

Get in the Game!

Grade Levels: 3-8

Duration: 90 min

Nothing beats boredom like board games! Students will build and practice all four computational thinking (CT) skills while designing their own game.



Outline

Frame the Challenge	15 min total
Activate prior knowledge	5 min
Getting started	10 min
Activity	75 min total
Abstraction	5 min
Decomposition	5 min
Pattern recognition	10 min
Algorithms	20 min
Share out	20-25 min
Debrief	10 min

Grade Levels: 3-8

Duration: 90 min

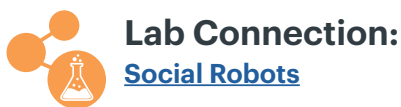
Concepts/Skills

Computational thinking: decomposition, pattern recognition, abstraction, algorithms

Objectives

Students will...

- Brainstorm the design for a board game.
- Use computational thinking elements to reflect thoughtfully on all elements of game design.



Background Information

Computational Thinking

Computational thinking is a problem-solving process that can be used in everyday life and applied to a wide range of tasks and problems. Although computational thinking may not be the first approach we think of when it comes to games, many of the techniques are actually applicable in the game design process and are used while playing games. This activity will guide students in using these techniques in a creative project while also featuring computer scientist Dr. Siobahn Day Grady and how she applies the very same techniques in her everyday work. For more information on computational thinking, check out our [Computational Thinking Tech Tip](#) and other [CT resources](#) on our Lessons and Activities webpage.



Ambassador Bio: Dr. Siobahn Day Grady

This project is supported by the Association of Science and Technology Centers (ASTC) and the If/Then Initiative, part of whose mission is to inspire girls to pursue interests in STEM careers by better representing women in STEM through media and learning experiences. Our featured guest for this activity is If/Then ambassador Dr. Siobahn Day Grady. Dr. Grady is a computer scientist and assistant professor in the School of Library and Information Science at North Carolina Central University. Some of her research interests include using machine learning to determine authorship of tweets and detecting false errors in autonomous, or self-driving, vehicles.

Listen to Dr. Grady's STEM journey and how role models helped her pursue her interest in computer science [here](#).

Materials and Preparation

Materials	
<ul style="list-style-type: none"> □ Get in the Game webpage, which includes: <ul style="list-style-type: none"> □ Links to the Get in the Game videos (6 total): <ol style="list-style-type: none"> 1. Getting started 2. Abstraction 3. Decomposition 4. Pattern recognition 5. Algorithms 6. Wrap-up □ Student handouts (one per student) □ <i>Optional:</i> Computational Thinking Element posters 	<ul style="list-style-type: none"> □ Way to show/share videos with students □ Scratch paper (1-2 sheets per student) □ Writing tools (1 per student) □ <i>Optional:</i> It may be helpful to have several board games and game instruction manuals available for students to get acquainted with pieces or board set-up. Limiting examples to simple classic games may help students focus on the key elements of the activity.
Prep	
<ol style="list-style-type: none"> 1. Watch the Get in the Game videos and make sure you have a way to show/share them with students. 2. Hang up/share the Computational Thinking Element posters as a reference for students during the activity. 3. There are a number of different ways that you can connect the board game to content students have been learning in class. Determine at this time if you will focus them on a specific topic or skill, or allow them to choose the focus of the game themselves. 	

Content Connections

Board games can be developed around any subject area or content that students are learning in class. If connecting the game directly to the content, introduce the goal to students at this time. Depending on the concept and age, focus student work by providing a topic or even an overall structure.

For example: In social science, students could create a community game about the Roman Empire; in ELA they could create a “choose your own adventure” game; a science game could focus on life cycle, habitats or resemble the popular board game Wingspan.




Frame the Challenge

The goal of this activity is to guide students through the thought process behind developing a board game. Although the extension includes opportunities for developing the actual game, this specific lesson does not get into the broader theory of game design. Instead, students will apply the process of computational thinking to the fun challenge of designing a board game. Therefore, keep the pace quick throughout, and encourage students to develop ideas further at a later time if they are excited to actually build their games.

