Best Practices in Computer Science Classes

GBEA, September 2017
Pam Whitlock
Presented by:

- PAM WHITLOCK
  - CHATTANOOGA HIGH SCHOOL
  - Courses taught: APCS, AP CSP, CSP, PGAS
  - Presentation content created by Pam Whitlock and Dr. LeahDee Kilgore
Take out a piece of paper and a pencil and draw a shape from these directions...

- This geometric drawing can be made by drawing a four-inch circle. Draw the horizontal and vertical diameter \(ab\) and \(cd\). Mark the point of intersection \(e\). Bisect \(eb\) and mark the point of intersection \(f\). With \(f\) as a center, and \(cf\) as a radius, describe an arc cutting \(ae\). Mark the point of intersection \(g\). With \(gc\) as a radius and \(c\) as a center, describe two arcs cutting the circumference at \(h\) and \(j\). With \(h\) and \(j\) as centers and the same radius, describe arcs cutting the circumference at \(k\) and \(l\). Connect \(c\) and \(l\), \(c\) and \(k\), \(h\) and \(k\), \(l\) and \(j\), and \(h\) and \(j\) to complete the drawing.
Turn the paper over and let’s draw one more shape...

Draw a Star with a circle around it.
Which drawing was easier?

- Both instructions asked you to draw a star so the end product was the same.
- The first set of instructions...
  - Had complicated instructions that were long and hard to read.
  - Used language that may have been unfamiliar to some
  - Looked intimidating which may have caused some of you to immediately get nervous, anxious or feel defeated before you even started.
- The second set of instructions...
  - Was more simple and familiar.
  - Many of you were more successful with this task.
Best Practice #1: Connect Concepts to Prior Knowledge

- Many students do not understand unfamiliar or abstract code.
- Use game or demonstrations to introduce coding concepts
- Students need to understand the logic behind the concept and the need for the concept before they see the code.
- Examples:
  - Conditionals: Simon Says
  - Loops: Do the “wave” around the room until a condition is met
  - Arrays: egg cartons or pill boxes with manipulatives
  - Random: rolling dice or spinning a wheel
  - String methods: play scrabble
Best Practice #2: Code Progression

- **Example Code:** Show good quality code and have the students predict the output. You are modeling high quality code.

- **Scaffolded Code:** Take good quality code and remove a few lines. Students need to fill in the code to create the expected outcome. This can be used in large group settings, small groups, partners or individually.

- **Code with Errors:** Give students code that contains errors. Students must go through the debugging process to fix the code.

- **From Problem to Code:** Finally, give students a problem and they write the code to solve it. **Problem > Analysis** (what is problem asking you to do, what are you given & what do you need) > **Pseudocode > Code**
Best Practice #3: Live Coding

1. **Present** a problem to the class.
2. **Brainstorm** an algorithm to solve the problem. Write pseudocode on the board.
3. **Code the solution together.** Either show code thru a projector or use software to display the teachers monitor to the students’ monitors. Possible logistics...
   a. Teacher types the code. Students give directions.
   b. Teacher randomly picks a student to type the code.
4. **Make mistakes!** No one programs perfectly. You are modeling debugging, problem solving, understanding error messages, how to read an API, etc.
Best Practice #4: Pair Programming

- **What is pair programming?** Students work together in pairs to code a solution to a problem. One student is the driver and the other is the navigator.
- **Driver:** this student types the code as directed by the navigator.
- **Navigator:** this student determines the approach/direction.
- **Switch roles every 10 minutes.**
- **Why?** Pair programming forces students to talk about their problem solving approach and coding solutions. When students verbalize their thought processes, they collaborate, notice more details and enhance their coding abilities.
Best Practice #5: Refine existing code
When you get a coding solution to a specific problem, challenge the students to find an alternative way of coding the same problem.

//students are asked to make
//“Howdy” appear 5 times

//first attempt
println("Howdy");
println("Howdy");
println("Howdy");
println("Howdy");
println("Howdy");

//this solution works
for (var i = 0; i < 5; i++)
println("Howdy");

//this also works
var x = 5;
while (x > 0) {
    println("Howdy");
    x--;
};
Best Practice #6: Give Immediate, Targeted Feedback

Poor feedback:
- Good job!
- That’s wrong.
- Try again.
- Cool!

Excellent Feedback:
- It was very clever to use a loop and a conditional to alternate the type of output.
- How can you change your code to only output the even numbers?
- You are repeating this same block of code many times. How can you make this more efficient?
- What would happen if...?
Ask for feedback too!

Question 1

//What is the result of the following expression?

\[1 + 2 \times 3 + 7 \times 2 \mod 5\]

Question *

- 11
- 2
- 5
- 21
- 1
What do you want to review?

- Arrays
- 2D arrays
- ArrayLists
- Searching
- Sorting
- Nothing...I just want class time to review independently
- Other
What do you want to review?

26 responses

- Arrays: 2 (7.7%)
- 2D arrays: 13 (50%)
- ArrayLists: 8 (30.8%)
- Searching: 13 (50%)
- Sorting: 15 (57.7%)
- Nothing...I just want: 6 (23.1%)
- Other: 0 (0%)
Search and Sort 7th period

Form description

Name

Short answer text

Select all that apply *

☐ I would like to have guided practice on the Search Lab

☐ I would like to have guided practice on the Sorts Lab

☐ I'm ok...I just need time to code.
Select all that apply

23 responses

- I would like to have more: 6 (26.1%)
- I would like to have less: 10 (43.5%)
- I'm ok...I just need: 14 (60.9%)
Best Practice #7: Teach Time Management Skills

Problem:
- New programmers do not have a frame of reference to determine how long a programming task will take. Programming is often a BRAND NEW skill for our students.

Solutions:
- Debrief at the end of class. What surprised you about this task? Discuss the amount of time specific tasks required.
- Give students a targeted amount of time. Tell them that they have 10 minutes to finish the mini-task. I do a quick check for completion at the end of class.
- Break large projects into small, manageable tasks. Do progress checks.
Best Practice #8: Teach the difference between collaboration & cheating

- Discuss at beginning of the course

- **Cheating**
  - Copy & paste with no understanding of the code
  - Borrowing to learn and submission as their own.
  - Code submitted and student cannot explain or modify.

- **Collaboration**
  - Discussing plan, ideas for a task
  - Asking for help when stuck with code the student created
  - Teaching another a new technique
  - Students accountable for understanding their code.
  - Students able to explain, recreate, modify or predict any code that they have created.
Best Practice #9: Be a Mover, not a Stopper

- **Movers**
  - DO NOT GIVE UP!
  - Problems create a learning opportunity and they find a way (google, partner, teacher).

- **Stoppers**
  - STOP
  - Get a challenge, give up
  - Whiners - tell me how
  - Recognize symptoms (heart rate, crying, anger, don't know, can't)
  - Set up strategies to overcome problem state (checklist of what to do)

https://pdfs.semanticscholar.org/6464/4ccec3db4e104e4d3ba2634563cea9cc145.pdf
Best Practice #10: Get out of your comfort zone. It will help you relate to your students!

When was the last time you did something for the first time?
Most of all...HAVE FUN! If you enjoy it, so will your students!