code jumper

Lesson 5
Debugging

Code Jumper Curriculum: Lessons

Lesson 5
Debugging

American Printing House for the Blind

Published by American Printing House for the Blind
1839 Frankfort Avenue, Louisville, KY 40206
www.aph.org | info@aph.org

Overview

Lesson Objectives

Students will:

- Understand the concept of a bug in computer programming
- Identify and fix simple bugs in their own programs

Expected Outcomes

Students will:

- All students: Identify simple bugs in existing programs
- Most students: Identify and fix simple bugs in existing programs
- Some students: Identify and fix bugs in existing programs and explain their process for correcting the error in the program

Lesson Plan Structure

- Unplugged Activity
- Guided Code Jumper Activity
- Exploration
- Standards and Check for Understanding

Resources

- Code Jumper Tutorial Videos
  - Code Jumper App: https://www.youtube.com/watch?v=vg72YPz6CWY
  - The Hub: https://www.youtube.com/watch?v=KGb51PW9zJQ&lis=
  - Play and Pause Pod: https://www.youtube.com/watch?v=446jCw8qcDI&t=
- Code Cards

Key Vocabulary

- **Bug:** An error or problem within a computer program.
- **Debugging:** The process of finding and fixing bugs (or errors) in computer programs.
- **Software Engineer:** Engineers who write computer programs for different types of software.

Materials

- Sample program set up and run by the teacher: Twinkle, Twinkle with an Error
- Sound Set: MIDI Instruments and then Piano at Thread 1:

```
THREAD 1 Piano
    PLAY C5 for 1/2 a beat
    PLAY C5 for 1/2 a beat
    PLAY G5 for 1/2 a beat
    PLAY F5 for 1/2 a beat
    PLAY A5 for 1/2 a beat
    PLAY A5 for 1/2 a beat
    PLAY G5 for 1/2 a beat
END THREAD
```

Unplugged Activity

Objective

Introduce the concept of programming bugs and learn how to identify the programming error in order to fix it and make sure the program runs smoothly.

Vocabulary

Bug: An error or problem within a computer program.

Debugging: The process of finding and fixing bugs (or errors) in computer programs.

Software Engineer: Engineers who write computer programs for different types of software.

Materials

- Five sets of Debugging Cards containing five cards each:
  - Each set of cards will contain four correct math problems and one incorrect problem
  - Math problems should be relatively easy for students to solve and can be tailored to the skill level of the group
  - See Debugging Cards at the end of the lesson

Instruction

1. Divide students into groups of three to five and have them select one group member to hold their group's stack of Debugging Cards.
2. Have the card holder lay out the Debugging Cards in a line next to each other, from left to right.
3. Once all groups have their cards lined up on a desk/table, explain that one of the cards contains a bug—that is, an error—and review with them that debugging is the process of finding and fixing bugs in computer programs
4. Explain to students that they will start with the first card on the far left and the goal is to determine if the card contains a bug. All group members must agree that the card is correct, and one group member must be able to explain their thinking.
5. Once the group agrees that the card is correct, continue the process with the next card in the line to determine if the card is correct or has a bug.
   a. If students identify the problem incorrectly, ask them if they are sure and encourage them to solve the problem without relying on the fact that the card is correct.

6. Have each group continue the process for each card on the desk/table.
7. When the students reach the card that is incorrect, ask them what is wrong with the problem and then have them discuss with their group the change that needs to be made for the card to be correct. Students will need to be able to explain their thinking as a group.
8. When each group of students completes their card set, review with the entire class how, in their small groups, they were able to **debug** a sequence of instructions.
9. Explain that the process they participated in is similar to what software engineers go through when **debugging** a computer program. Explain that software engineers typically work with a team, and when something in the program is not working correctly they systematically go through the program and determine where the error is and problem-solve how to fix it.

Closure

10. Explain that in Code Jumper they will experience bugs in their programming and will have to work to debug the programming sequence in order for it to work correctly.

Guided Activity: Debugging Code Jumper

In this activity, students will analyze code for bugs/errors and will learn how to identify and correct a bug.

