We Can Make Games Too!

Unit Overview (Remote Optional)

This unit can be viewed as a substitution for the “Algorithms are Everywhere!” and “We Can Make Apps Too!” units, that instead requires computers and can be done remotely.

Students will individually build a simple game using MIT’s Scratch platform while learning algorithms, conditionals, and loops. They will then be introduced to the impact of computing, adapt their game to address a real-world problem, and finally present their game to the rest of the class.

**Lesson 1: Intro to Scratch** - Students will become familiarized with the Scratch environment. (AP-07)

**Lesson 2: Objects** - Students will be able to describe what objects are in Computer Science and create their own objects in Scratch. (AP-06)

**Lesson 3: Algorithms** - Students will be able to create and understand simple algorithms. Students will also explore applying this to the Fish Chomp project. (AP-01, AP-06, AP-10)

**Lessons 4: Conditionals** - Students will be able to create if/then statements for situations from their own life and also use conditionals in Scratch. (AP-05)

**Lesson 5: Loops** - Students will be able to create simple loops and apply this to the Fish Chomp project. (AP-04)

**Lesson 6: Object Interaction** - Students will learn how objects communicate between each other in Scratch and apply this to complete the base Fish Chomp project. (AP-03, AP-05, AP-06)

*Note: Lesson 6 will likely take 1-1.5 hours to complete.*
Lesson 7: Variables - Students will be able to define what a variable is and add variables to the Fish Chomp project. (AP-02)

Lesson 8: Technology in Community and Culture - Students will understand how technology has shaped/changed our lives and solved many problems we face, and think of additional problems either in their personal lives or in the world as a whole that technology could help solve. (IC-01)

Lesson 9: Accessibility - Students will consider and learn about the ways to make apps more accessible, including the Fish Chomp project. (IC-02)

Lesson 10: Adapting Game to Real World - Students will draw on the previous lessons to adapt their game to address a real-world problem. (AP-08)

Lesson 11: Presentation and Feedback - Students will present their final games and give positive feedback to their peers. (IC-03)
Lesson Overview

Introduction
5 minutes

Exploring Fish Chomp
10-20 minutes

Reflection
5 minutes

Materials
1 Laptops
Welcome to the world of Scratch! Throughout this unit we will be using the MIT Scratch tool to introduce students to basic programming skill and theory. While Scratch has a generally easy-to-use interface, we recommend that you grow familiar with it so you can assist students should they run into technical issues.

Scratch is linked [HERE](#).

Play with the different programs, modify them! You’ll be pleasantly surprised at how fun block programming can be!

The goal of this lesson is to allow your class to, as you hopefully did, become familiar with the Scratch environment. While this largely means playing with the game on Scratch, students should also begin to think about its inner workings: How does this game work? How can I change it?

Inform your students about the roadmap of the next few weeks. They will be engaging with the Scratch environment, and learning to recreate a game they are about to play. This will involve adding characters, score systems, and movement.
Exploring Fish Chomp

Pass out laptops to students (if there are not enough computers for everyone to have their own, then have even groups). Lead them to the fish chomp game.

The Fish Chomp game is linked [HERE](#).

Allow students to largely have control over what they do and how they play the Fish Chomp game at first. Hover around groups to see if any might be having trouble with Scratch, but otherwise let them explore the game and what it does.

After a while, ask your students to view the code of the game by pressing **See Inside**.

This allows students to view the code for the project. Encourage students to speculate on what different blocks of code contribute to the work--remember, every block matters!
Once you feel students have explored the code enough, conclude the activity and remind students that they will be working to make the same thing.

**Reflection**

After the activity has ended, let students share their likes and dislikes about the game. Some questions you can use to facilitate conversation include:

- What did you enjoy about Fish Chomp?
- What would you have changed to make the game better?
- Did you figure out what some of the blocks (instructions) did?
Lesson Overview

- Introduction: 5 minutes
- Student Exploration: 15 minutes
- What are Objects?: 10 minutes
- Reflection: 5 minutes

Standards
- AP-06: Break down problems

Materials
- 1 Laptops
Introduction

In the last lesson, students familiarized themselves with the Scratch environment. In this and future lessons, students will work on making the Fish Chomp game themselves! While this might seem like a lot, remind students that the process will be broken down into concepts. This lesson, they will start with objects!