Activate Prior Knowledge (5 min)

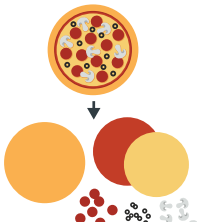
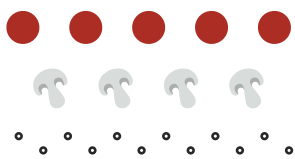
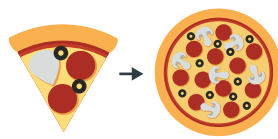

1. First, build student interest and excitement by tapping into what they already know about board games.
2. Have students share what they know about board games. Take notes on their ideas somewhere they can refer back to later.
 - *What are some board games you have played before?*
 - Examples: Candyland, Monopoly, Pandemic, Risk, Sorry, Clue
 - *What are the different parts of a board game?*
 - Examples: A board, rules, dice, cards, how to win, game pieces

Getting Started (10 min)




1. Have students watch the [Getting Started video](#) (2:52 min) .
2. To build excitement and connect the activity to the real world, use the following scenario:
 - *Today, we are all board game developers. Our job is to brainstorm a new board game to learn about **[insert skill or topic]**. We don't need an actual prototype—or physical model—of a game yet, just the instructions on how to play. To help us work quickly, we're going to use computational thinking to create a game instruction manual.*
 - Fill in the skill/topic yourself if you have chosen a content connection. Alternatively, let students know they should choose a purpose/topic of the game themselves based on something they want to teach others about.
3. Reinforce students' understanding of Computational Thinking (CT).
 - If you have the Computational Thinking Elements posters, refer to them at this time.
 - Explain to students that they use these skills all the time, but today they're going to use them for something they might not do every day — game design.

Computational thinking is a problem-solving process that involves skills that you use every day: abstraction, decomposition, pattern recognition and algorithms.

If students need an example for each CT skill, refer to our [Computational Thinking Tech Tip](#) or use an example like this:

<p>Decomposition</p>  <p>What are the different parts that create this?</p>	<p>Pattern Recognition</p>  <p>Do you notice anything that repeats?</p>
<p>Abstraction</p>  <p>How do you know what this is?</p>	<p>Algorithm</p>  <p>What steps do you need to follow to create this?</p>

4. The activity is outlined as a design problem, with criteria and constraints below. Explain it to students and address any questions they might have.

Design Problem	Create an instruction manual for a board game using computational thinking.	
Criteria	Use all four elements: abstraction, decomposition, pattern recognition and algorithms.	
Constraints	Time limit: 40 minutes	

5. If possible, place students in teams of 2-4 to collaborate or share their work.
6. Provide students with the Student Handout and address any questions they have. Additional scratch paper may be useful as well.
7. Let them know that at the end of the activity that they are going to pitch their board game ideas to each other.



Distance Learning Adaptation

In a virtual setting, share the Getting Started video and Student Handout with students asynchronously.

- If you are able to schedule synchronous sessions, do the first parts of the activity together as a class and have students finish the remainder on their own, before sharing virtually or in a synchronous video call.
- For more tips on adapting Design Challenges to a virtual setting see our [Educator Tips for Remote STEM Learning](#).

Activity

This activity will rely on the rapid design process. Encourage students to move along even if they have not finished their ideas in each section. The goal is that they experience the elements of computational thinking during game board design.



Abstraction (5 min)

1. Although the CT skills can be used in any order, for the purpose of this activity, students will start with abstraction.