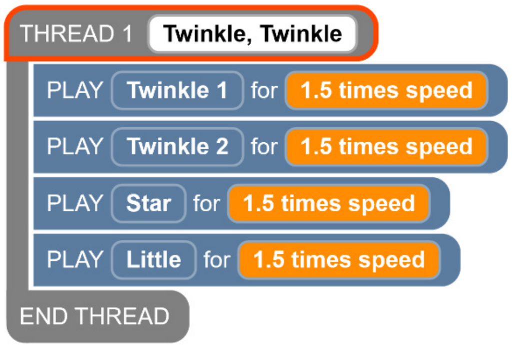
Objective

- Listen to a sample program and identify a bug
- Correct the bug in the program

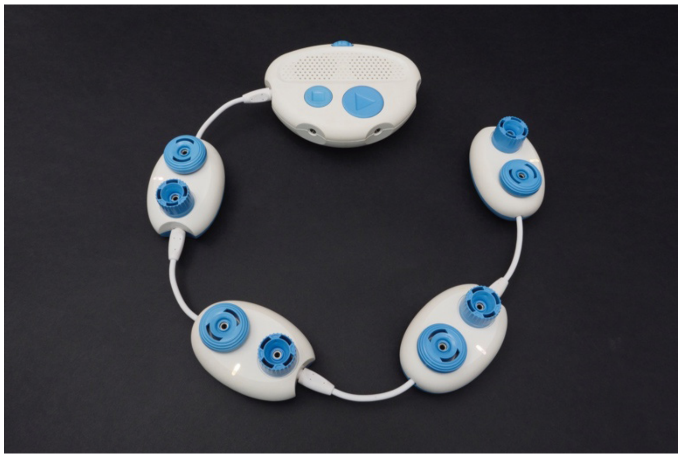
Materials

Debugging Code Card for Twinkle, Twinkle with an Error:

```
THREAD 1 Twinkle, Twinkle
    PLAY Twinkle 1 for 1.5 times speed
    PLAY Twinkle 2 for 1.5 times speed
    PLAY Star for 1.5 times speed
    PLAY Little for 1.5 times speed
END THREAD
```



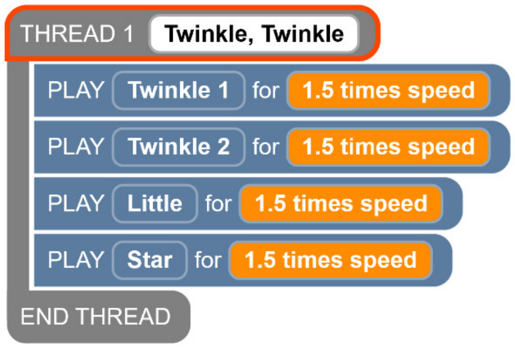
```
THREAD 1 Twinkle, Twinkle
    PLAY Twinkle 1 for 1.5 times speed
    PLAY Twinkle 2 for 1.5 times speed
    PLAY Star for 1.5 times speed
    PLAY Little for 1.5 times speed
END THREAD
```



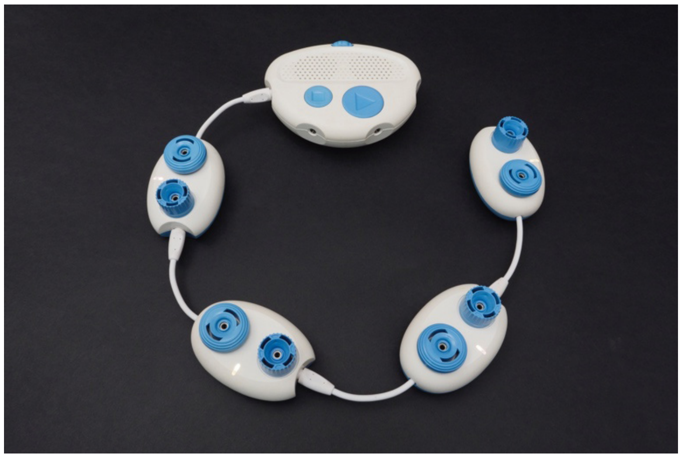
[Figure Caption:] At the top is a screenshot of the Code Jumper app showing a program that has one thread. Under THREAD 1 Twinkle, Twinkle, the commands read, PLAY Twinkle 1 for 1.5 times speed, PLAY Twinkle 2 for 1.5 times speed, PLAY Star for 1.5 times speed, PLAY Little for 1.5 times speed; the commands are followed by End Thread. Below this screenshot is a photo of a Code Jumper program, with four Play pods connected to the Hub at Port 1.