Have students get onto the computers, ideally one student per computer, but make groups of students if needed. Note that these groups should stay the same for the rest of these lessons.

Students will need to create an account on Scratch so that they can save their work for use in subsequent classes. Have them create the account and use a password that they will definitely remember. Have each student create a new project. If students are in groups, it may be a good idea for you to log in on all computers with your account and to have each group have their project on your account. Otherwise, if the member of a group whose account has the project misses a lesson, their group will not be able to make any progress.
Student Exploration

We will begin building the game with “objects.” Inform students that the fish from the Fish Chomp game—all of them—are in fact, objects!

Scratch gives students a default object to start, but it is incomplete! Allow students to try and make this default object into the big fish. Afterwards, they should also try to make the little fish! Allow students to explore Scratch and try to figure out how to perform these desired actions on their own. They may struggle some, but you can have students help each other and then also help them along.

You should point out that the fish do not need to look exactly like they do in the game. They can mix and change the objects’ appearances to their liking!

After ~10-12 minutes, have a student volunteer to explain how to add objects into Scratch.
What are Objects?

Now that students have an example of an object in mind, we must explain what exactly objects are in Computer Science! To segue into this, ask students what examples of objects they can find in the classroom, based on their normal understanding of the word “object.”

After a student gives an example of an object, ask students to describe that object. What are its properties? What is its color, shape, size?

Explain that Computer Science objects are very similar to those in real life. Specifically, both can be described by their properties! Look at the big fish object as an example. Have students describe some of the fish’s properties. Students should notice the similarity.

Explain to students that having different objects is quite important! This is because each object will have specific code to it; each object has different actions, so it needs different code!

Reflection

- Are animals objects in the real world? Can they be objects in Computer Science? What about humans? (The idea here is to have students think about how even things we wouldn’t consider objects in the real world can be objects in programming)
Lesson Overview

Introduction
5 minutes

Shoe Tying
10-20 minutes

Moving our Fish
10-20 minutes

Reflection
5 minutes

Standards

- AP-01 Create multiple algorithms
- AP-06 Break down problems
- AP-10 Identify and debug errors

Materials

- 1 Laptops
- 2 Shoe w/ laces
Introduction

The goal of this lesson is to introduce students to the idea of an algorithm. While it sounds like a very abstract/complicated term, in fact, we all experience algorithms in our day to day lives. The teacher can ask students to guess before explaining that an algorithm is a sequence of steps done to complete a task, solve a problem, etc.

Ask students if they can think about some algorithms in their life. If students struggle, offer up some examples. I.e. brushing your teeth, making a PB & J, etc.

Algorithms can be very simple like the ones we described above, but they can become quite complicated. Let’s explore how a computer might respond to a more complicated task.
Shoe Tying

Students will collaborate to vocally instruct the teacher how to tie their shoelaces.

All the teacher will need for this activity is a shoe with laces. Explain to the students that they must provide **very specific** step-by-step instructions to help the teacher tie their shoe completely.

Ask for students to volunteer to give the first step in the instructions. You, the teacher, should begin tying the shoe, but follow the instructions given by the students **exactly**. Follow the students’ “algorithm” (instructions) until you either tie the shoe or make a mistake. If the algorithm is incorrect (it likely will be) ask the class to fix it (this is the equivalent of debugging in computer science) by starting the process from the beginning and giving more specific instructions. The teacher should emphasize that specificity and order matters by leading students to explain the process of tying a shoe in words so exact that they could not be misinterpreted.

Now that students have experienced this, we can try and put algorithms to work in Scratch!
Moving our Fish

Bring out laptops again so that students may return to their Fish Chomp Project.

Today, students will focus on the “move” and “turn” functions, as these bear similarity to the Shoe Tying activity.

