Professional Development Situation: Virtual Training
Skill Focus: Developing a STEM Identity
Time Required: 120 minutes

YOU'RE A COMPUTATIONAL THINKER

Participants will learn how they are computational thinkers, how these practices fit into other activities, and how to develop computational thinking practices with youth.

Agenda

Welcome, introduction, and explanation of Zoom - 10 minutes
Introduce the skill with "Figuring it Out" – 15 minutes
Learn to play "Game with No Instructions" – 35 minutes
Discuss engaging youth in computational thinking – 15 minutes
Exploring how computational thinking can be used in other areas – 30 minutes
Conclusion – 15 minutes

Materials

- Computer with internet connection, camera and speakers
- A headset is not required, but may be helpful
- Paper and pens/pencils for taking notes may be helpful
- One die
- Crayons or markers
- One set of handouts
- Handouts (Send as attachments in pre-session email and download to your computer for screen sharing)"
    - Defining Computational Thinking (1 per participant)
    - Computational Thinking activity from Code.org (1 per participant)
    - Adding Computational Thinking to Other Activities (1 per participant)

Before the Session

- **Read this training guide** to familiarize yourself with the content and personalize the activities to best suit your style. Watch all videos and read informational materials.
    - *Italics indicate text that can be read aloud or emailed to participants.*
- Send a reminder email about the meeting. Determine if any participants require accommodations (sight; hearing; etc.). You can set up a meeting and send an email about the meeting through Zoom. The following should be included with the Zoom invite you send.
    - *The next professional development opportunity to enhance our STEM skills will be on DATE at TIME. Our focus for this session will be "You're a Computational Thinker". This is a virtual training, so before we begin the session, gather the materials and print one copy of each of the attached handouts. [we need to put links for 4 handouts in this set of materials so the facilitator can download the files and attach them in this email]*

      Have paper and a pen or pencil for taking notes.

      You will also need one die and a computer with an internet connection, a camera and speakers or headphones.

      Please try to avoid calling in on a phone. It will be more beneficial to use a computer with video and the chat box which is more difficult to use with a mobile phone.

      There are three attachments to print. You should not fill them out before the session—we will do that during the training.

      Finally, have Zoom (or another video conferencing tool) installed on your computer before the session. Test the tool before the meeting. If you are using Zoom, there are tutorials for using the video, audio, and screen sharing features. These can be found in the lower left corner of the screen after you have logged into Zoom.

      Let me know if you require any accommodations to participate in the training. I am happy to answer any questions you have and look forward to seeing you at the workshop. I can be reached at CONTACT INFO.
- Develop questions and responses participants might have during the training.

- Gather all materials and download handouts before the session to use as visual aids if necessary.
- Test the audio and video equipment.

Preparing for virtual trainings

- This training is written for Zoom, but you can use any virtual meeting platform you have access to and are familiar with. Practice using the virtual meeting software before the meeting and be sure you know how to use all the features that will be used in this training.
- Participants can download, install, and use the free version of Zoom. They will be able to use the whiteboard feature and breakout rooms as long as the host has the Pro version. As host, you need the Pro version to schedule a session that is longer than 40 minutes. You can contact Zoom to customize a plan for your circumstances.
    - Zoom plans and pricing https://zoom.us/pricing
    - Zoom for Education https://zoom.us/education
- If your organization has Zoom, they might have a URL that you should use to sign up for Zoom. If you are using a different platform, provide participants with the URL they need to download the platform. Set up Zoom (or whatever system you are using) before sending the email reminder because there is a calendar feature in Zoom so you can add your training session to participants' calendars.
- We recommend at least two facilitators for a virtual CSPD training. A co-facilitator will help you manage participants and answer questions in the chat box, but they will not have all the abilities in Zoom that a host has.
- Zoom allows up to 50 people to use video in a meeting with the Pro version. You can schedule two sessions if you have over 50 people. When participants join the meeting, their name should appear which is helpful for calling on someone.
- We recommend all facilitators have dual monitors so you can share one screen and have the participants and breakout rooms on the other screen. It will make it easier for you to manage everything.
- For this training, you will need to know how to share your screen, use the chat, breakout rooms, and whiteboard features. You may also want to record your meeting.
- Learn how to use the chat feature. Be sure you know how to save the chat. You must do this before you end the meeting.
- Learn how to use breakout rooms. There are tutorials on how to set up and manage them. You need to enable breakout rooms before you start the meeting. We recommend assigning participants automatically unless you have groups within your training that you