Code Card for Twinkle, Twinkle Without an Error:

```
THREAD 1 Twinkle, Twinkle
    PLAY Twinkle 1 for 1.5 times speed
    PLAY Twinkle 2 for 1.5 times speed
    PLAY Little for 1.5 times speed
    PLAY Star for 1.5 times speed
END THREAD
```



```
THREAD 1 Twinkle, Twinkle
    PLAY Twinkle 1 for 1.5 times speed
    PLAY Twinkle 2 for 1.5 times speed
    PLAY Little for 1.5 times speed
    PLAY Star for 1.5 times speed
END THREAD
```



[Figure Caption:] At the top is a screenshot of the Code Jumper app showing a program that has one thread. Under THREAD 1 Twinkle, Twinkle, the commands read, PLAY Twinkle 1 for 1.5 times speed, PLAY Twinkle 2 for 1.5 times speed, PLAY Little for 1.5 times speed, PLAY Star for 1.5 times speed; the commands are followed by END THREAD. Below this screenshot is a photo of a Code Jumper program, with four Play pods connected to the Hub at Port 1.

Instruction

1. Set up a program like Twinkle, Twinkle. Make sure that the program is something that the students are familiar with so they can easily spot an error. If Twinkle, Twinkle is not appropriate for your specific classroom, choose a different program or add in a custom Sound Set. Create an error in the program and run it so that all students can hear.
2. Run the program on normal speed and ask students to listen carefully. Oh no! Something is not right! Ask students what they think has happened.
3. Ask students: Is something wrong in the program? What do we call it when the Code Jumper program has a problem? (Answer: A **bug**—that is, an error.)
4. Ask students: How do we fix the problem? **Debugging** is the process of finding and fixing bugs in computer programs. Talk as a group as to what it means to fix an error in their program. Have students write the definitions of bug and debugging in their Computer Science Journals.
   a. **Bug:** An error in a computer program.
   b. **Debugging:** The process of finding and fixing bugs (or errors) in computer programs.
5. Run the program again and ask students to listen very carefully to the program Twinkle, Twinkle with an Error again.
6. As a class, sing the song, poem, etc. as a refresher of what the program should be running.
7. Ask students: Can you identify the error? Which part of the song do you think is wrong? (Expected response: Star and Little are reversed.)
8. Ask students: Who do you think fixes errors on the computer programs that you use every day? Correct the error and play again.

9. Introduce the term **software engineer**. These are people who write computer programs. Have the students write the definition of Software Engineer in their Computer Science Journals.
10. **Software Engineer:** A software engineer is a person who applies the principles of software engineering to the design, development, maintenance, testing, and evaluation of computer software. (A simpler definition would be people who write computer programs.)
11. Ask students: What ways do you think a Code Jumper Program could be broken? (Expected responses: The sound or duration parameters could be wrong, the Pod could be plugged into the wrong thread, the wrong Sound Sets could be selected, etc.)
12. Give each group the Code Card below in step 15 or one that is more appropriate for the specific class. Ask the students to create the code exactly as it is written on the Code Card. In this activity, students will be looking for a bug. First, they will need to listen to all the sounds and figure out which one does not belong in the group. The sound that does not fit in the group is the bug and needs to be replaced with a sound that is also an animal.
13. Ask students to create the code and listen carefully and discuss in their small groups what category of sound it is. Ask the students to trace the code while they are listening. Have students write down in their Computer Science Journals the sounds that they hear, what the error is, and which Pod has the error.
14. Once the error has been determined, students can then find the correct sound to replace the bug with and then play it for the teacher in the correct order. **Tip for Teachers!** Create the code correctly first and then add in the bug when creating the Code Cards. Errors can be added to the program by making some changes to the sound and duration parameters.