- *Abstraction involves looking at the bigger picture and what's important. This means that you're not thinking about the details at this point.*

2. Have students watch the [Abstraction video](#) (1:31 min). 

3. Once students have been introduced to the concept, have them make notes on their handouts.

4. At this point, students can also be invited to give their game a theme or topic if they haven't already.

- A theme is not a necessary component, though it may happen naturally. Some games like Sorry and Trouble have no theme and focus on the mechanics, whereas others like Candyland and Monopoly have a narrative or theme.
- If building a game connected to content, the theme will most likely come from the topic.
Example: A game about the solar system might have a space race theme.
- For an interesting twist, students could consider doing a mash-up for their theme and add in a random topic.
Example: A game about the solar system could have a candy theme, while a game about the voting system has an apocalypse theme.

Guiding Questions: If students are having a hard time brainstorming or completing the handout, try to activate some of their prior knowledge using the following questions. Encourage them to think about games they have played before as well as the one they are designing:

- *What's the purpose or goal of your board game?*
- *What do you want the players to do by the end of the game?*
- *Is this game collaborative, or are players serving their own interests?*
- *Is there a single winner? Are there teams? Does everyone have to cooperate to win the game?*



Decomposition (5 min)

1. Next, have students move on to decomposition.

- *Decomposition involves breaking a problem — in this case, the game — into its smaller parts.*

2. Have students watch the [Decomposition video](#) (1:45 min). 

3. This part of the activity is similar to creating a to-do list. Students will list all the parts of a game, but they do not need to plan for or design ideas for each part. Instead they are thinking about all the aspects that might need to be developed later.

Guiding Questions: Use these questions to help students think about the games they have played before as well as the one they are designing.

- *How many players will you need? What materials and playing pieces?*
- *What information is usually included on a game box or instructions? How is it organized?*
- *How can you break your game down?*
- *What are all the different parts that make up a game?*
 - Examples: Rules, player information, game pieces, how points are won/lost, etc.



Pattern recognition (10 min)

1. Next, have students look at pattern recognition.

- *In pattern recognition, we're also looking for similarities. In computer programming, pattern recognition helps us recycle solutions from previous programs and apply them to new problems. We can use this with games as well. Think about your favorite games. Are there things you can borrow from them? What mechanics would be useful here?*
- *Example: Every time players land on a "ladder," they might go down on the board. Every time they roll a 6, they have to go back to start.*

2. Have students watch the [Pattern Recognition video](#) (2:22 min). 

3. Students might find it useful to sketch, create tables or charts to explain their ideas in this section. They can even use a scratch piece of paper to sketch initial ideas for the game board or playing pieces.

Guiding Questions: Use these questions to help students think about the games they have played before as well as the one they are designing.

- *What patterns and similarities have you noticed when playing other games?*
- *How will players move? What patterns do you notice in the way you use dice rolls vs. spinners, card decks, etc.?*
- *Are there rules that you want your board game to repeat?*
 - *Example: In Monopoly every time you pass 'Go' you collect \$200.*
- *How will you lay out the board? Are there color tiles or grids that have a specific purpose or pattern?*



Algorithms (20 min)

1. In their final step of game design, students will develop a simple algorithm for their game.

- *Algorithms are step-by-step instructions to solve a problem. In this case, our algorithm is going to describe how the game is played, the rules.*

2. Have students watch the [Algorithms video](#) (1:51 min). 

3. This section can be the most time-consuming. To focus students, have them try to develop at least **five** rules. The algorithm can be very simple.

Guiding Questions: Use these questions to help students think about the games they have played before as well as the one they are designing.

- *What rules can you borrow from other games you have played?*
- *What do players do first?*
- *What comes next?*
- *How do they progress through the game?*



Tip

Provide students with some sample rules from real games before they do this section.

For example in Snakes and Ladders:

1. Roll the dice to decide who goes first. Highest score starts.
2. Take turns rolling the dice. Move your counter forward the number of spaces on the dice.
3. If you land on the bottom of a ladder, move up to the top of the ladder.
4. If you land on the head of a snake, slide down to the bottom of the snake.
5. The first player to get to the finish is the winner.



