedback

Feedback	Improved Feedback
<p><i>"Great job on your code."</i></p>	<p><i>"Great job on problem 3. I love how you used comments to make a plan before you wrote the code. This approach will help you succeed as we move onto longer and harder assignments."</i></p>
<p><i>"When you added the new condition in Part B you didn't follow the correct procedure."</i></p>	<p><i>"I can see what you were going, and your thought process is correct it just needs a small adjustment. You correctly added the condition in part A, but not part B. Part B is tricky but I know you can do it. Look back at the lab assignment sheet to make sure you understand the expectations for Part B, and check with a peer if you still aren't sure."</i></p>

- Red** = Personal Assurance
- Green** = High Standards
- Blue** = Actionable Next Step

Activity: Wise Feedback



How would you adjust these pieces of feedback to be a more inclusive response?

Good job. Please review the notes I put on in the rubric evaluating your project.

I looked over your code. It needs substantial revision, as you will see when you read my comments.

Learn more about Inclusive Teaching

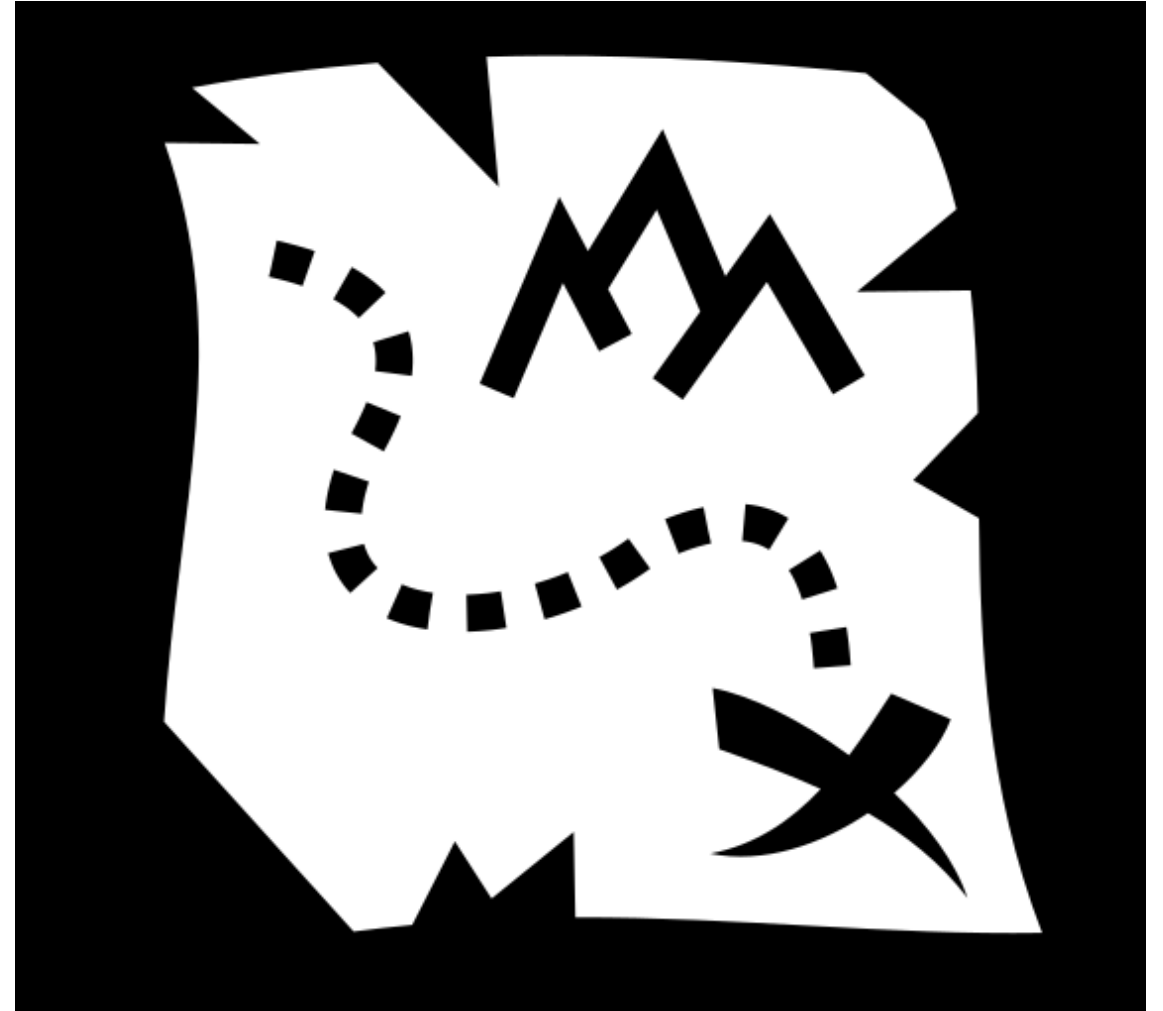


Integrated into the rest of our training sessions



Integrated into Classroom Plan

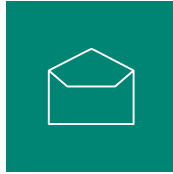
- Daily Handoff
- Weekly Sync
- Classroom Flow & Handling Challenges



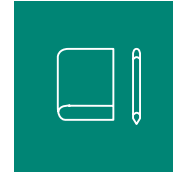
Classroom Plan Check-in #3



Read “Engaging Parents/Guardians” which is the last section of the classroom plan.



Open the linked **Sample Letter to Parents/Guardians**



As a group, begin drafting your own letter to send home with students during the first days of school



YOU HAVE 8 MINUTES.

Co-Teaching

**Not just for the
Co-Teaching Model**



Strategies for Effective Co-Teaching



Think about...

What are the potential benefits to having multiple instructors in a classroom?

What are the potential challenges in having multiple instructors in a classroom?

What extra steps might be necessary when planning to run a class with multiple instructors?

Activity: Mini-Research Project

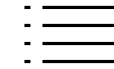


You will be given a co-teaching model.



Working with a partner, write down:

- How does the model work?
What is each instructor's role in the model?
- When might each model be most useful in a computer science class?
- What are the risks or challenges in using each model?



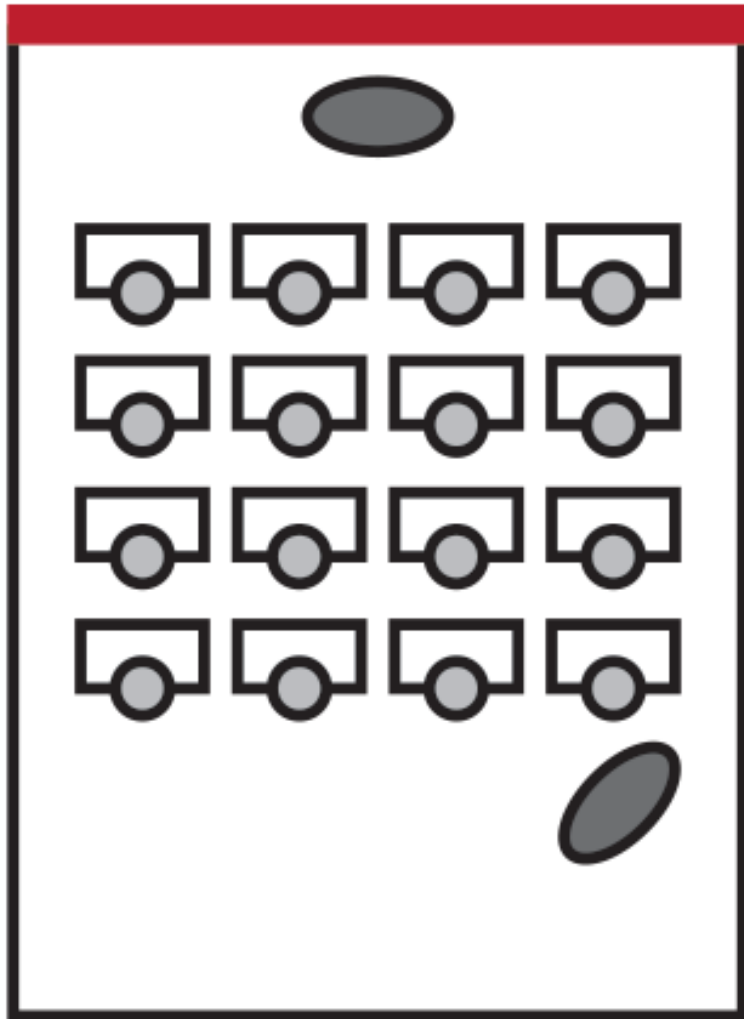
Fill in the appropriate row in your graphic organizer.



You have 5 minutes.

Activity Packet page 5

Model 1: One Teach, One Support



Good for

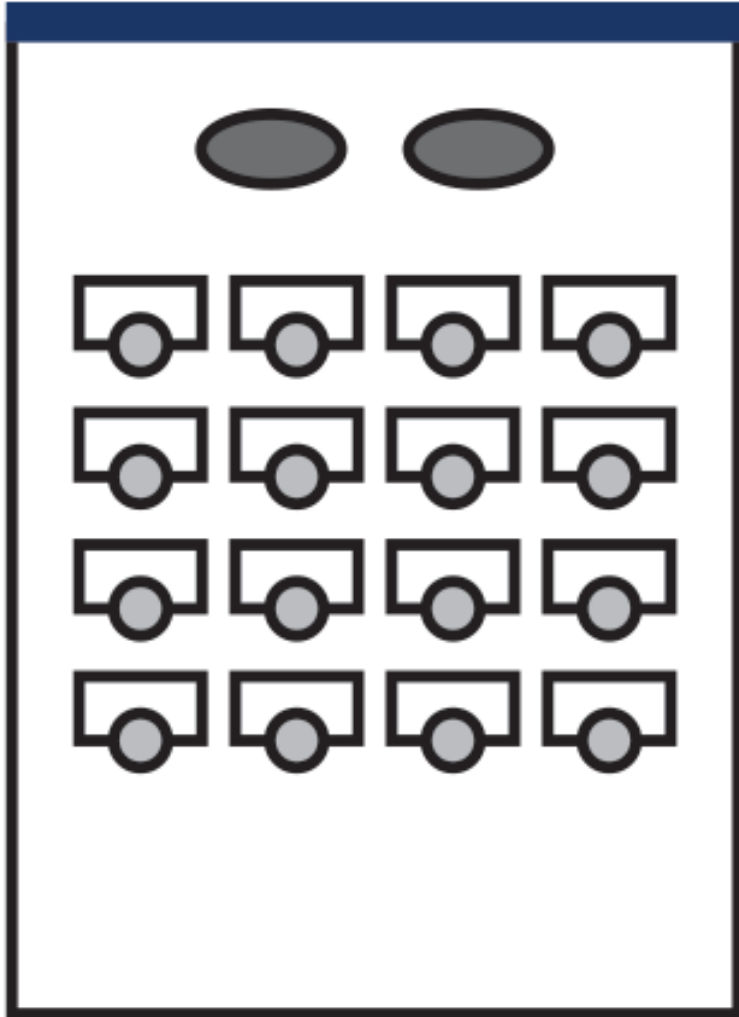
Introduction of new material
Review of exercises/activities