15. In the program below, the pattern is that all the sounds are animal sounds. The error is in Play pod 3, which is Thunder and should be Birds. (Thunder is not an animal sound so it does not fit this category.)

Program with an error:

```
Thread 1 Nature
    Play Sea Gulls, speed: 1 times
    Play Crickets, speed: 1 times
    Play Thunder, speed: 1 times
    Play Frogs, speed: 1 times
End Thread
```

Program without an error:

```
Thread 1 Nature
    Play Sea Gulls, speed: 1 times
    Play Crickets, speed: 1 times
    Play Birds, speed: 1 times
    Play Frogs, speed: 1 times
End Thread
```

16. Ask students to recreate the code EXACTLY as it is on the Code Card. Once the code is complete, have the students run the code and listen carefully.

    By tracing the code, students should determine which line of code or Pod has the error.
17. Ask students to identify the error on the Code Card. If this is done in braille, ask the students to mark the incorrect line of code with a tactile sticker or wax stick.
18. Students should then correct the code in Code Jumper and

run the code again. On the Code Card, ask students to write the correct line of code from the Sound Set or duration. If braille is being used, ask the student to write the correct line of code on a separate piece of paper and attach it to the original Code Card.

Closure

1. Ask students to explain what a bug is and to give an example of one they found in one of this lesson's programs.
2. Ask students to explain what debugging is and to give an example of how they debugged their programs.

Exploration

Overview

In this activity, students will create a program and then introduce a bug for another group to find and fix.

Materials

- Code Jumper kit
- Computer Science Journal

Instruction

1. Ask students to create a program. It may be one from the Sample Sounds Sound Set or from a custom Sound Set.
2. Have students create the Code Card for their program in their Computer Science Journals.
3. Ask students to put a bug in the program and indicate where it is in their Computer Science Journals.
4. Have students switch groups and try to find the bug in the other programs. Instruct students to record in their Computer Science Journals where the bug is. Which Pod was incorrect? For example, Play pod 3 out of 6 Pods.

5. Ask students to regroup and explain the discovery to the creator of the program.

Closure

Ask students to reflect on strategies they used to find the bug and then record these strategies in their Computer Science Journals.

Standards and Check for Understanding

CSTA K-12 Computer Science Standards*

- 1A-AP-14: Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
- 1B-CS-03: Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies.
- 1B-AP-15: Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.

National Curriculum of England*:

Key Stage 1:

- Understand what algorithms are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions
- Create and debug simple programs
- Use logical reasoning to predict the behavior of simple programs
- Use technology purposefully to create, organize, store, manipulate and retrieve digital content
- Recognize common uses of information technology beyond school
- Use technology safely and respectfully, keeping personal

information private; identify where to go for help and support when they have concerns about content or contact on the internet or other online technologies

Key Stage 2:

- Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts
- Use sequence, selection, and repetition in programs; work with variables and various forms of input and output
- Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs
- Use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content
- Use technology safely, respectfully and responsibly; recognize acceptable/unacceptable behavior; identify a range of ways to report concerns about content and contact

Closing Activities and Check for Understanding

Have students identify situations in their school where they have run into a problem with a series of steps that they had to follow. Discuss how they identified that it was a problem and have students write what steps they took to solve the problem in their Computer Science Journals.

Possible Examples:

- Working on math problems
- Writing a paper
- Doing a science experiment

Check for Understanding	**Completed**
Student can identify an error/bug in a sequence of actions.	Yes / No
Student can explain what a bug is.	Yes / No
Student can describe the steps they would take to identify a bug.	Yes / No

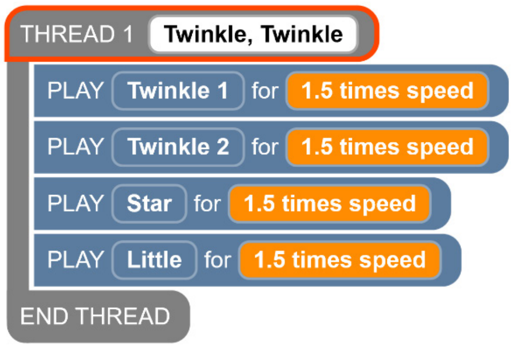
*Computer Science Teachers Association (2017). CSTA K-12 Computer Science Standards, Revised 2017. Retrieved from http://www.csteachers.org/standards

