



Short Answer

Teaching Tips



To help foster a supportive environment for feedback, introduce the [importance of feedback](#) and [how to give empathetic feedback](#) with our mini-lessons.



The best questions are open-ended and ask students to **justify** opinions, **analyze** material, **articulate** a thought process, or **evaluate** a claim.



Avoid fact-based recall questions.



The best feedback criteria are **positively oriented** and relevant to learning objectives. Criteria can focus on both **content knowledge** and **writing structure**.

- Make students aware of the criteria so they know how to craft their response. Or, have students decide what appropriate criteria would be.
- Emphasize that students have to make a choice about assigning feedback. *Both* or *neither* aren't options!



The best discussion questions ask students to **verbalize their thought process** about the feedback they gave. For example: "93% you said Response 1 explained the concept of photosynthesis better. Can someone share what aspect of the response made you think that?"

- Have students **predict** what results will be.
- Focus discussion on the qualities of the **responses** rather than on the students who wrote the responses.



To promote students' **metacognitive development**:

- Encourage students to **ask themselves questions** during the activity:
 - "Does my response meet all the criteria?"
 - "How did giving feedback improve my understanding of the content?"
 - "How can I improve my response using the feedback I got?"
- Provide clear **time signals** throughout the activity.



To **incorporate feedback in the moment**, ask students to reflect on how they can improve their response after receiving feedback. You may want to take time to have them **revise** their responses, either in class or for homework.




Computer Science

In all Short Answer activities, your students **create** responses, **compare** peer responses and provide scaffolded feedback, then **converse** results as a class.

Short Answer gets your students the immediate feedback they need through social, engaging peer feedback activities and gets you deeper insight into what your students know.

Short Answer can be used at every stage of your Computer Science lesson plan from problem solving skill foundations to code review.

| | |
|--------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Bellringer | Get students engaged by using Short Answer to prompt written responses about what stands out from yesterday's class or to preview new material with a warm-up problem. |
| Check for understanding | Break up lectures with quick feedback activities that get students interacting with one another. Deepen understanding while getting a quicker, more accurate pulse of what they know on an individual level. |
| Guided practice | Group students together to write responses, solve problems, and give feedback to other groups, or provide a model response in Short Answer. |
| Independent practice | During Short Answer activities, encourage students to reflect on how their response matches up to the ones voted as the strongest by the class. Invite revision and iteration of responses as another in-class activity, exit ticket, or homework. |
| Exit Ticket | Complete a quick, one-round Short Answer activity to leave students thinking about the most important points of the day. |
| Homework | Have students complete writing assignments about core content and bring them in next class for peer feedback activities and discussion to deepen understanding. |

 See the following page for two detailed Computer Science use cases with example questions, feedback criteria for students to evaluate responses with, and standards alignments.

Sample Use Cases: Computer Science

Note: Short Answer doesn't support code-friendly formatting yet, but works well when discussing generalized problem solving strategies and procedures.

Procedural Knowledge and Problem Solving

Activity Time: 5-10 minutes

Before introducing a single line of code, use Short Answer to hone students' problem solving and logical thinking skills by asking how they would tackle problems.

Sample Questions

- Describe how you would find the smallest and largest number in an unsorted integer array.
- What steps would you use to reverse a string in place?
- How can you swap the numerical value of two variables without using a third placeholder variable?

Feedback criteria: efficient; accurate; feasible; creative

Standards Alignment Examples

- Compare and refine multiple algorithms for the same task and determine which is the most appropriate (*Computer Science Teachers Association, 1B-AP-08*)
- Use flowcharts and/or pseudocode to address complex problems as algorithms. (*CSTA, 2-AP-10*)
- Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. (*CSTA, 2-AP-13*)

COMING SOON: Coding Feedback

Activity Time: 10-15 minutes

Have students copy-and-paste code snippets or create original code in Short Answer, then give feedback to peers on how code can improve. Students can see how the same problem can be solved with different lines of code and discuss strategies for optimization.

Sample Questions:

- Write a function `get_average` that returns the average of all exam scores in the attached dataset.
- Insert your code to get the robot from A to B while picking up all objects in the path

Feedback Criteria: efficient; concise; desired result occurs; creative

Standards Alignment Examples

- Create programs that include sequences, events, loops, and conditionals. (*CSTA, 1B-AP-10*)
- Compare and refine multiple algorithms for the same task and determine which is the most appropriate (*Computer Science Teachers Association, 1B-AP-08*)
- Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation, and review stages of program development. (*CSTA, 1B-AP-16*)