want to collaborate together. Be sure you know how to set up breakout rooms, assign participants, manage the rooms, and broadcast a message to all rooms.

- Learn how to use the whiteboard feature. Be sure you know how to insert text on, draw on, save and clear your whiteboard.
- Learn best practices for meetings and webinars that explain how to engage participants and follow up after the session.

Training Outline

Welcome, introductions, and explanation of Zoom (10 min)

- Greet participants as they join the meeting and ask them to introduce themselves in the chat by sharing their name and where they are from. Encourage everyone to turn on their video. You may want to record the training, especially if someone is absent.
- Introduce yourself and the topic for this training: "You're a Computational Thinker".
    - *Today, we may be using Zoom (or another platform you've selected) in ways you haven't experienced before. We are going to* [what you say will depend on the size of your session, options are listed below]
- *If you have less than 12 participants, have each person introduce themselves and share one expectation or hope for today's workshop as you say their name. This will assist people in knowing when it their turn to speak.*
- *If you have 12 or more participants, explain how breakout rooms work. Explain that everyone will introduce themselves and share one expectation or hope for today's workshop in their breakout room. Tell them you will bring them back from their breakout room in 8 minutes.*
- *After 6 minutes, use the "Broadcast a message to all" button and tell the participants they have 2 minutes left to finish introductions.*
- *After 7 minutes, click the "Close All Rooms" button. This will give each breakout room a 60-second countdown, and will automatically end the breakout session and return them to the main room when the timer ends.*
- Direct participants to https://support.zoom.us/hc/en-us/articles/115005769646-Participating-in-Breakout-Rooms if they need help with the breakout rooms.

Introduce the skill with "Figuring it Out" (15 min)

    - *"Unplugged" refers to computer science activities that do not require a computer. Computational thinking helps solve problems by devising solutions that a computer*

can execute. However, you don't have to use computers to learn about computational thinking practices. Today we will focus on using unplugged activities to teach about computational thinking.

- You will be using the unplugged Computational Thinking activity from Code.org.
- Introduce the activity "Figuring it Out" in the "Computational Thinking" lesson plan.
    - *Let's start out by exploring computational thinking with a challenge. The challenge is to sum up all the integers from 1 to 200 in your head.*
- Open the whiteboard tool and type/draw "Find the sum of numbers from 1 to 200".
- Pause for a second and then add the words "in your head".
    - *You will need to sum the numbers in your head. I'll give you one minute to work on the challenge, then you can share your solution.*
- Watch the videos of participants. Who seems overwhelmed? Who seems to be working on the problem in their head?
    - *Who is ready to share their solution?* (Pause to see if anyone wants to share.) *Who thought the problem was so hard that they didn't really try? It's a big problem to do in your head, isn't it? Let's see if we can break it up into smaller pieces that are easier to manage.*
    - *We'll start with the two ends. What is the sum of 200 plus 1?* (Pause and write the math sentence on the whiteboard.) *What about 199 plus 2?* (Pause and write the math sentence on the whiteboard.) *198 plus 3? Are you seeing a pattern?* (Pause and ask participants to describe the pattern in the chat box.) *How many pairs will add up to 201?* (Pause.) *What is the last pair in this pattern?* (Pause and ask participants to write the last pair in the chat box - 100+101=201.)
    - *That means we have 100 pairs of numbers that equal 201. Does that make it easier to figure out the sum of all the numbers?* (Pause) *If we added all these pairs together, we would be adding 201 + 201 + 201 + 201 for 100 times. Does anyone have a quick way to add the same number 100 times?* (Pause and ask participants to write the answer in the chat box. Then write the sentence on the whiteboard 201*100=20,100.)
    - *Now let's think about how we could adapt this solution to find the sum of all numbers between 1 and 2000. What would that look like? If we think about it, perhaps we could even come up with a solution that would work for any set of integers.* (If the group is interested, give them time to think about the algorithm, or just continue sharing the solution.)