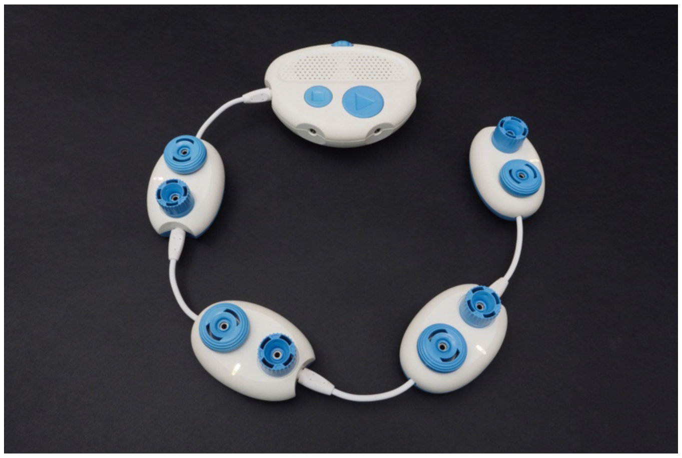
*Education, Department for. "National Curriculum in England: Computing Programmes of Study." GOV.UK, 11 Sept. 2013, www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study

Code Cards

Debugging Code Card for Twinkle, Twinkle with an Error for Guided Activity

```
THREAD 1 Twinkle, Twinkle
    PLAY Twinkle 1 for 1.5 times speed
    PLAY Twinkle 2 for 1.5 times speed
    PLAY Star for 1.5 times speed
    PLAY Little for 1.5 times speed
END THREAD
```

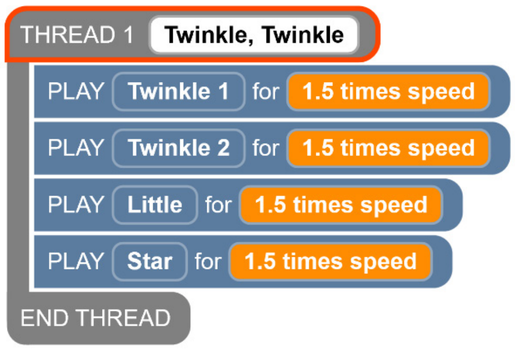

THREAD 1 Twinkle, Twinkle
PLAY Twinkle 1 for 1.5 times speed
PLAY Twinkle 2 for 1.5 times speed
PLAY Star for 1.5 times speed
PLAY Little for 1.5 times speed
END THREAD



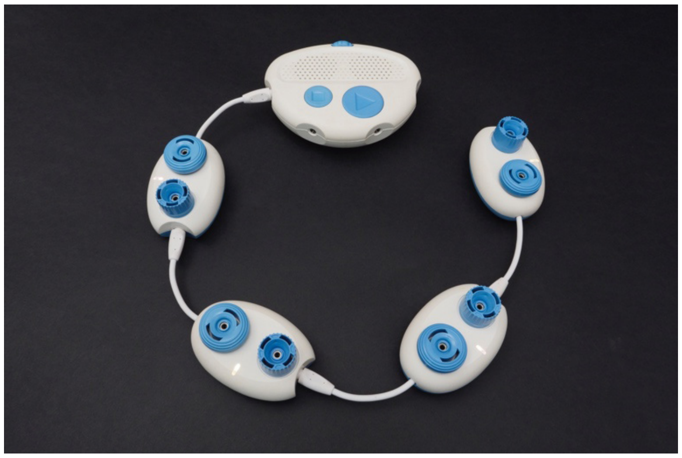
[Figure Caption:] At the top is a screenshot of the Code Jumper app showing a program that has one thread. Under THREAD 1 Twinkle, Twinkle, the commands read, PLAY Twinkle 1 for 1.5 times speed, PLAY Twinkle 2 for 1.5 times speed, PLAY Star for 1.5 times speed, PLAY Little for 1.5 times speed; the commands are followed by End Thread. Below this screenshot is a photo of a Code Jumper program, with four Play pods connected to the Hub at Port 1.