You can have students copy the blocks shown in this image to start. Generally, the Control blocks of repeat and forever are used for motion to be continuous, though these will be discussed in loops. For now, teachers can explain that this is just to repeat the movement so we can see it, and that it will be discussed in depth later.

Allow students to explore the different paths which they can give the fish using the motion blocks. The teacher can also offer progressively difficult paths for students to try:

- Can students make the fish move in a straight line?
- An “L”?
- A square?
- ...And onward
Moving our Fish (cont.)

Students should be allowed to test and fail as they progress, but if students ask about the actions of different Motion blocks, the teacher should explain.

After students have interacted with the blocks for a bit, conclude the activity.

Reflection

Students should reflect on the nature of an algorithm and its applications to today’s work on the Fish Chomp. Questions might include:

• What is an algorithm?
• How might group work help solve these problems quicker?
• What kind of block would I use in which case?
# Conditionals

**We Can Make Games Too: Lesson 4**

## Lesson Overview

<table>
<thead>
<tr>
<th>Activity</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>5 minutes</td>
</tr>
<tr>
<td>What are Conditionals?</td>
<td>10 minutes</td>
</tr>
<tr>
<td>If/Then in Scratch</td>
<td>10 minutes</td>
</tr>
<tr>
<td>Reflection</td>
<td>5 minutes</td>
</tr>
</tbody>
</table>

## Standards

| AP-05                  | Implement conditionals |

## Materials

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Laptops</td>
</tr>
</tbody>
</table>
Introduction

Having learned objects and movement, students are ready to explore conditionals. Discuss that students must have run into issues in the previous lesson (for example, movement was limited and always in order). But what if we want our big fish to move to the right only when there is a small fish to the right? Or to the left? How can we get these objects to respond to different situations? The answer is conditionals.

What are Conditionals?

Begin this activity by discussing the idea of if/then statements. Recite some situations and call on students to offer responses to those specific situations. While there are many examples you can use, some are provided here for you:

If it’s raining...
then I’ll use an umbrella
If I’m hungry...
then I’ll get some food to eat

Explain that all of these statements are called conditionals. We’ve described a condition and what we do depends on it. The condition of raining, for example, means we take an umbrella. But if it’s not raining, we do not take an umbrella. A condition changes the actions we take!

You can repeat this explanation with your own examples as well.

Next, ask students how we might use a conditional with the big fish in the Fish Chomp project. Think: what does the big fish do, and when does it do it? Try and write down an if/then statement for this on paper.

(If our mouse is far away from the big fish, then the fish should turn and move towards our mouse)
If/Then in Scratch

Now have their students try to turn this if/then statement into a conditional in the code for the big fish. They should explore on their own to find the different blocks they need to fulfill their desired functionality. As usual, help students when they get stuck.

Reflection

• What are other conditionals you think of every day? (For example, how do you decide what clothes to wear for the day?)
Loops
We Can Make Games Too: Lesson 5

Lesson Overview

- **Introduction**: 5 minutes
- **Learning Loops**: 10 minutes
- **Loops in Scratch**: 10 minutes
- **Reflection**: 5 minutes

**Standards**

- AP-04 Implement loops

**Materials**

- 1 Laptops
Introduction

The goal of this lesson is to introduce students to the idea of loops. Explain to students that they have already used loops in previous lessons (such as through moving their objects in Fish Chomp). Describe that if you want something to repeat in your work, you need to use a loop!

Learning Loops

Students will discuss with the teacher the real-world examples of loops. Start by reiterating the definition of a loop, then asking students if they can come up with things they might use a loop for in their day-to-day lives.

Here are some examples:

Consider a sport like basketball or baseball. Each quarter or inning might be considered a part of a loop!

How about the school bells? Each day they loop through different periods, like the start of the day, breakfast, lunch, and the end of the day!

What if a chef was told to make 10 sandwiches? They would repeat the pattern they used to make that sandwich 10 times!