- Write the formula for a general solution on the whiteboard. “The sum of all numbers between 1 and _____ is _____/2 * _____+1” (Pause so the group can discuss this solution if they want to.)
- Erase and close the whiteboard.
    - *Now, I have four new words to introduce. We will use these terms to reflect on how we solved this challenge together. The definitions are on your handout “Defining Computational Thinking”.* Open the handout on your computer and share your screen.
    - *The new terms are algorithm, decompose, abstraction and pattern matching. You can refer to this handout for the rest of our session together.*
- Allow participants a minute or two to read through the definitions.
    - *Let’s think about how computational thinking and the vocabulary terms at the bottom of the page are related to the math challenge we just solved.* (Participants need to understand that the vocabulary words refer to the practices or steps that learners use in computational thinking.)
    - *This was a really big challenge when we started, but we used **decomposition** to break it down into smaller pieces that were easier to manage. Then we noticed the pattern of pairs adding up to 201. **Pattern matching** made the challenge a little easier. In the end, we used **abstraction** to think about how we could describe a solution that would work for any set of integers, and we wrote that solution as an **algorithm**.*

Learn to Play “Game with No Instructions” (30 min)

- *Now we are going to play a “Game with No Instructions”. You will need your Computational Thinking handout from Code.org, dice, and crayons or markers. In your breakout room, you will figure out how to play the game using the handout and the computational thinking practices. Read the user experience scripts to get an idea of how others have played the game, then think about how you can use the computational thinking practices to figure out how the game works. I’ll give you 15 minutes to work in your breakout rooms, write your set of instructions, and play a couple rounds of the game. If you need help just click the “Ask for Help” button in the meeting controls and “invite host”. If you finish early, return to the main session. That will help me know that you are ready to move on.*
- Share your screen with the first page of the Computational Thinking student handout called “User Experience Scripts.”

- Send participants to their breakout rooms.
- For groups that click the "Ask for Help" button, tell them to circle the words that are identical for all three players and underline the words that change from player to player. Then have the participants look for the pattern.
- After 10 minutes, use the "Broadcast a message to all" button and tell the participants they have 5 minutes left. When you have one minute of time left, click the "Close All Rooms" button. This will give each breakout room a 60-second countdown, and if they haven't returned to the main room by the end of that time, Zoom will automatically end the breakout session and return them to the main room.
    - *How did your group figure out how to play the game? Use the chat box to tell us a little about what happened in your breakout room. Now we are going to wrap up this activity with a Flash Chat. This is a brief discussion that will help us reflect on and apply your experience today. We're going to go back to the breakout rooms, and share three questions, one at a time. I want you to discuss the questions. Pick one person in your group who will report back to the rest of us about your discussion. I'll let you know when time is running out, and we will gather back here.*
- Send participants to their breakout rooms. If you have less than 12 participants, you may choose to have them discuss these questions using the chat box in the main session.
- Click on "Broadcast a message to all" and share your first question.
    - *What strategies have you learned today that can help you when you have a problem that is too hard? What can you do?*
- In 3 minutes, share another question in the same way,
    - *After this experience, what will you do first when asked to do something which you don't know how to do?*
- In 2 minutes, share the last question,
    - *If you have a problem you can't solve that is just a little different from a problem that you have a solution for, what can you do?*
- In 2 minutes, click the "Close All Rooms" button and give the breakout rooms a 60-second countdown.
    - *As we come back together, I want the people designated by their small group to type in the chat box a little bit about your discussion.*

Discuss Engaging Youth in Computational Thinking (20 min)

- *What would you expect to happen if you led this activity with youth at your program? What would they get out of it? Would they think of themselves as*

computational thinkers? Share your thoughts in the chat box. (Pause so participants can type.)