Challenges

Role of supporter

- Hand out papers
- Drive the demo/computer
- Monitor student engagement
- Chime in with useful questions or comments
- Give out raffle tickets

Model 2: Team Teaching



Good for

Asking model questions

Paraphrasing

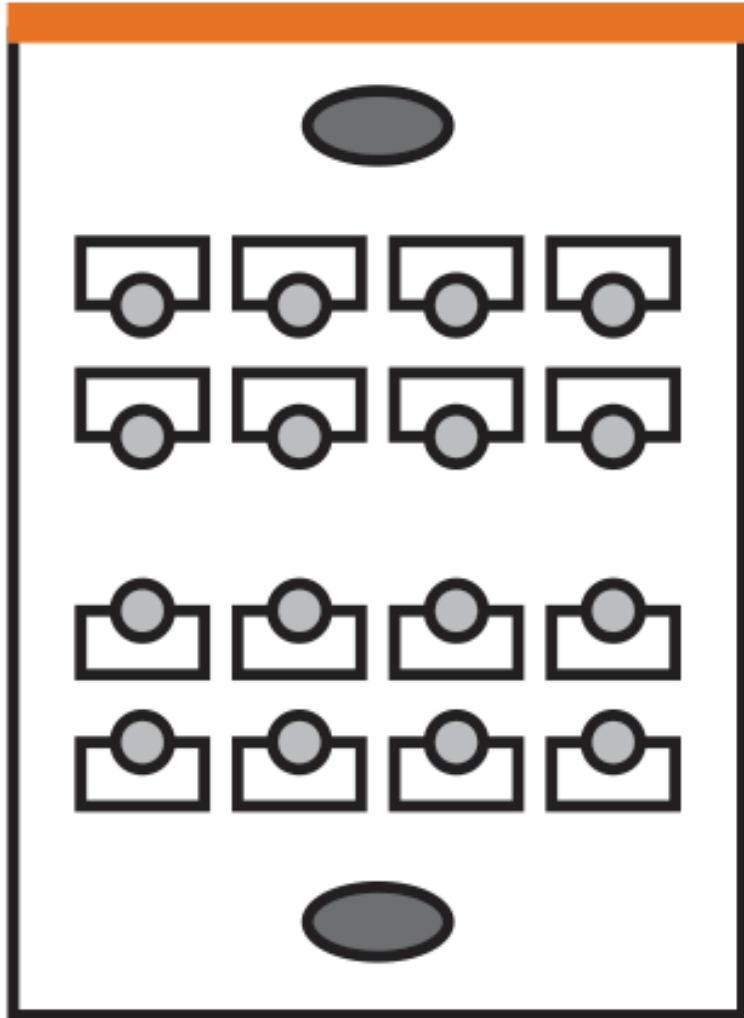
Debates

Challenges

Clear transitions

Student point of focus

Model 3: Parallel Teaching



Good for

Class discussions

Highly interactive activities

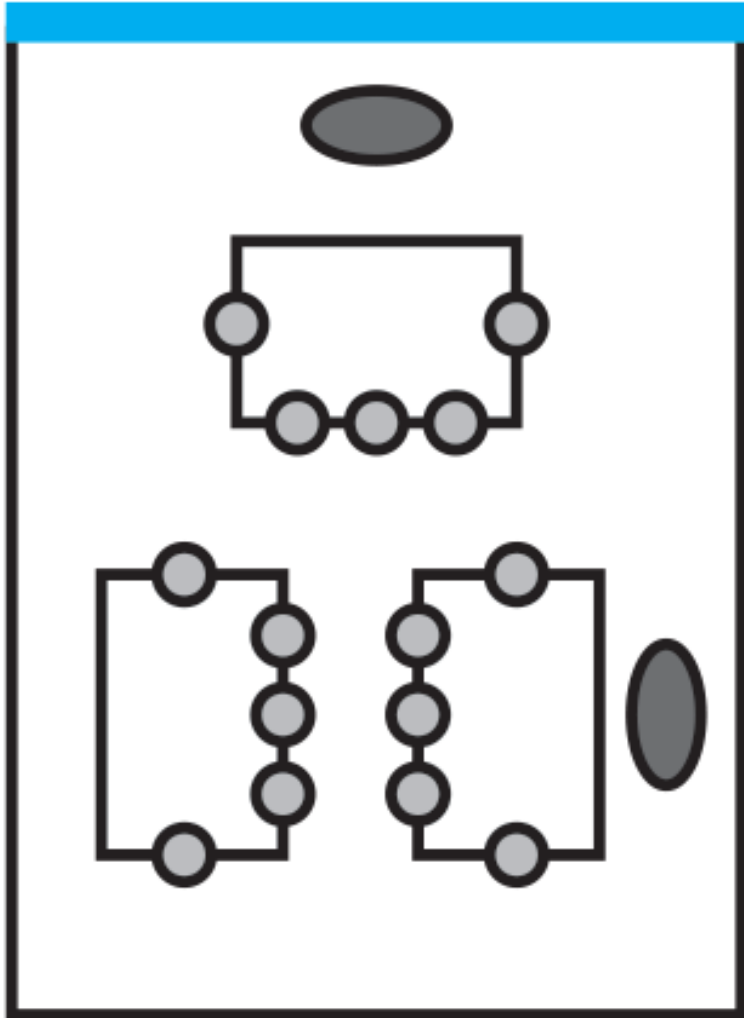
Challenges

Ensuring consistent content

Timing/pacing

Noise levels

Model 4: Station Teaching



Good for

Lab work

Review days (e.g. for tests)

Student choice of activity/topic

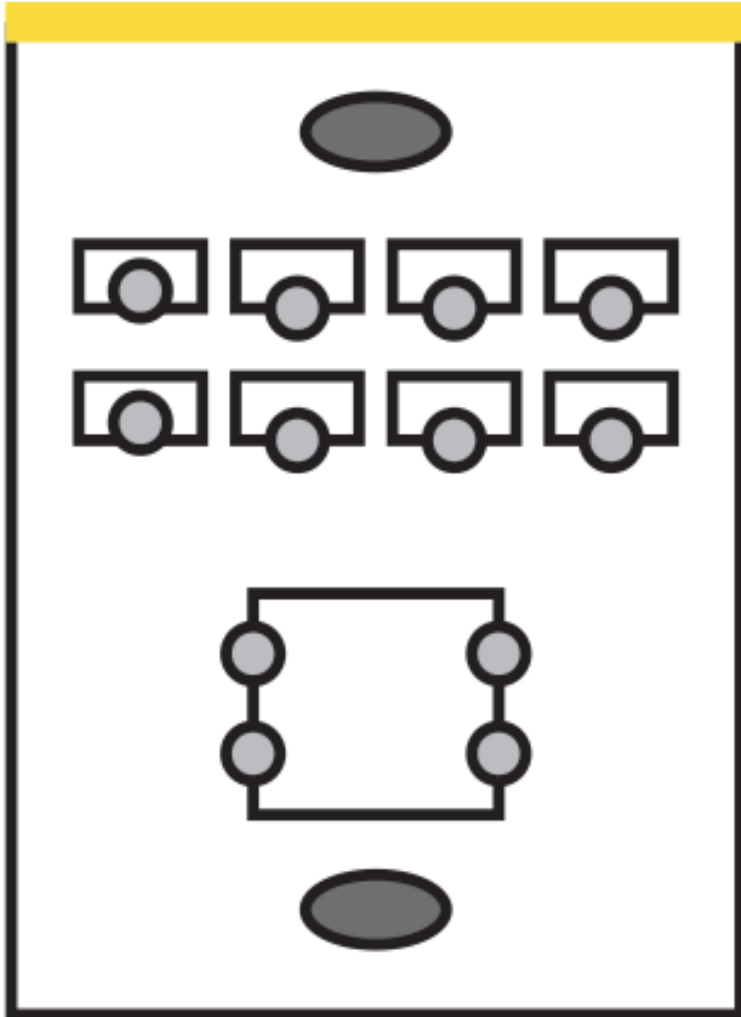
Multi-phase assignments

Challenges

Timing

Equal student attention

Model 5: Alternative Teaching



Good for

Catching up absent students

Extra help/support

Enrichment

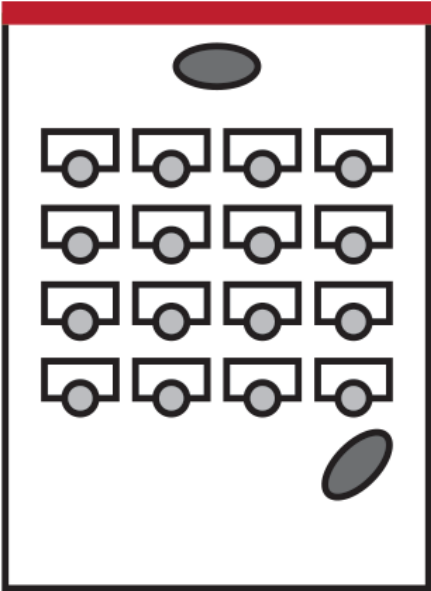
Challenges

Avoiding embarrassing or singling out students, or signaling "low expectations"