As we can see, there are many examples of loops that we can find. Note that some of these loops can go on forever (like maybe our daily morning rituals), while others stop, or terminate, after a specific amount of times (such as in a sports game). We should consider how we can use these in our Fish Chomp Project.
Loops in Scratch

Give students time with the laptops to work on loops in their Scratch projects. In previous lessons, loops were already provided (point them out—the forever and repeat loops). Allow students to try motion without the loops. See? We need loops to continue and repeat actions in our programs.

Allow students to work on the project after this demonstration. As usual, offer help if needed.

Reflection

• Try and think of another loop you experience in your life? Is it a loop that is repeated, or does it end? (An example might be your daily routine).
Object Interaction
We Can Make Games Too: Lesson 6

Lesson Overview

Introduction
5 minutes

Student Exploration
20 minutes

What are Objects?
10 minutes

Reflection
5 minutes

Standards
AP-03 Implement sequences
AP-05 Implement conditionals
AP-06 Break down problems

Materials
1 Laptops
Introduction

This is going to be a longer lesson plan. It will likely take students between 1-1.5 hours to complete all of the tasks. Please split up this lesson plan as needed. At the end of this lesson, students will have written all of the code that is in the fish game.

Let students know that today they’ll be writing a significant portion of their games. They have many tools for creating programs, but now they must learn about broadcasting.

Student Exploration

Have students try and think of what they have left to do and plan out how they will achieve the remaining tasks. If they need more direction, here are the tasks left to complete the game:

- Starting with just 1 little fish, make it move randomly forever
- If the little fish and big fish touch, we broadcast to the big fish to eat the little fish (meaning we will also make the second block of code for the big fish)
- All little fish should have the same code

For the most part, students will need to explore the different Scratch commands to find the ones that can help them. After about 20 minutes, all students should hopefully be working on writing the code for the interaction between the big and little fish.
Object Interaction

Have students reflect back to objects. Objects are like the big and little fish in our project, each with distinct properties and code/actions. However, what if they need to communicate? How does one fish know to disappear after it has been “eaten”?

You can have the students try to answer this question. They may talk about how in real life, the big fish would see the little fish and it would be obvious to it that it has caught the little fish and can now eat it. You want to explain that in real life, the fish has a brain. However, our fish object does not have a brain. It only knows what we tell it! This is a very important part of programming.

Offer an example to help students in their thinking. If we have an astronaut that wants to talk to the space center on Earth, how might they achieve this? Since they are far too far, what else can we use?

Students should have answers along the lines of “the astronaut and space center have a connection so that they can talk to each other, kind of like walkie talkies.”

Given this answer, explain the if, broadcast, and receive sections of this phrase: If the astronaut wants to speak to the space center, they broadcast their message to the center, and the center receives their message. Now, make students think about which code blocks they might use for this task.

Have students resume working on their games, giving them help as needed and sufficient time to complete the project. This may be another 30-45 minutes. As usual, offer help to stuck students.

After all their work, be sure to congratulate your students on a job well done! It was certainly a lot of work!
Reflection

- Do you feel like you learned computer science while building this game? Why or why not?
Lesson Overview

- **Introduction**
  - 5 minutes

- **What are Variables?**
  - 10 minutes

- **Variables in Scratch**
  - 10 minutes

- **Reflection**
  - 5 minutes

**Standards**

- AP-02 Store and modify data

**Materials**

- 1 Laptops
Introduction

The goal of this lesson is to introduce students to the idea of variables. To begin, ask students what features most games typically have (video games, sports, board games, etc). If they can’t get it, explain that most competitive games have SCORES! After all, the point of many games is competition—how high of a score can you get? Can you beat your old score? Can you beat your friends’ scores?

We’ve finished our Fish Chomp project, but we can definitely improve it! Why don’t we consider adding a score system? But to do that, first we must explore the concept of variables.
What are Variables?

Let’s think about variables. The term variable comes from the word “vary,” which is when something changes. Variables are pieces of information that can change.

We already described that the score of a game can be a variable. This is because as you do things, like get a point, the score changes. However, it can go beyond just numbers. Can you think of more examples of variables we see in our lives?