- *When we do computer science in 4-H, it isn't just about the content it is also an opportunity to support positive youth development. Computer science activities can help youth develop a sense of confidence and competence. Computational thinking builds problem solving and critical thinking. So now we are going to watch a video of youth doing this activity and think about how Dagen facilitates the experience to support developing a STEM identity, - seeing themselves as a computational thinker.*

- Watch the activity overview video, Playing a Game without Rules.
  - *What did you notice in the video? Share your observations in the chat box.* (Pause so participants can type.) *Now let's watch a second video that focuses how Dagen facilitates the activity. As you watch the video, notice the positive and negative emotions youth experience. Are they learning to manage their emotions? Notice how Dagen adapts the activity to support youth in seeing themselves as a computational thinker.*
- Watch the skill video, Developing Computational Thinkers. Use the questions below to facilitate a discussion in the chat box about the videos.
  - *How does Dagen adapt the activity to support youth in seeing themselves as computational thinkers? Do you think he was successful?*
  - *What did he add that wasn't part of the activity when we did it?*
  - *What positive and negative emotions do you see in the video? Are they learning to manage emotions in this activity?*
  - *How would you adapt the activity to meet the needs of the young people you work with?*

  - *Positive youth development needs to be an important component of your thinking about computer science and computational thinking. We want this to be more than developing knowledge, we want to use these activities to help youth develop their full potential as thinkers, problem solvers and innovators.*

Exploring how computational thinking can be used in other areas (25 min)

- *The next handout we will be using is "Adding Computational Thinking to Other Activities." With your breakout group, you will brainstorm how computational thinking fits into other program areas. Thinking of computational thinking as a problem-solving method, come up with ways of incorporating computational thinking into each area on*

the handout. Then add three additional areas that you have in your program and do the same for these areas.

- Share your screen and show the handout "Adding Computational Thinking to Other Areas."
    - *Think about the activities you are already doing in your program. For example, if youth in your program play basketball, think about integrating computational thinking in that area. If you lead a horse club, think about integrating it in the activities your club does with horses.*
    - *Choose one person in your group to be the recorder and one to be the presenter. The recorder will open the handout on their computer to record the group's ideas on their handout. Towards the end of your time, choose your three best ideas. Your recorder will use share screen to share their notes and your presenter will share your top three ideas with the entire group. You have 15 minutes to work with your group. I will broadcast when you have 5 minutes remaining and we will come back together at* ____ [say what time it will be in 15 minutes]
- As groups are working, move between breakout rooms and answer questions. If groups are really struggling, you can share one or two of these examples.
    - Physical fitness – athletes use decomposition to figure out how they can increase their fitness, change their diet, or improve their performance.
    - Music – patterns in music help define jazz, country music or R&B.
    - Cooking – a recipe is an algorithm that is used to help food taste the same every time.
    - Team sports – the rules of a game can be thought of as an algorithm. In games like golf, players practice patterns of making the same movements over and over to get consistent results.
- After 10 minutes, use the "Broadcast a message to all" button and tell the participants they have 5 minutes left and it is time to select their three best ideas to share.
- After 15 minutes, click the "Close All Rooms" button and bring the group back together. Have the recorders get ready to share their screen when it's their group's turn to present so that the presenter can talk about their ideas.
    - *As we report back, your recorder will share their screen with your notes as your presenter shares your top three ideas.* (If you are short on time, have the presenter only share one idea.) *Keep the conversation moving quickly.*

Conclusion (15 min)