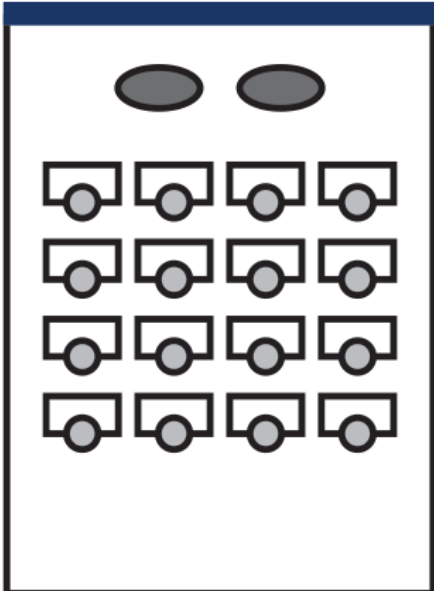
Appearing opt-in while nudging students

Missing important content from "main" group

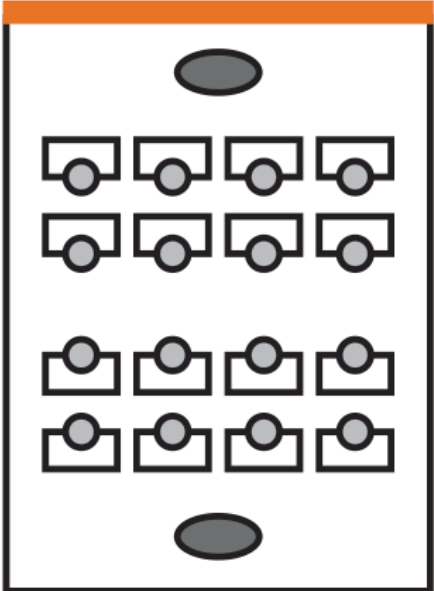
Co-Teaching Recap



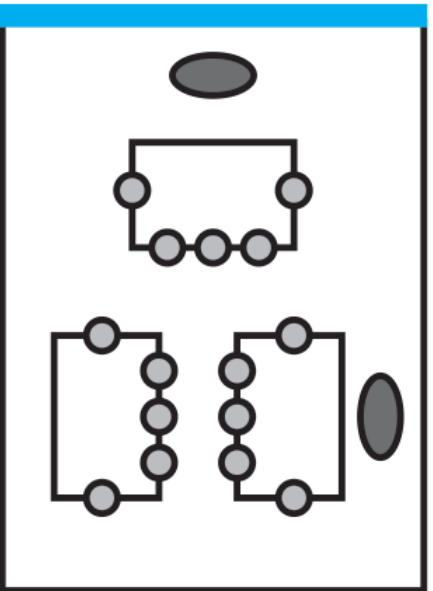
One Teach,
One Support



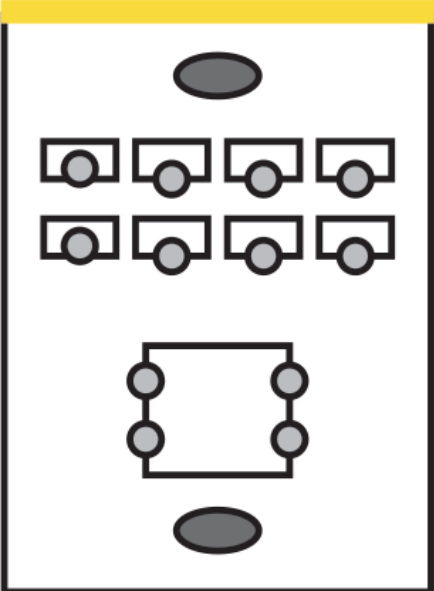
Team
Teaching



Parallel
Teaching



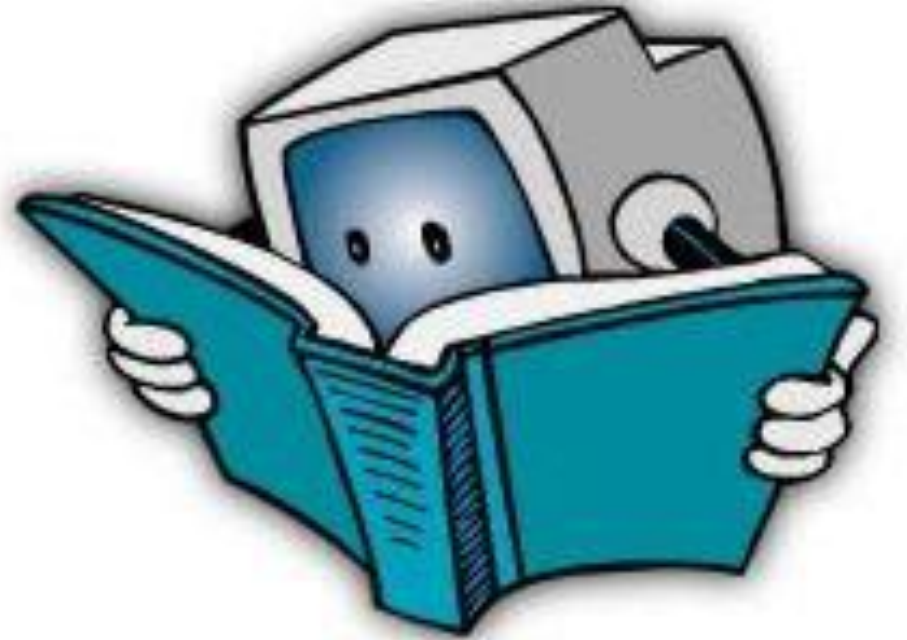
Station
Teaching



Alternative
Teaching

Use **ALL** of these in your classroom!

Preparing to Teach



Hello, World: Lesson planning

Review the “Lesson Planning” document located on the Classroom Plan website:
<https://tealsk12.org/classroomplan>

Complete the worksheet as if planning the early lesson below:

- AP CS A: Lesson 1.03
- AP CS P (code.org): Unit 1, Lesson 4
- Intro CS: Lesson 1.1

Focus on

Adapting the lesson to be relevant to *your* students

The comfort level of each member of the team with the content and/or teaching methods

The role each team member will take during the lesson



YOU HAVE 15 MINUTES.

Putting It All Together

So far, we've discussed

- Your goals, strengths, and worries
- Roles and responsibilities
- Strategies for communication
- Computer Science Pedagogy
- Inclusive Teaching
- Co-teaching models
- Planning lessons together

These are key elements of making your TEALS team effective

Next up for classroom teachers

- Team meetings
- Regional Manager check-ins and Mock Teaching
- Recommended events:
 - Stick around for this afternoon!
 - Session 2
 - Session 3

Exit Ticket – only people who are leaving now

<https://aka.ms/TEALSSummerTraining1>



Student-Centered Learning

"What matters is not what the teacher teaches...

but what the student learns."



“What matters is not what the teacher teaches... but what the student learns.”



Take 3 minutes and discuss with neighbors your reactions to this quote from the Do Now this morning.

Diving Back In

During this lesson, you have two jobs:

1

Be active learners of the content (take notes, participate, etc.)

2

Think about the *structure* and *format* of the lesson

Take notes **IN YOUR NOTEBOOK** on both



Guiding Questions

Use these questions to guide your thinking (also in your packet)

1

How is the lesson structured? What parts or phases of the lesson can you identify? Roughly how much time was spent in each phase?

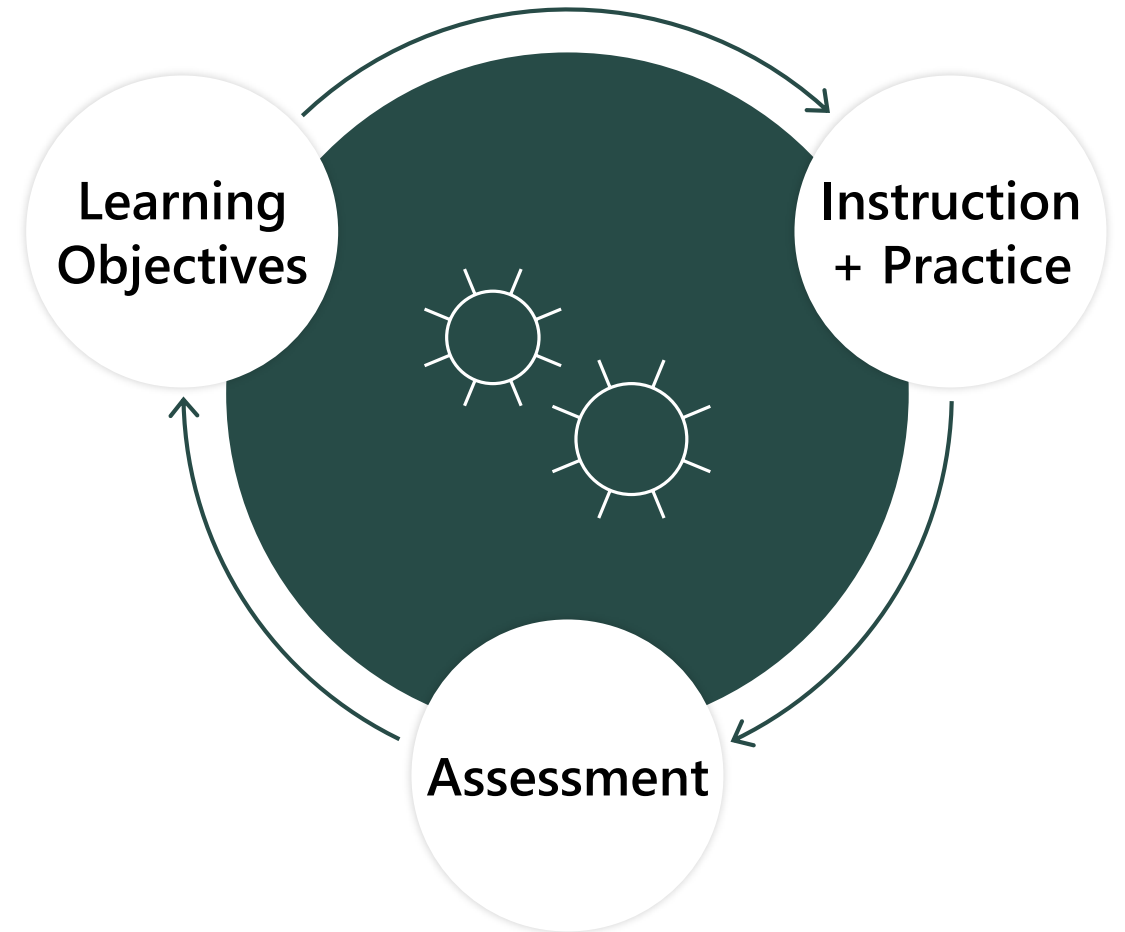
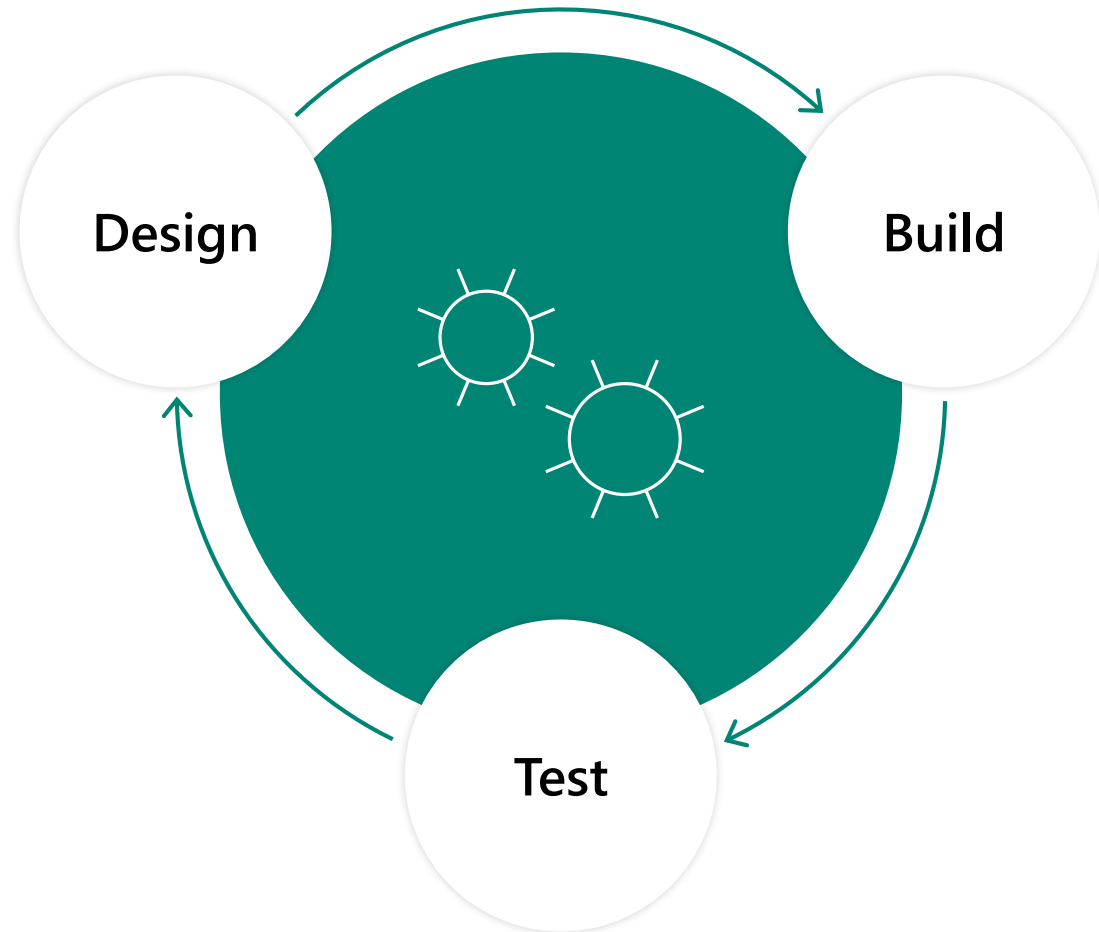
2