Share Out (20-25 min)

1. Have students watch the [Wrap-up video](#) (3:06 min).
2. Facilitate a way for students to share their board game idea and instructions.
 - Depending on your resources and setting, this can vary from an informal gallery walk to a formal presentation.
 - Keep the sharing brief and encourage students to focus on how they used the CT skills.
3. Encourage students to give each other feedback on their games.
 - They can also notice places where they saw strong use of one of the CT skills. For example:
 - *Wow! You really broke down the parts of the game into small sections. I liked your decomposition.*
 - *I noticed you also used a pattern in your algorithm. All of your rules began with “The players will...”*

Guiding Questions: To focus their sharing on one of the CT skills, use these guiding questions. Students can choose one or two to respond to:

- Abstraction
 - *What type of game is it?*
 - *What are the objectives?*
 - *How did you decide on what kind of game to create?*
- Decomposition
 - *What are the different parts that make up your game?*
 - *What does general game play look like?*
 - *What are the components of your game?*
- Pattern recognition
 - *What did you borrow from other games?*
 - *Are there any events that repeat in the game?*
 - *Are there rules in the game that favor or disfavor patterns?*
- Algorithm: *Show your algorithm/flowchart.*



Distance Learning Adaptation

- Students can capture the work they did and share it asynchronously through video or photo using Flipgrid, Google Draw, Padlet or other tools.
- Use video calls to have students take turns “presenting” and sharing their ideas with each other.
- Students can also seek feedback from family members and share notes with their input.
- For a more authentic audience, invite a community member, computer programmer, or game designer to speak with students virtually and hear their presentations.
- In addition, if you or your students share any of their ideas online, we encourage you to use the hashtag #TheTechatHome.



Debrief (10 min)

1. After students share their games, bring the conversation back to computational thinking elements. Lead a short debrief with some of these questions:
 - *What did you notice about the computational thinking elements as you used them?*
 - *Which elements felt the most natural? Example: I use this all the time!*
 - *What were the most challenging elements to use?*
 - *Can you think of other times in your life where you think you use computational thinking? In school? At home?*



Extensions

- If you have more time, continue on to the next steps of game design! Have the students create physical versions of their game ideas. This development will have to take place over the course of several days. Build in opportunities for students to test their ideas with other teams, encouraging iteration. By creating the full game, they will have more opportunities to practice the CT elements. This is especially true for algorithms, as they will have to refine their logic to write more concise rules. This can be likened to “debugging.”
- You could also focus this activity around a topic from class. For instance, students could make a board game that relates to a concept that they have learned about in science or U.S. history. In this case, they will also use abstraction to glean the parts of the topic matter that are relevant for their game.



Lab Connection






Visiting the Tech on a Field Trip? Sign up for one of our Science or Innovation Labs. This activity works well before or after our Social Robots Lab. Learn more at thetech.org/innovationlabs.

Standards Connections

CA Computer Science Standards

Grades	Standard	Description
3-5	AP.13	Decompose problems into smaller, manageable tasks which may themselves be decomposed. (P3.2)
3-5	AP.14	Create programs by incorporating smaller portions of existing programs, to develop something new or add more advanced features. (P4.2, P5.3) <i>Note: In this case the “existing programs” are previous board games</i>
6-8	AP.13	Decompose problems and subproblems into parts to facilitate the design, implementation and review of programs. (P3.2)
6-8	AP.14	Create procedures with parameters to organize code and make it easier to reuse. (P4.1, P4.3) <i>Note: This is a general application rather than specific to code.</i>
K-12	P3	Recognizing and Defining Computational Problems
K-12	P4	Developing and Using Abstractions

Vocabulary

	Computational Thinking	A problem-solving process that can be broadly applied across content areas and everyday life.
	Decomposition	Breaking down problems into smaller problems.
	Pattern Recognition	Recognizing if there is a pattern and determining the sequence.
	Algorithms	Step-by-step instructions to solve a problem.
	Abstraction	Generalization of a problem – focus on the big picture and what’s important.