- *Now, let's reflect on today's training. Do you see yourself as a computational thinker now? Are there times you use strategies like breaking a problem down into smaller pieces, looking for patterns or similarities that can help you solve a problem, thinking about how you can adapt a proven solution to a new problem, or creating a list of steps to reach a solution?*
- *As we think about how to apply what we've learned about computational thinking in our work, please use the chat box to share your ideas so we can learn from each other. I will pause after each question to give you some time to think and type.*
- *Why do you think learning computational thinking is important for the youth in your program?* (Pause so participants can type.)
- *How will it help prepare them for their future?* (Pause so participants can type.)
- *How does integrating computational thinking into other program areas support this goal?* (Pause so participants can type.)
- *Now take some time to scroll through the chat, read each other's responses and comment on them. You can make your chat larger so that it is easier to read.* (Pause so participants can read and type.)

- Comment on ideas shared as appropriate and as time allows.
  - *Thank you for coming to the session today. I hope you learned what computational thinking is, how you can incorporate it into activities you already do, and how you are a computational thinker. Following the session, I will share the notes from the chat boxes and whiteboards we created today. You can also take the Computational Thinking activity and all the handouts with you to use in your programs.*
- Answer any final questions participants may have.
- Remember to save the chat before you close the meeting.

After the Session

- Compile ideas from the whiteboards and chat boxes into lists of "Computational Thinking Strategies" (from the flash chat), "Incorporating Computational Thinking into Program Areas", and other themes that emerged during the reflection and processing time.
- Within three weeks of the training, send an email to participants with the message below and the lists they generated in the workshop.
  - *Thank you for your participation in the recent Click2Science training "You're a Computational Thinker." I hope you found it useful. Consider meeting with a co-worker, supervisor, or friend to share what you learned.*

Here are some notes from our discussions that you may find helpful.

I look forward to continuing our learning at the next session on SKILL/FOCUS on DATE at TIME at LOCATION. Please let me know if you have any questions. I can be reached at CONTACT INFO.

Want to Earn Credit? Click2Science has teamed up with Better Kid Care to provide continuing education units. Check it out at: http://www.click2sciencepd.org/web-lessons/about

This material is based upon work supported by the National Science Foundation under Grant No. 1840947

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Defining Computational Thinking

The International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA) have collaborated with leaders from higher education, industry, and K–12 education to develop an operational definition of computational thinking. The operational definition provides a framework and vocabulary for computational thinking that will resonate with all K–12 educators. ISTE and CSTA gathered feedback by survey from nearly 700 computer science teachers, researchers, and practitioners who indicated overwhelming support for the operational definition.

Computational thinking (CT) is a problem-solving process that includes (but is not limited to) the following characteristics:

- Formulating problems in a way that enables us to use a computer and other tools to help solve them.
- Logically organizing and analyzing data
- Representing data through abstractions such as models and simulations
- Automating solutions through algorithmic thinking (a series of ordered steps)
- Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources
- Generalizing and transferring this problem solving process to a wide variety of problems

These skills are supported and enhanced by dispositions or attitudes that are essential dimensions of CT including:

- Confidence in dealing with complexity
- Persistence in working with difficult problems
- Tolerance for ambiguity
- The ability to deal with open ended problems
- The ability to communicate and work with others to achieve a common goal or solution

This definition for computational science listed above can be found at:
https://www.iste.org/explore/computational-thinking/computational-thinking-all

Decompose describes the thinking practice of breaking a problem down into smaller pieces.
Pattern Matching describes the thinking practice of finding similarities, or patterns, between things.
Abstraction describes the thinking practice of pulling out specific differences to make one solution work for multiple problems.
Algorithm is a list of steps that you can follow to finish a task or reach a solution.

Adding Computational Thinking to Other Activities

*Add three additional areas you have in your program at the bottom of the list and brainstorm ideas for these areas as well.

Area	Activity	How to add Computational Thinking
Healthy Living		
Civic Engagement		
Animal Science		
Cooking Projects		
Physical Fitness		