What methods for presenting new information do you notice? Which of these did you find most engaging? Why?

3

What types of questions are asked? How are these questions asked? How was someone chosen to answer the questions?

The Teaching Process



Learning Objectives

Students Will Be Able To...

[thinking or doing verb] [standard or concept]

1

Students will be able to create a program that accepts user input and returns data

2

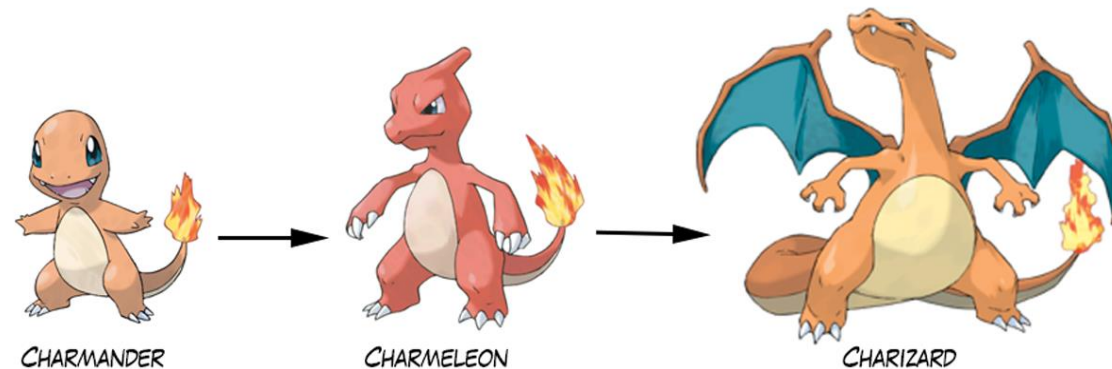
Students will be able to design and implement a basic model of gravity

Training Objectives

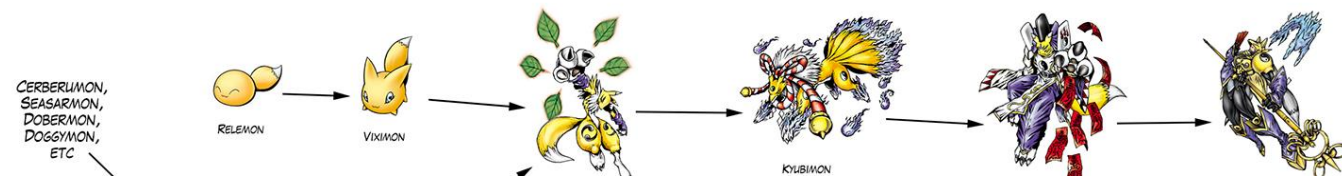
SWBAT (Students [a.k.a. you] Will Be Able To)...

- List the roles and responsibilities of each member of your teaching team
- Describe your team's plans for communication and working together
- Explain how your teaching team will use each member's strengths and weaknesses to achieve success in your classroom
- Define "pedagogical content knowledge" and list some CS-related examples
- Create a classroom culture where **all** students can be successful

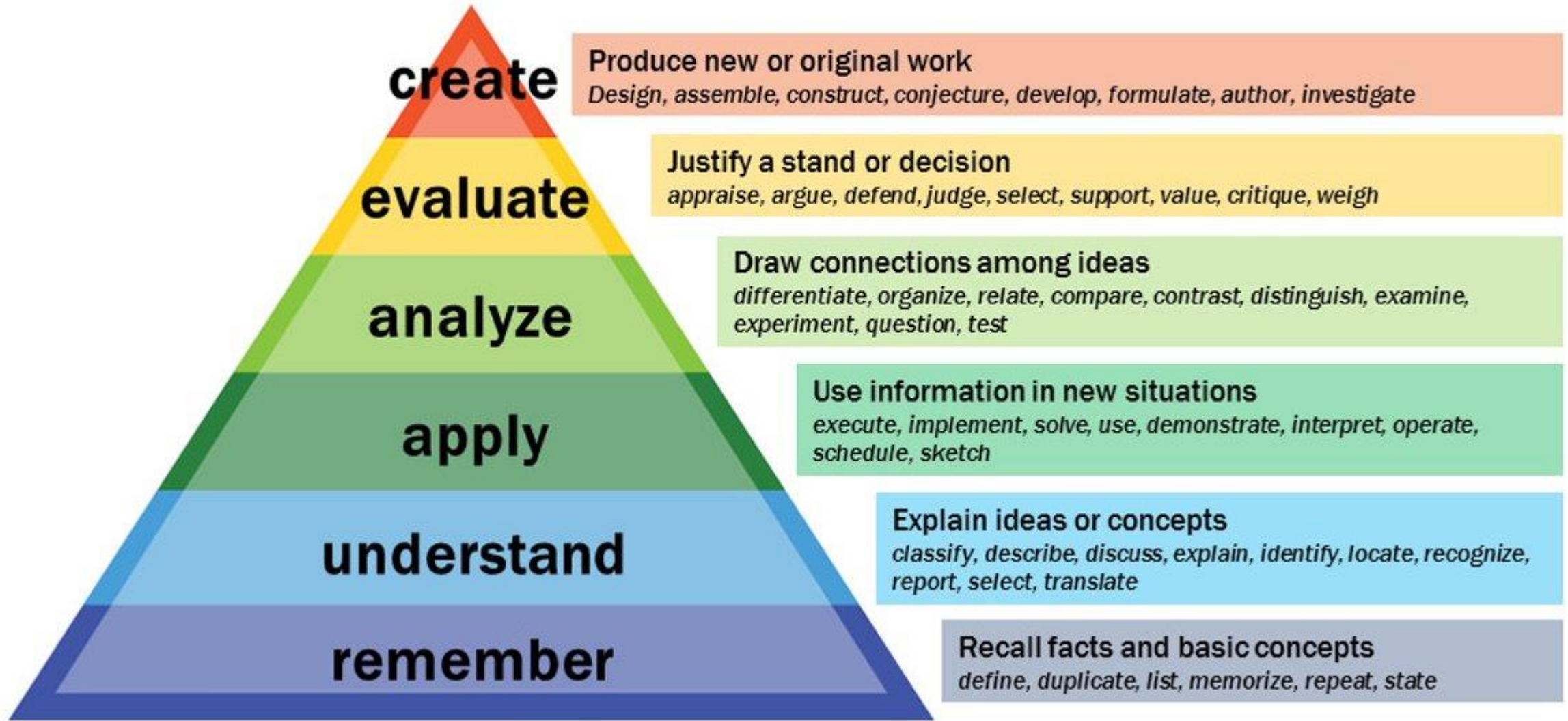
POKEMON EVOLUTION



DIGIMON EVOLUTION



Bloom's Taxonomy



Sample Learning Objectives

AP CS A Lesson 1.3

Students will be able to...

- Correctly assemble a complete program with a class header, body, and main method
- Correctly use print, println, and escape sequences

Intro Lesson 1.3

Students will be able to...

- Construct simple algorithms to draw shapes
- Convert algorithms into SNAP programs

Learning Objectives Activity

Complete the activity in your packet.

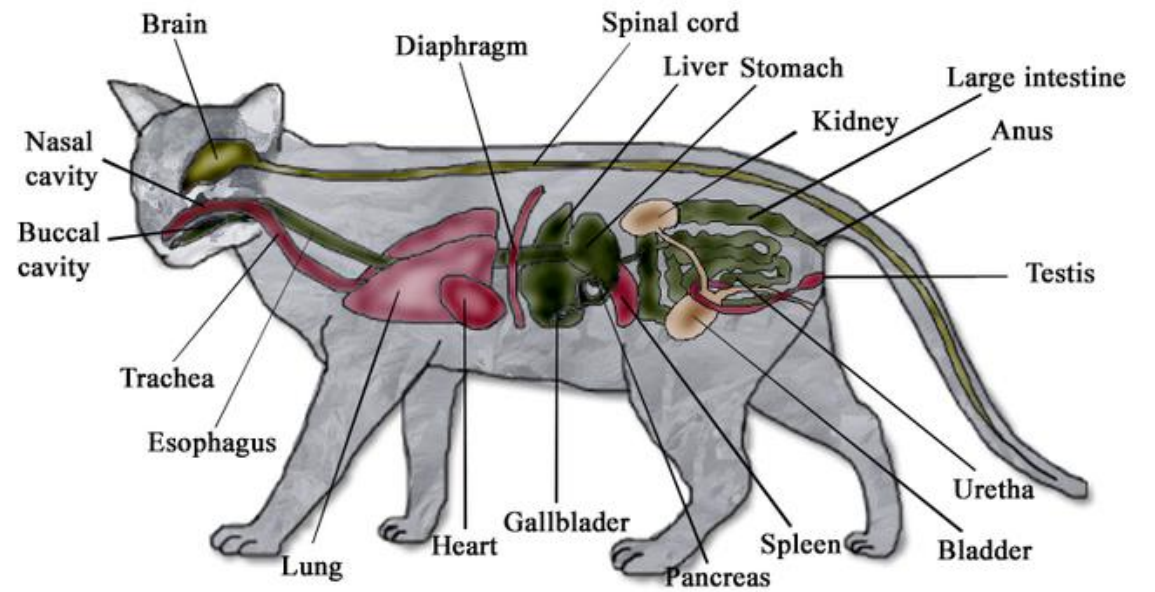
You may work in pairs.