Here are some examples:

How about the day of the week (a word variable)? That changes every day right! We can see it on our phones and calendars, each day a new name!

The temperature is another example (a number variable). What’s the temperature right now? Is it the same as it was yesterday? How about an hour ago?

We can use variables to add a score system to our Fish Chomp game to make it more interesting!
Variables in Scratch

Allow students to try and apply their knowledge of variables to the Scratch project. If they are confused, try and hint them towards the variables tab, so that they can create an object that tracks the score. If they are struggling to think of what the “score” should be, ask them what is the objective of the game? They should hopefully arrive at the idea that the score should be how many little fish the big fish has eaten. They should also name their variables appropriately!

As always, provide help when needed.

Reflection

- Can you give examples of three types of variables based on your own life?
Lesson Overview

Introduction
5 minutes

Identify Problems and Solutions
20 minutes

Reflection
5 minutes

Standards
IC-01 Consider technology's influences

Materials
1 Whiteboard
2 Marker
Before you Begin

The goal of this lesson is to introduce students to the ways computer science has transformed our communities and cultures. We will teach the concept first by using real world examples to demonstrate how technology has changed society in ways that have solved public problems. Next, students will think about how code that already exists could be used or altered to solve problems they know about.

Introduction

Start by asking the class to think of ways computer science has changed the world. If they have trouble, ask them to think about how they would find information if Google did not exist. Use examples like this to prompt the class' thinking about how the world would be different without technology.
Identify Potential Problems/Solutions

Start by having the students share a couple of ideas for problems they see, whether it is within the school, in their community, or just in the world in general. If students are struggling, ask questions like “What about everyone who rides the bus? Is it ever frustrating how the bus gets to your home at a different time every morning and you never know exactly when it'll get there? What if there was an app that tracked where the bus was, would you like that? What would it look like?” Other example problems could be:
- Not knowing what the school lunch is going to be that day
- Struggling to remember all of your homework due dates
- Unhealthy eating
- School bullying

Write these issues on a whiteboard for the class to see.

Next, ask students to identify which problems could be best solved by using Computer Science. Of the problems identified, allow students to choose an issue they would like to think about a solution for.

Finally, group students based on alike interests, with ideally 3-4 students in each group. Some students may have to agree to work on their second choice in order for there to be balance. For the next 10-15 minutes, allow students to discuss with each other different ideas of how technology could help solve or at least raise awareness of their given issue. This is meant to be a thought exercise, not actually designing a product that they will build.

Have students share the ideas they come up with the rest of the class.
Reflection

- Are there some problems technology is not an appropriate solution for?
Lesson Overview

- **Introduction**: 5 minutes
- **Researching!**: 20 minutes
- **Reflection**: 5 minutes

**Standards**
- IC-02: Explore accessibility of technology

**Materials**
1. Papers
2. Pens or Pencils
Introduction

The goal of this lesson is to have students think about how accessible games can be. The reality is that as games become used by more and more people, developers come across issues that they did not consider before. This is usually true because they have never experienced these certain issues in their own lives.

When it comes to computer science, accessibility is about creating technologies that are suitable for everyone, including people with disabilities. Disabilities usually include physical, auditory, visual, and many others. It is important to have students understand how disabilities are diverse and impact people in different ways.

Researching!

Start a class discussion about common disabilities and how they affect people. The students should slowly take over the discussion and think about other disabilities.

After, students should come up with example apps or games that might not be accessible to everyone. Students should write down potential solutions to this issue, and then repeat the process with the Fish Chomp project. They are encouraged to really think of possible accessibility issues with their apps and how they can make the apps more accessible. Think visuals, audio, and controls for the game!

Have students share their ideas amongst each other.
Reflection

• Why do you think so many games and apps are not accessible to some people?
• How can we make sure the things we design are accessible to everyone?
Lesson Overview

Introduction
5 minutes

Brainstorming
10 minutes

Edit Games
15-35 minutes

Reflection
5 minutes

Standards
AP-08 Iteratively develop programs

Materials
1 Laptops
Introduction

At this point, students are quickly approaching the end of this Scratch unit! We have explored building Scratch games, the helpfulness and applications of computer science, and we learned how to make our products accessible!

