STEM Careers Coalition™

ELEMENTARY CLASSROOM ACTIVITY

Testing the Code

OBJECTIVE

After creating detailed instructions that simulate code, students will perform quality assurance testing to ensure their peers' code functions optimally.

MATERIALS

- Game Design handout, enough for half the class
- Coloring materials, enough for the class to share
- Coding Sheet, enough for half the class

ENGAGE

- Ask students to raise their hand if they have ever made a sandwich *or* seen someone make a sandwich.
- Then instruct the class to imagine that they have a friend who has never made a sandwich, seen a sandwich being made, or even eaten a sandwich before...and it's now up to them to teach this person how to make a peanut butter and jelly sandwich.
- Recommend that they begin by writing a list of the ingredients and materials that this person will need. Record the list on the board as the class shares their thoughts. Encourage students to be as specific as possible.
- Then challenge students to explain the step-by-step instructions for making a PB&J and record the sandwich recipe on the board as the class develops it. Remind students to include as many details as possible, even if they seem obvious! When the class thinks the recipe is complete, roleplay the step-by-step instructions as a class. This will help students realize if they have left anything out. If any steps or details were missed, work together to edit the instructions as needed.
- Wrap up by leading the class in a brainstorm around other activities where detailed directions are important—such as assembling a piece of furniture or learning a new dance routine!

EXPLORE

- Explain that computers also rely on detailed instructions. Every small step that a computer completes is written in code. Code is a special computer language that tells the computer exactly what to do. It is similar to a recipe—when something is left out or a line is unclear, the computer will not run as it should.
- Tell the class to pretend that they will now be helping create a computer game. The opening scene of the game will show a character running on a path through a treacherous area full of obstacles!
- Divide the class into pairs and distribute one Game Design handout to each pair. Ask students to pretend to be video game designers and instruct each pair to create a path (by coloring squares on the grid) from Start to Finish. The path may be mostly straight, or it could wind all around. As long as the path consists of different squares that touch each other, the design of the path is up to them!
- Next, instruct the video game designers to draw obstacles on the rest of the grid that the player should avoid. It may be helpful to brainstorm some fun obstacles—like alligators or quicksand—and be sure to stress the importance of being school-appropriate.
- Then distribute a Coding Sheet to each pair and tell students that they will now pretend to be programmers. Explain that a software developer is responsible for bringing games to life by writing the code that will tell the computer how the game should run. Remind the class that the game will begin with an opening scene in which a character goes along the path they have drawn, and the computer needs to be told *exactly* where this character should go! If the code isn't clear, the character may run off the path and into one of the obstacles.
- Reiterate that the code needs to be as clear as possible. Each new direction (such as turn right, continue straight for two squares, or turn left) should be a new step. Then give the student pairs time to write their "code."

APPLY

- Explain that once a game is developed, it goes through quality assurance (or QA) testing. During QA testing, a QA tester uses the program and looks for errors or bugs.
- Tell the class that they are going to take turns being QA testers for their games' opening scenes, and pair partners together to make groups of four.
- Ask pairs to sit back-to-back so one pair is facing one direction and the second pair is facing the opposite direction. They should then exchange their Game Design handouts with each other.
- Next, instruct pairs to take turns reading the code that they wrote on their Coding Sheet to the other pair. As one pair reads their code, the other pair should follow their instructions and draw their path on the Game Design handout. Remind pairs to follow the instructions they receive and *not* the path that they see. The code may mistakenly lead them off the path. If this happens, they should continue to follow the instructions!
- Once both pairs have tested their code, they should return the Game Design handouts to the original programmers so each pair can see if their code worked and kept the player on their path. If not, students should edit their instructions and (if time allows) perform another round of QA testing.
- Finally, bring the class back together and discuss:
    - Were there any bugs in your code or glitches that needed to be fixed? What happened? What did you learn?
    - Why is it important for code to be as clear and specific as possible?

K–2 CONSIDERATIONS

- Keep the age level of the students in mind as you explain concepts and substitute simpler words as needed.
- Try to model sample responses to open-ended questions before you ask students to brainstorm or respond on their own.
- During the *Explore* activity, younger students may talk through their directions rather than writing them. Pairs can then sit side-by-side with another pair and verbally give their instructions instead of reading them.

GAME DESIGN

							FINISH		
START									

CODING SHEET

- Begin at *Start*.
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- You have safely reached *Finish*!