You have 7 minutes!

Activity Packet page 6

Anatomy of a Lesson



Let's Reflect...

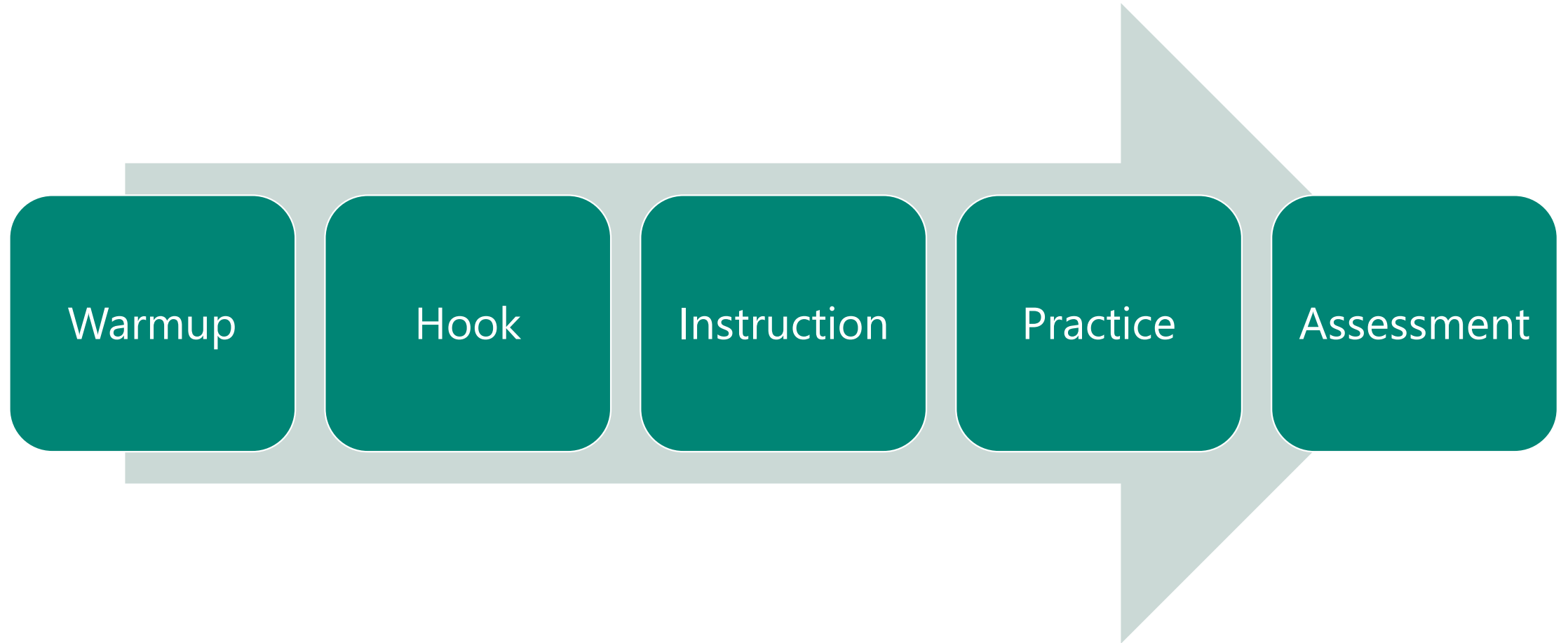
Think about the lesson we just covered on student centered learning and learning objectives.

How would you describe the structure of that lesson?

(Hint: check your notes...)



Anatomy of a Lesson



Warm-up (aka Do Now, Bellwork, Boardwork, Bellringer, ...)

- Set student mindset
- Set up the topic for the day
- Review old content or lead into new
- Often a single homework or practice problem or an open-ended question
- Short, achievable by all



Instruction

Can be new content or review

25% of class time

Not just lecture!

- Definitions
- Explanations
- Demos & examples
- Group activities
- Research



Practice

- 75% of time
- Could be...
 - Worksheets
 - Group presentations
 - Labs
 - Projects
 - Textbook problems
- Can be combined with instruction

```
1
2 public class Pokemon {
3     private int hp;
4     private int attack;
5     private int defense;
6     private int specialAttack;
7     private int specialDefense;
8     private int speed;
9     private String type;
10
11     public Pokemon(String type, int hitpoints, int a, int d, int s){
12         hp = hitpoints;
13         attack = a;
14         defense = d;
15         speed = s;
16     }
17
18     public void consumeVitamin( int hpUp, int protein ){
19         hp += hpUp;
20         attack += protein;
21     }
```


Assessment

Anything that helps gather data about student achievement

Extremely important, but often overlooked

Two types:

Formative Assessment

Check for understanding during the learning process

Multiple times **per lesson**

Smaller, fewer points (if any)

Quiz, labs, discussion, questions

Summative Assessment

Prove what students know at the end of the learning process

Once or twice per unit

Larger, more points

Tests, projects

Activity: Seeing the Pieces

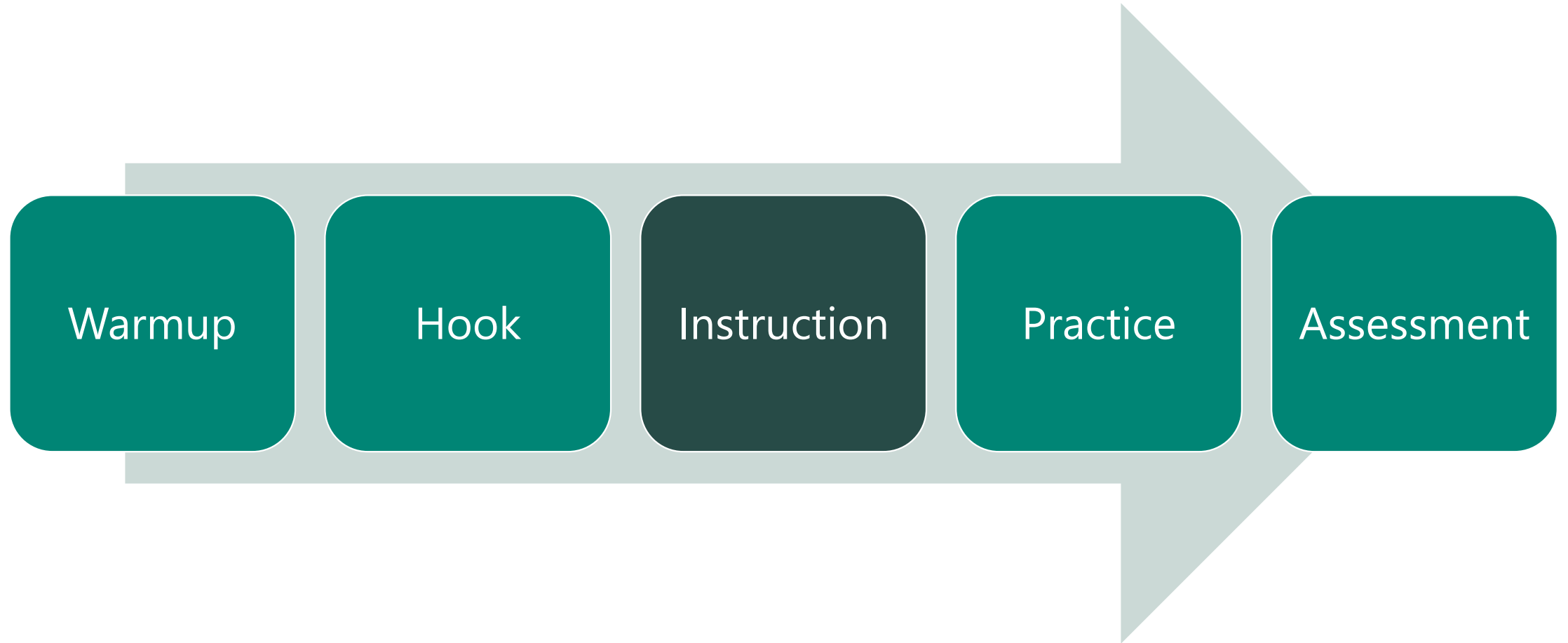
- 1 Navigate to the lesson aka link indicated in your packet
- 2 Read the lesson plan
- 3 Determine what in the lesson represents each piece: warm-up, hook, instruction, practice, assessment
- 4 Once you are finished, **switch packets with a partner and peer-grade**



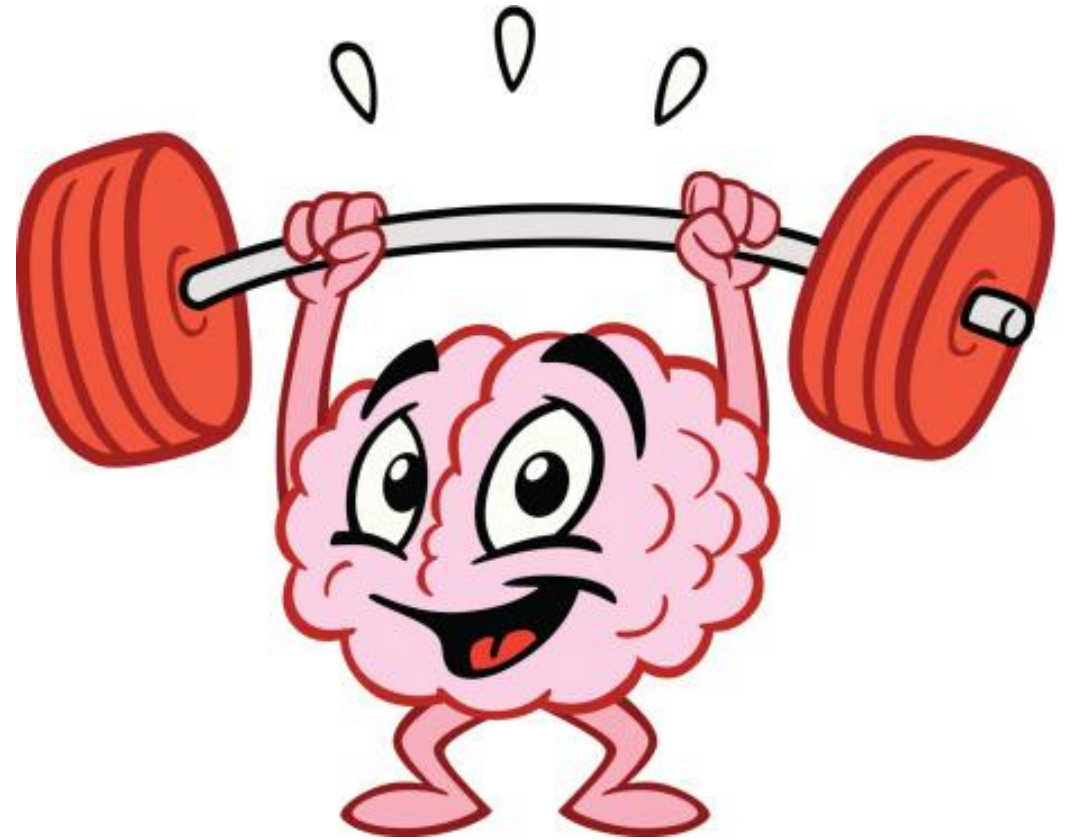
You have 7 minutes!

