GAINING STEAM
CREATING GAMES WITH CODE

Coding and programming allow us to make technology do what we want it to do. This month’s activities let children explore the basics of coding by creating games. Children can complete either or both of the activities to gain an understanding of how coding directs technology to act in particular ways.

Coding for the Win

Explain that computers can only follow direct instructions; they cannot think for themselves or make accidental errors like people can. The instructions that computers follow are called code, and computer code must follow a specific format.

Demonstrate how coding works by leading a game of Simon Says. When you precede an instruction with the phrase “Simon says,” the children should do nothing because the “code” is incomplete and thus incorrect. Play Simon Says until the group seems to understand the concept of correct coding.

Then move the group to a space that has been set up with mats or papers on the floor in a tiled grid (e.g., 64 pieces of paper arranged in 8 rows of 8 sheets). Designate one tile as the “Start” tile. Place an object (e.g., a crown, trophy, or other object that suggests winning a game) on a random tile within the grid. Give each child an index card, and ask them to write a simple movement instruction on the card. Examples include “Move forward 1 tile,” “Move backward 2 tiles,” or “Move right 2 tiles.”

Shuffle the index cards to create a code deck. Invite one child to stand on the “Start” tile and play the game. Draw a card from the code deck and read the instruction to the child playing the game. The child must follow the instructions from the code deck, with the end goal of landing on the tile with the winning object. Make sure the child faces the same direction for the entirety of his or her turn. If an instruction causes the child to leave the grid, his or her turn comes to an end, and another child can begin from the “Start” tile.

With the code deck in a random order, it should be difficult for players to get from the “Start” tile to the winning
object quickly. Lead a conversation about why this type of random code isn’t efficient for completing a specific task. Ask the children how they could use the code deck to reach the winning object every time and more quickly; lead them toward the idea of arranging the code deck in a specific order that makes moving through the grid easier. Break the children into two or three teams, giving each team some of the code-deck cards. Have teams work together to arrange, or program, their code deck so that a player on the “Start” tile can quickly move to the winning object every time.

When all groups have programmed their code decks and tested their codes to make sure they work, bring the groups back together to see how each group completes the game. Let each team demonstrate its code, and help children track which team’s code seems to “win” the game the quickest.

Close the activity with a discussion about how coding means putting specific instructions in a specific order so that a program can achieve a certain goal. Make sure to include booktalks of titles that include gaming and programming, like Chris Grabenstein’s Escape from Mr. Lemoncello’s Library and Gordon Korman’s Ungifted.

Creating a Joke Computer Game

Start with children facing a blackboard, whiteboard, or other space where instructions can be written and viewed. Introduce the concept of if/then statements and let the children experience some if/then statements in action by writing two or three “rules” on the board; for example, “If the leader claps her hands, everyone else stumps their feet.” After the rules have been written, the leader performs a series of actions, and the children try to respond according to the if/then rules. Talk about how it can be difficult to keep all the rules straight, especially when there are lots of rules or when they must act quickly. Computers, which are programmed with rules, do not have this problem.

Transition to computer stations to put the if/then knowledge into practice by creating a simple video game. Before the program, set up a library account for Scratch (https://scratch.mit.edu), a free web-based visual programming language; then allow children (either individually or in pairs) to create a project within the library’s account.

The goal is for children to create a video game in which, through a series of “If … then” rules, the player learns the answer to a joke. (To see an example of a completed game in Scratch, visit http://tinyurl.com/lj6qvdq. Click on the blue square at the top left of the game screen to play.) Have some joke books on hand, like Rob Elliott’s More Laugh-Out-Loud Jokes for Kids, in case children need some inspiration.

After clicking “Create” to start a new project, have children create a backdrop by clicking on the paintbrush icon in the lower left of the screen (see www.librarysparks.com for illustrated instructions). Select “backdrop 1” from the editing sidebar; then use the text function to type the beginning of the joke (e.g., “What has a face and two hands but no arms or legs?”). Note: Use the return/enter key to break up long lines of text.

Use the line function to create blank spaces for each letter in the joke’s answer (e.g., “a clock”). The backdrop for the game is now complete, although students can add decorations.

Create “sprites” for all the letters that will appear in the joke’s answer, as well as a few extra letters to add an element of guessing. First, modify “Sprite1” (a cat by default). Select the sprite; then select “New Costume: Choose costume from library” in the editing menu in the center of the screen. Select “Letters” from the list in the sprite options, and choose the letter and style desired. For additional letters, select “New sprite” from the lower left window. Then select the letters from the available characters. Move the sprites into place by clicking and dragging each one into the game view screen (top left window).

Once the sprites are created, each one requires an “If … then” code in the game. To add code to a sprite, click on that sprite in the bottom left window; select the “Scripts” tab in the editing menu. Each sprite should receive two pieces of code: one to indicate what happens when the green flag is clicked (i.e., when the game is restarted) and a second to indicate what happens when the sprite is clicked. These represent “if” statements. The code blocks are found under the Events category. Click and drag the “if” statements into the editing field.
Children should add “then” statements from the “Motion” and/or “Sound” categories. For example, when the green flag is clicked, each sprite will return to its starting position. With the sprite in the desired starting position in the game view screen, go to the “Motion” category and select “go to x: 00 y: 00.” The coordinates for x and y should autopopulate. Drag the code block into the left window and attach it to the “If” statement for the green flag. When sprites that are part of the joke’s answer are clicked, they should be programmed to move to the appropriate blank space in the answer. Children can set the coordinates for this move by clicking and dragging the sprite to the correct position in the game view screen. They should then return to the “Scripts” menu and repeat the previous step, attaching the code block to the “If” statement for when the sprite is clicked (the new coordinates should autopopulate in the code). Sprites that are not part of the answer can be programmed to make a noise, such as a meow, when they are clicked. Under “Scripts,” select the “Sound” category; then attach the “play sound” code to the “If” statement for when the sprite is clicked.

When all sprites have been assigned their code, the game is ready to play. Have all the children click the blue “See Project Page” button in the upper right corner. Then have them change seats to try one another’s joke games. Be sure to let children know you have books on programming, like Super Scratch Programming Adventure! from the LEAD Project, available if they want to explore coding further.

**Reading Connections**

**Fiction**

*Escape from Mr. Lemoncello’s Library* by Chris Grabenstein. 3–6. Kyle is a huge gaming enthusiast. When the eccentric Mr. Lemoncello selects Kyle and some of his fellow students to explore his new library overnight, Kyle is in for the game of his life.

*Ungifted* by Gordon Korman. 4–6. Donovan has a tendency to get into trouble with his love of pranks. When his latest prank lands him in a gifted-and-talented program instead of reform classes, he thinks he’s in nerd city—until he sees what the class can do with robotics.

**Nonfiction**

*More Laugh-Out-Loud Jokes for Kids* by Rob Elliott. 2–6. This volume is packed with hilarious jokes, riddles, and tongue twisters to amuse all ages.

*Super Scratch Programming Adventure! Learn to Program by Making Cool Games!* by the LEAD Project. 3–6. This graphic novel–style how-to guide gives readers a clear, simple introduction to Scratch and all its gaming capabilities.

**Amy Koester** is the youth and family program coordinator with Skokie Public Library. She has presented at conferences nationwide on the topic of STEAM programs and services for children, and she regularly blogs about program plans and ideas for the ALSC Blog and as the Show Me Librarian. Visit her website at http://showmelibrarian.blogspot.com.