

Collision Detection

(Also available for [WeScheme](#))

Students use function composition and the distance formula to detect when characters in their games collide.

Prerequisites	Piecewise Functions
Relevant Standards <div>CC-Math</div>	Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).
Lesson Goals	Students will be able to: <ul style="list-style-type: none">• Explain how the distance formula is related to the Pythagorean theorem.• Write a function for the distance formula.
Student-Facing Lesson Goals	<ul style="list-style-type: none">• I can explain how the distance formula is connected to the Pythagorean theorem.• I can write a function that takes in 2 points and returns the distance between them.
Materials	<ul style="list-style-type: none">• Lesson slides (Google Slides)• Sample game file - no distance lines (Pyret)• Sample game file - with distance lines (Pyret)• original (Page 58)
Supplemental Resources	<ul style="list-style-type: none">• Absolute Value (Desmos)• Absolute Value Inequality Illustrator (Geogebra)• Absolute Value (Quizizz)• Distance Formula (Geogebra)• Distance Formula (Quizizz)• Pythagorean Theorem (Quizizz)• Pythagorean Theorem (Geogebra)
Preparation	<ul style="list-style-type: none">• Make sure all materials have been gathered• Decide how students will be grouped in pairs
Key Points for the Facilitator	<ul style="list-style-type: none">• The distance formula is an excellent review of Circles of Evaluation. Have students work out the expression in small groups to foster discussion.

For a textbook-like version of materials similar to these, you may wish to see the [prior unit-based version](#)

Glossary

Boolean :: a type of data with two values: true and false

circle of evaluation :: a diagram of the structure of an expression (arithmetic or code)

pixel :: the smallest unit that makes up a digital image. The more pixels, the more detailed an image or video can appear.

Warmup

Problem Decomposition Returns!

20 minutes

Overview

Students revisit the problem decomposition concept from [earlier lessons](#).

Launch

Problem Decomposition is a powerful tool, which lets us break apart complex problems into simpler ones that we can solve, test, and then glue together into a complex solution.

Students may remember that there are two strategies for doing this:

1. **Top-Down:** Describe the problem at a high level, then fill in the details later
2. **Bottom-Up:** Focus on the smaller parts that you're sure of, then build them together to get the big picture

Problem Decomposition is the focus of [an entire Bootstrap lesson](#), is used to solve [onscreen?](#), and build up the 2-dimensional [distance function](#).

Investigate

For the following complex word problem, have students **first** decide which strategy they want to use, and then apply the Design Recipe to build the functions they need.

A retractable flag pole starts out 24 inches tall, and can grow at a rate of 0.6in/sec. An elastic is tied to the top of the pole and anchored 200 inches from the base, forming a right triangle. Define functions that compute the height of the pole and the area of the triangle after a given number of seconds.

Have students complete the [Top Down / Bottom Up \(Page 57\)](#) worksheet, using Problem Decomposition and the Design Recipe to solve this problem!

Synthesize

- Which strategy did students use?
- Did they start out with one, and then switch to another?

Collision Detection

20 minutes

Overview

Students once again see function composition at work, as they compose a simple inequality with the `distance` function they've created.

Launch

Knowing how far apart our characters are is the first step. We still need the computer to be asking: "True or False: is there a collision?"

Investigate

Using [Word Problem: is-collide \(Page 58\)](#), have students write a function that takes in two coordinate pairs (four numbers) of two characters (x1, y1) and (x2, y2) and returns a **Boolean** as to whether or not the two characters have gotten within 50 **pixels** of each other.

Synthesize

- Since students started out with the `distance` function first, which strategy are they using to decompose collision detection?
- Explicitly point out that this function is easy to write because we can *re-use* the `distance` function

- Explicitly point out that this function is easy to write because we can re-use the distance function.
 - Connect this back to `profit`, `revenue`, `cost` and `onscreen` from [previous lessons](#). Problem Decomposition is powerful!
-

Additional Exercises:

- For characters that are much taller than they are wide (or wider than they are tall!), using the radius to determine collision won't work very well. Have students compute the [Manhattan Distance](#) to take the more-rectangular dimensions of their characters.