Activity Packet page 9-10

Anatomy of a Lesson



Instruction: Active Learning



How Learning Works

Nope.



Yep.



Not Just Lecture

How long can you focus on a lecture?



Instructional Techniques

- In the next **two minute** write down (in your notebook) as many instructional techniques as you can think of...

Discussion

Research

Journaling/
Writing

Exploration/
Discovery

Demo

Student
Presentation

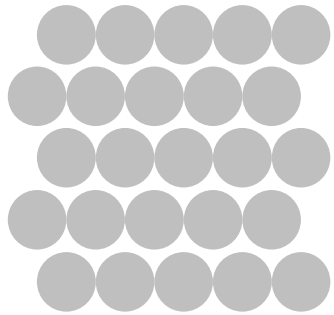
Peer
Instruction

Walkthrough

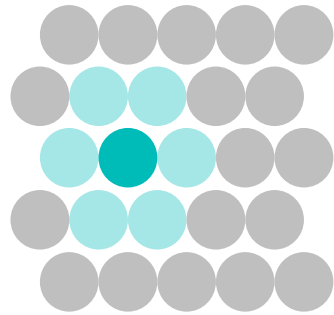
Small Group
Work

Roleplay

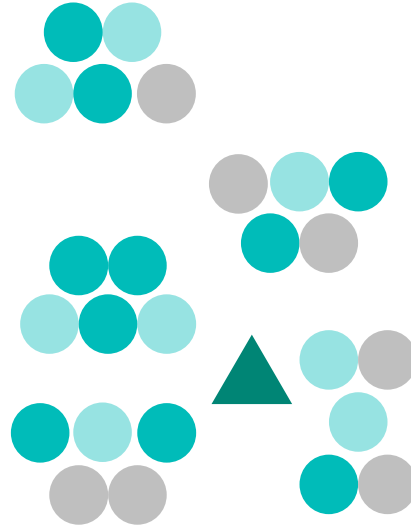
Engagement



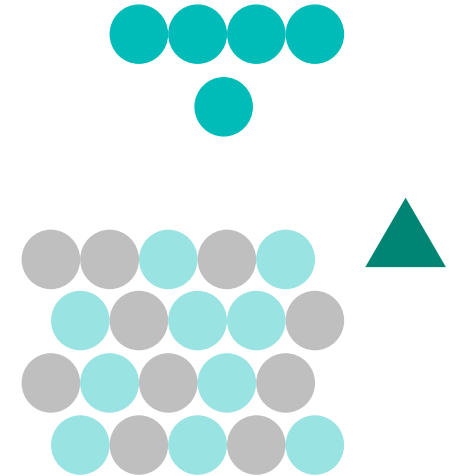
Lecture



Question



Group
Discussion

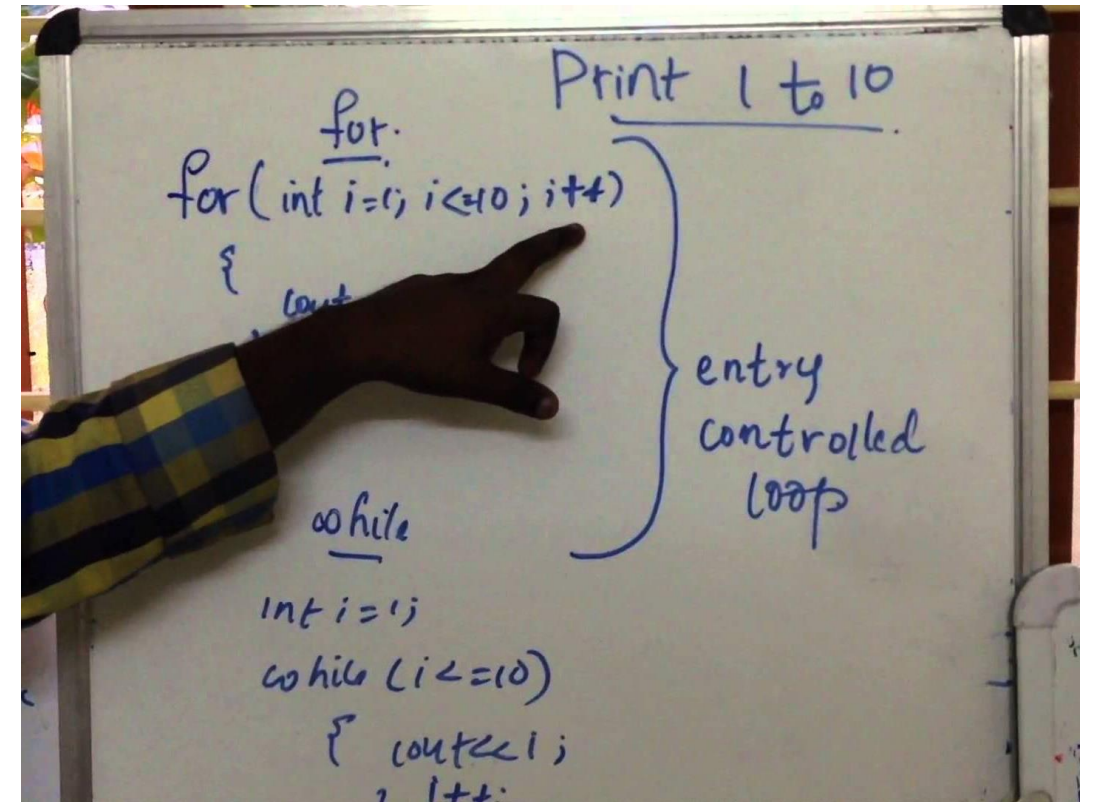


Group
Presentation

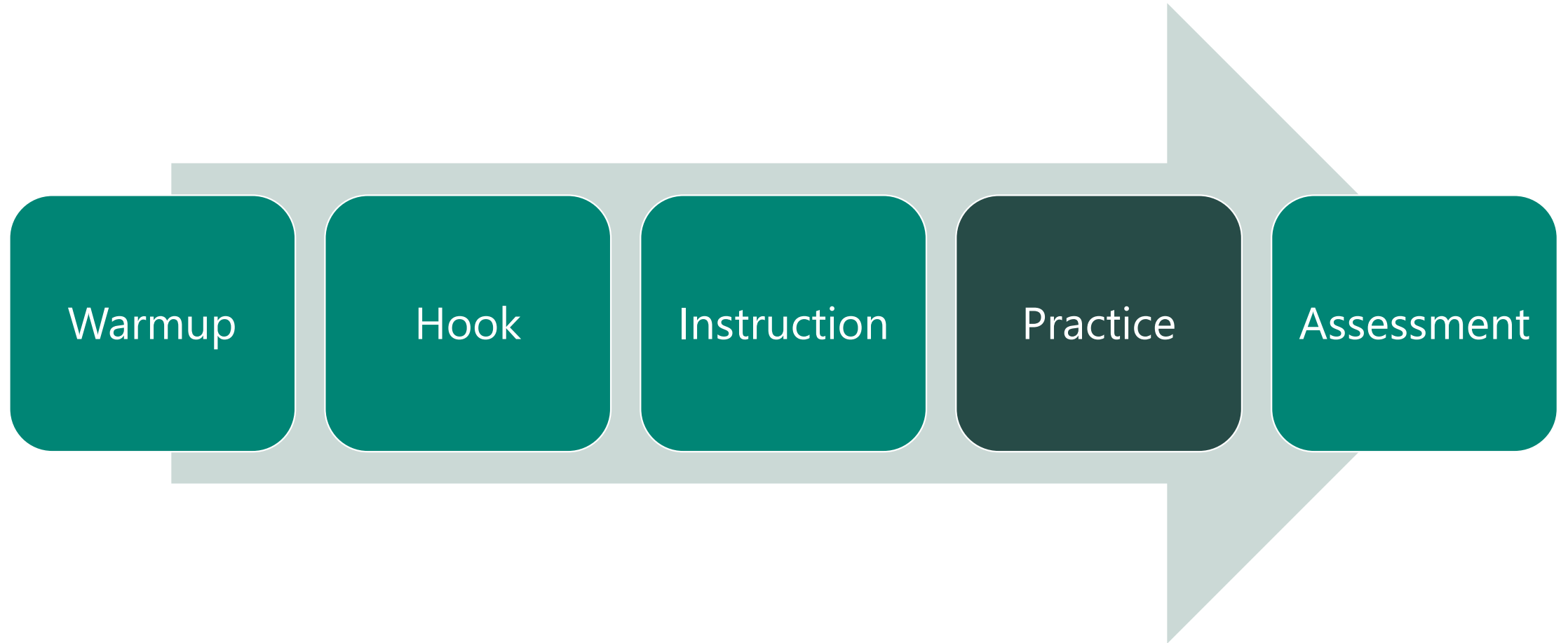
Whiteboarding and Active Learning

Question

What are the **benefits** of using the whiteboard instead of slides during instruction?



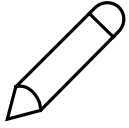
Anatomy of a Lesson



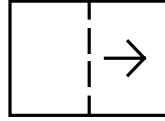
Practice: Lab Management



Follow the Directions



You'll need a pen.



Use either side of your index card.

Giving Clear Directions

Effective Directions...

- Are complete
- Are given in a consistent format
- Are brief
- Are actionable (and specific)
- Include time expectations
- Are repeated back by the students to confirm understanding

When Giving Directions...