Code Card for Twinkle, Twinkle without an Error for Guided Activity

```
THREAD 1 Twinkle, Twinkle
    PLAY Twinkle 1 for 1.5 times speed
    PLAY Twinkle 2 for 1.5 times speed
    PLAY Little for 1.5 times speed
    PLAY Star for 1.5 times speed
END THREAD
```



```
THREAD 1 Twinkle, Twinkle
    PLAY Twinkle 1 for 1.5 times speed
    PLAY Twinkle 2 for 1.5 times speed
    PLAY Little for 1.5 times speed
    PLAY Star for 1.5 times speed
END THREAD
```



[Figure Caption:] At the top is a screenshot of the Code Jumper app showing a program that has one thread. Under THREAD 1 Twinkle, Twinkle, the commands read, PLAY Twinkle 1 for 1.5 times speed, PLAY Twinkle 2 for 1.5 times speed, PLAY Little for 1.5 times speed, PLAY Star for 1.5 times speed; the commands are followed by END THREAD. Below this screenshot is a photo of a Code Jumper program, with four Play pods connected to the Hub at Port 1.

Debugging Practice Sets

Debugging Practice Set #1:

$$7 - 4 = 3$$

$$6 + 2 = 8$$

Debugging Practice Set #1:

$$9 + 2 = 13$$

$$5 - 1 = 4$$

Debugging Practice Set #1:

$$8 + 2 = 10$$

Debugging Practice Set #2:

$$9 - 4 = 5$$

$$2 + 10 = 12$$

Debugging Practice Set #2:

$$4 + 3 = 7$$

$$2 - 1 = 0$$

Debugging Practice Set #2:

$$9 + 3 = 12$$

Debugging Practice Set #3:

$$8 - 4 = 4$$

$$1 + 9 = 10$$

Debugging Practice Set #3:

$$7 + 5 = 12$$

$$8 - 1 = 6$$

Debugging Practice Set #3:

$$9 + 0 = 9$$

Debugging Practice Set #4:

$$10 - 4 = 6$$

$$3 + 2 = 5$$

Debugging Practice Set #4:

$$5 + 4 = 8$$

$$9 - 3 = 6$$

Debugging Practice Set #4:

$$4 + 3 = 7$$

Debugging Practice Set #5:

3 – 0 = 3

6 + 6 = 12

Debugging Practice Set #5:

$$8 + 1 = 9$$

$$12 - 6 = 6$$

Debugging Practice Set #5:

$$4 + 7 = 13$$

Text only for complete set of debugging cards

Practice Set #1:

7-4=3

6+2=8

9+2=13

5-1=4

8+2=10

Practice Set #2

9-4=5

2+10=12

4+3=7

2-1=0

9+3=12

Practice Set #3

8-4=4

1+9=10

7+5=12

8-1+6

9+0=9

Practice Set #4

10-4=6

3+2=5

5+4=8

9-3=6

4+3=7

Practice Set #5

3-0=3

6+6=12

8+1=9

12-6=6

4+7=13

Debugging Practice Set #1: (Blank)

Debugging Practice Set #1: (Blank)

Debugging Practice Set #1: (Blank)

Debugging Practice Set #2: (Blank)

Debugging Practice Set #2: (Blank)

Debugging Practice Set #2: (Blank)

Debugging Practice Set #3: (Blank)

Debugging Practice Set #3: (Blank)

Debugging Practice Set #3: (Blank)

Debugging Practice Set #4: (Blank)

Debugging Practice Set #4: (Blank)

Debugging Practice Set #4: (Blank)

Debugging Practice Set #5: (Blank)

Debugging Practice Set #5: (Blank)

Debugging Practice Set #5: (Blank)

code jumper

For more resources, visit codejumper.com

AMERICAN PRINTING HOUSE

1839 Frankfort Avenue
Louisville, Kentucky 40206
502-895-2405 • 800-223-1839
www.aph.org • info@aph.org