Explain to students that they will have to now design their own game in order to solve a real world problem. They should use the many skills they’ve learned in the past lectures.

Brainstorming

Teachers can either have a quick brainstorming session as a class, or with students split into groups, or students individually. Encourage students to think about how they can change the different images in their game in order to achieve some sort of positive message. Maybe something that encourages good habits? Something that addresses a problem we discussed? Maybe they want to add in different objects, maybe “bad” objects that they want their main character (big fish) to avoid. Teachers should allow students to use their creativity. If some students are struggling for ideas, you can propose some of these:

- Eating healthy
- anti-bullying
- cleaning up your room

For any of these ideas, students should be able to think of what they should change the fish images and background image to and how those changes create a message for their game.
Edit Games

You should give students a good amount of time to complete their edits, taking up another class if needed. As students are editing their game, encourage them to think of how accessible their game is. How easy is it to see everything? (would those hard of sight or colorblind be able to play the game?) How easy is it to play the game? (Would those with limited mobility be able to play the game?) Students may not have to make any particular changes to address accessibility concerns, but the key point here is that they consider how those with different abilities would use their game.

At the end of this lesson, each student should have a game that has either the same or very similar functionality to the fish game, but now has some sort of positive message applicable to the real world.

Reflection

Our games don’t really “solve” any particular issue. But do they help in some way? Do you think playing a game that encourages good behaviors would encourage you to also behave well every day?

What about a game that draws attention to an issue? Even though it does nothing to directly solve the issue, is raising awareness of the issue still helpful?

The point of these reflection questions is to hopefully help the students see how even at a young age, they can use computer science to make some sort of positive impact on others.
Presentation and Feedback
We Can Make Games Too: Lesson 11

Lesson Overview

Introduction
5 minutes

Gallery Walk
20-30 minutes

Reflection
5 minutes

Standards
IC-03 Seek diverse perspectives

Materials
1 Laptops
2 Sticky Notes
Introduction

Note: Please see the “Giving Feedback” lesson in the Miscellaneous folder. If you have not discussed giving constructive feedback with your class before, we strongly recommend teaching the Giving Feedback lesson plan before having students present their apps.

We are ending the unit by having students present their games to the class!

Remind students that everyone worked hard on these projects, and that we want to show support for each other and be very respectful. Also, it’s good to ask questions about the apps to learn more about them!

Start by telling students how we will be doing the presentations. If you have your own desired presentation method, feel free to use it. Otherwise, here’s a method you can use:

Assign each group (or student if done individually) as either being Red or Blue. The Red groups will present their app first. Students in the Blue groups will have sticky notes and walk around to learn about the different games. You can time it so that groups rotate every 5 or so minutes, with each Blue group starting at a Red group’s computer. The Red groups will explain what problem they were tackling, show their game, and explain what it does to address the problem. Students then can write down a piece of positive feedback for the presenters.

After the Blue groups have finished rotating through, switch sides so that the Blue groups are now presenting, and repeat!
Gallery Walk

Now to do the presentations as described above! (unless you’re using your own method)

Before students begin listening to presentations, give them some sticky notes or note cards that they can write short feedback on and give to the groups they listen to. Some examples of what they can write are “My favorite part of your game was…”, or “I like how you used … in your game!” The feedback should not be generic like “this game is cool!” It should include something specific from that game.

As the presentations are happening, circulate the room listening to presentations and also make sure that all feedback is respectful and positive. If students are not following instructions, pause the presentations, review expectations with them, and then resume.

Note: If teaching remotely, students can still present! On your remote teaching application, allow students to screen share their games one at a time. Have them demonstrate the features of their work and explain how and why they made certain choices. Give each student a few minutes to present and then receive feedback from peers.

Reflection

- What do you think of your project?
- How is it in terms of accessibility?