- Prime your students
- Get their attention first
- Deliver clear & concise instructions
- Follow through (don't change your instructions)
- Reinforce your instructions

*length of class (mins) * number of instructors*
number of students in class



The Way Of The Meerkat

- ~30 sec – 1 minute in each interaction
- Triage, actionable feedback, move on.
 - Do not get monopolized.
- See every student's progress at least twice in 1 period *even if they don't ask for help*
- Collect formative assessment data
 - By the end of class, your team should know each student's progress on the lab

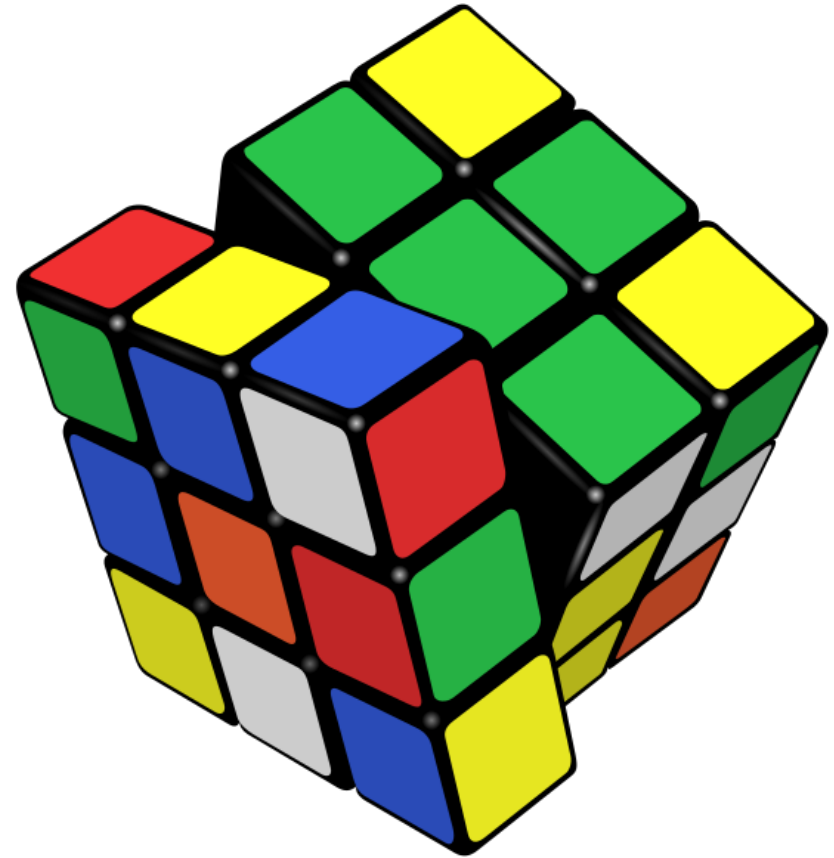


Lab Time MANAGEMENT Tips

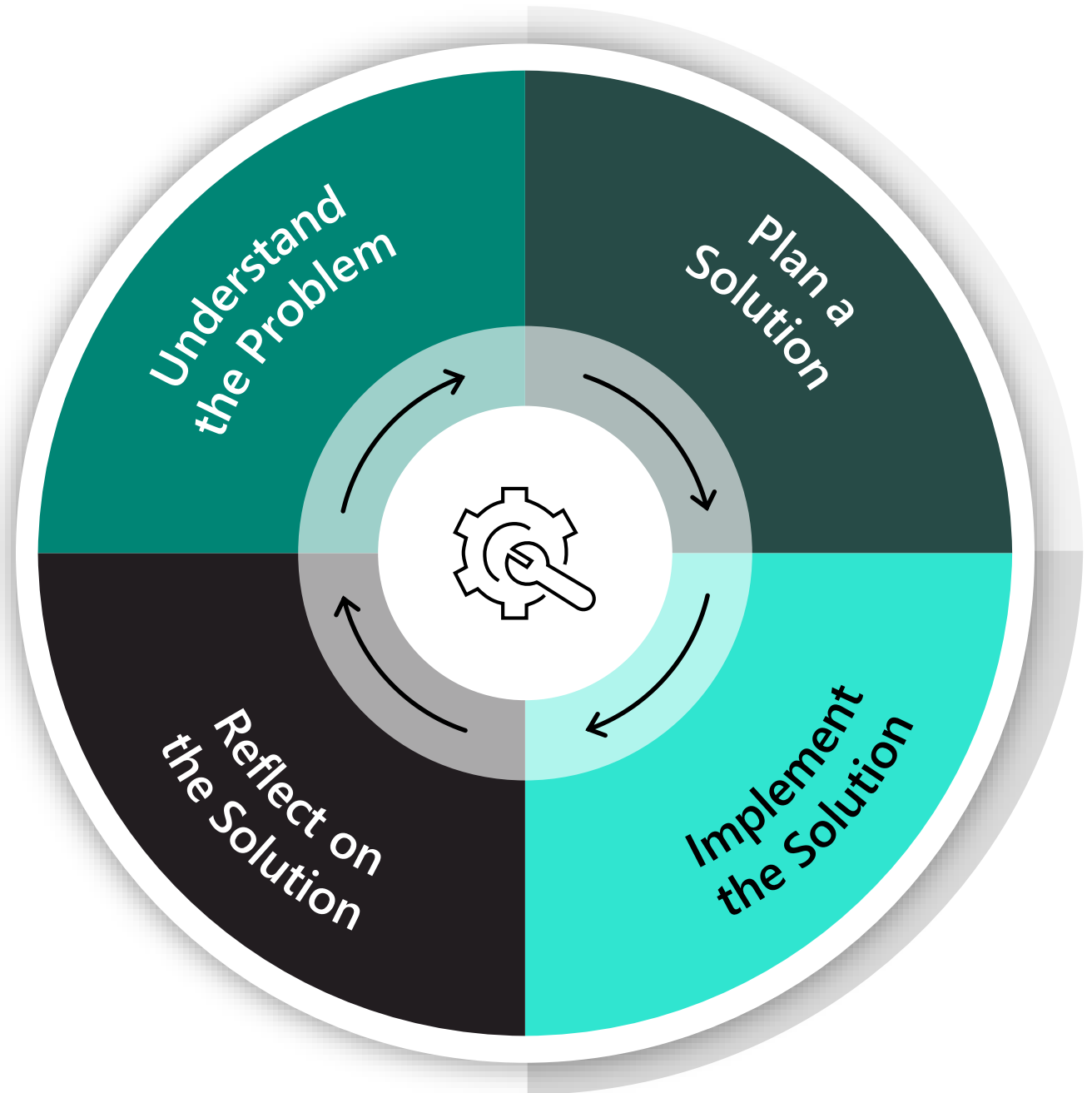


Tips	Action
Transition Cue	Use music or a “ call and response ” as a cue when it is time to transition to another activity or format.
C2B4	Expect students to “ see two peers ” and/or to check their own notes prior to requesting instructor help.
Requesting help	Create a system for students to passively request help, like showing a red/green cup or post-it or keeping a queue on the whiteboard.
Staying on task	Monitor whether students are working on the right task and gently redirect them by asking about their progress. Avoid escalation.
Track progress	Establish a shared location to record information about each student’s progress during lab.
Divide & conquer	When multiple instructors are present, consider splitting up and using lab time for notebook checks or conferencing with individual students.

Practice: Problem Solving Skills



Four Steps To Solving Any Problem



Let's Work Through a Problem Together...

Write a program that prompts the user for a number between 1 and 10. If the number is within range, calculate and output its factorial; otherwise, tell the user that the input was out of range. This is a one time run, only one loop is needed.

Step 1: Understand the Problem

Some questions you can ask

- What are you trying to find or do?
- Can you restate the problem in your own words?
- Can you think of a picture or diagram that might help you understand the problem?
- Is there enough information for you to find a solution?
- Do you understand all the words used in stating the problem?
- How will you know you have a correct solution?



You have 3 minutes to work through this step in your packet.

Step 2: Plan a Solution

Some strategies you can suggest

- Can you write out the steps of how to solve the problem in plain English?
- Are there any constructs you know that relate to the problem?
- Can you break the problem down into smaller pieces?
- Are there any patterns? (logic, looping)
- Can you think of a similar problem you've solved?
- Are there any special cases you need to consider? (eg. edge cases)
- Can you work backwards (bottom-up)?
- Can you introduce a supplemental element that will help you?



You have 3 minutes to work through this step in your packet.

Step 3: Implement the Solution

Some aspects to consider during implementation

- Is each part of your solution implemented?
- Does each part of your solution work? (compared to specification)
- Is your code organized in easy-to-understand modules with comments?
- Did you read your code line-by-line (troubleshooting/debugging)?



You have 3 minutes to work through this step in your packet.

Step 4: Reflect on the Solution

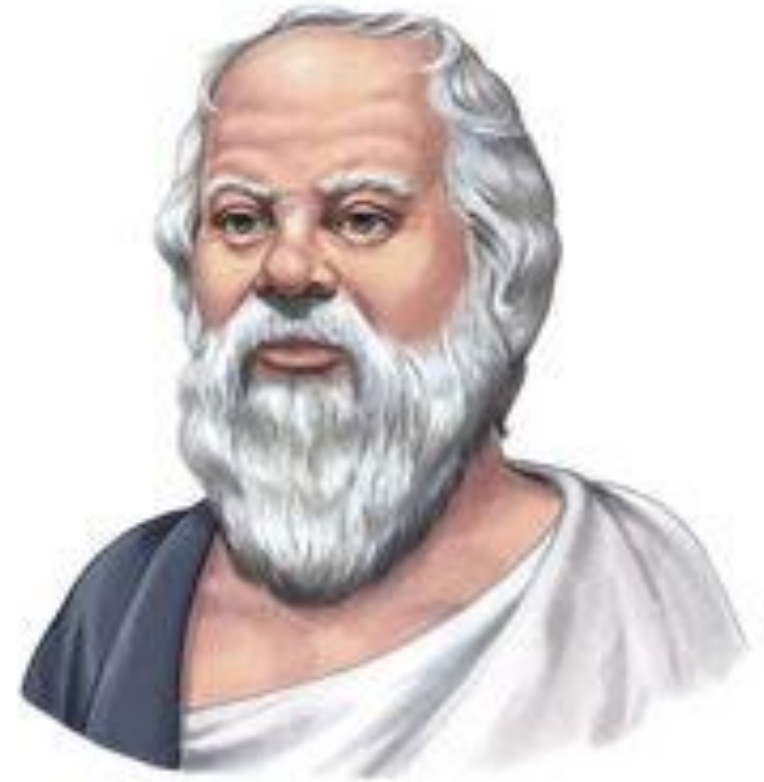
Some questions
to ask when
solution is
implemented

- Does your answer make sense? Does it solve the original problem?
- What worked and what didn't?
- Can you think of another (easier or more efficient) method to solve the problem?



You have 3 minutes to work through this step in your packet.

Practice: The Socratic Method



Questions Only!

In this game, have a conversation with a partner using **ONLY** questions.

Whoever makes a non-questioning statement first, loses.

Topic...

You are in your TEALS class on the first day of school.

<https://www.youtube.com/watch?v=tkxRzV3gtDc>

Socratic Questioning



Diagnose Misunderstanding

- How are you doing?
- What is this supposed to do?
- How does it work?



Ask Leading Questions

- Where in your code do you calculate the sum?
- How many times will this loop execute?
- Will the loop continue when n is 10?



Actionable Next Step

Question or a statement:

- *“Do you know what your next step is?”*
- *“Look at last week’s lab on loops and see if you can find something similar that will help you.”*

Use “Wise Feedback”

- High expectations
- Personal assurance
- Actionable next step

Wise Feedback & Socratic Method

Challenge: Insert the missing expression so that the program prints the maximum value from a list stored in the variable numList:

Student Progress: {0 < n}

```
max = numList[0]
foreach n in numList:
    if{MISSING CODE}:
        max = n
print max
```

Teacher Response

Diagnose

- What happens when you run this code?
- What would be printed if numList is [9,45,15,1]?

Leading Question

- As you loop through each number in numList, what comparison should you make to determine whether that number is the biggest one you've seen so far?

Actionable Step

- I think you might need to use the variable max as part of the condition for the if statement.
Try exploring different possibilities that use max. You can trace your code by talking through it or run it in Trinket with the list we just made to see if it works correctly.

Give Encouragement

- Encourage effort, grit and resilience
- Start with a compliment
- Find small victories
- Celebrate challenges as well as solutions
- Be measured and proportionate
- Use small rewards (e.g. raffle tickets)



Activity: Question Generator

Directions

Work in a group to come up with three leading questions that get more and more specific

Goal

Help the student find a next step



You have 4 minutes!

Activity Packet page 14



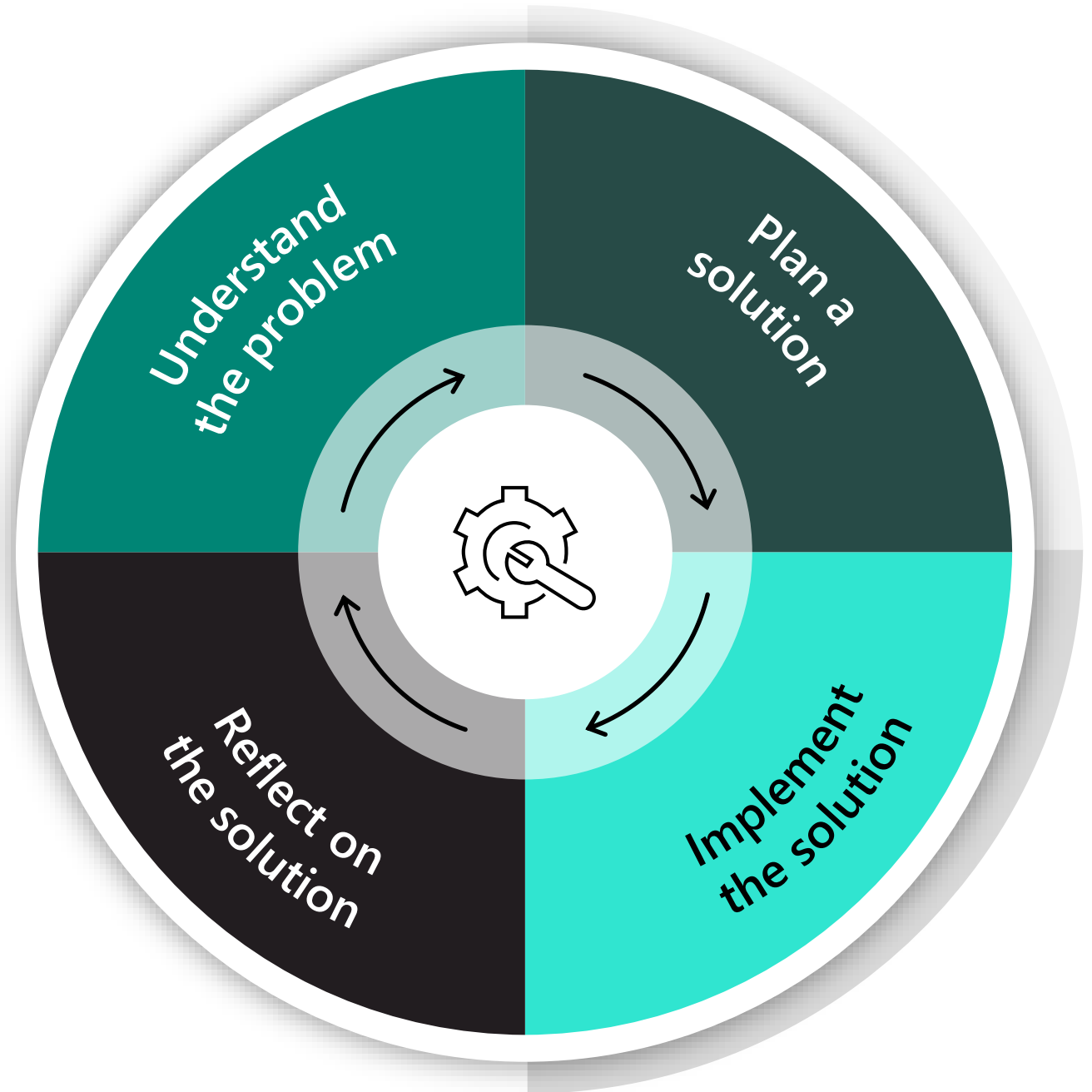
Socratic Tea Ceremony. Illustration: Jon Kudelka

Debrief: What was challenging about this activity?

Mock Lab



The Four Steps...



Where's the Problem?

- Instead of counting the total number of vowels in a word, the student is counting the number of 'e's.
- The student is printing the result of a computation instead of returning it.
- The student's loop is running twice as many times as expected.
- The output given is correct but incomplete.
- The program works correctly for positive inputs, but not for negative inputs.

For each error, hold up the number of fingers matching the step where the problem is occurring:

1. Understand
2. Plan
3. Implement
4. Reflect

Activity: The Misconception Game

Work in pairs. Assign a role to each person in your pair:

Student	Teacher/TA
Spend two minutes reading your code and the error	Do not read the student misunderstanding written on the packet!
Act out your misunderstanding with your TA until they can help you find the error	Read through the student's code and use Socratic questioning to diagnose the misunderstanding
Determine your next course of action based on your TA's help	Help the student get to a next step (not necessarily to a solution)

Activity packet page 15-16 for round 1, page 17-18 for round 2

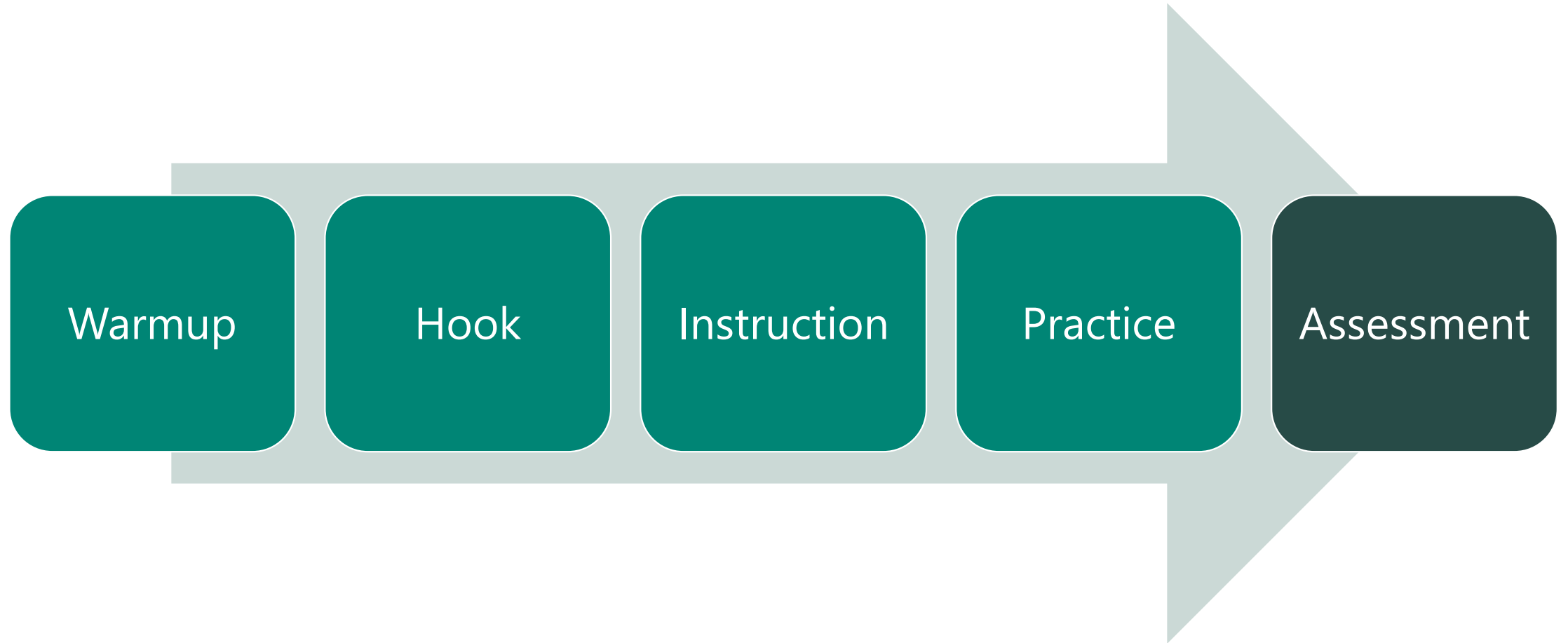
Discussion

What worked well?

From both the
"student" and "teacher"
perspective

What was difficult/
frustrating?

Assessment



Wrap Up



Homework

- **Online training: Canvas**
 - Materials from today
 - Online assignments
 - Optional resources to reinforce today's lessons
- **Schedule a team meeting and begin your Classroom Plan**
- **Schedule check-in meeting with your Regional Manager (as a team)**

Coming Up

Summer Check-ins

Week of <Date>

Training Session #3/School Year Kickoff

<Date/Time>

<Location>

Training Session #2

<Date/Time>

<Location>

First TEALS Monthly Meetup

<Date/Time>

<Location>

Exit Ticket!

<https://aka.ms/TEALS2019Summer1>

