Recreating Movie Credits in Java: Introduction to Lists, Loops, and Syntax

Learning Objective

Students will be able to create a Java program that stores a list of movie credits in an array, iterate through the list using a loop, and print each credit to the screen with a time delay, demonstrating understanding of Java array syntax, for loops, and the Thread.sleep() method.

Assessments

Summative Assessment: Students will write a Java program that:

1. Declares and initializes a String array containing at least 5 movie credits (names or roles)
2. Uses a for loop to iterate through the array
3. Prints each credit to the screen with a 1-2 second delay between outputs
4. Compiles and runs without errors

Students will also complete a brief reflection: “Explain one key difference between how you created and printed this list in Python versus in Java.”

Key Points

- **Array declaration and initialization in Java:** Java requires explicit type declaration (e.g., `String[] credits = new String[5];`) and differs from Python’s dynamic list syntax
- **For loops in Java:** Java uses traditional C-style for loops with explicit initialization, condition, and increment (different from Python’s for-in loops)
- **Thread.sleep() for delays:** The `Thread.sleep(milliseconds)` method pauses execution; requires a try-catch block to handle InterruptedException
- **System.out.println():** Java’s standard output method replaces Python’s print()
- **Index-based access:** Like Python, Java arrays use zero-based indexing, but the syntax and declaration differ

Opening

- **Hook activity (5 minutes):** Display the JavaScript movie credits demo from the website they saw previously. Ask: “How would you recreate this in Java? What do you think would be different?”

- **Activate prior knowledge:** Ask students to brainstorm: “In Python, how did you store a list of items? How did you loop through it? How did you add a delay?” Record responses on the board.
- **Connect to today:** “Today, we’re going to do the exact same thing, but in Java. You’ll see that the ideas are the same, but the syntax—the rules for writing the code—is different.”

Introduction to New Material

- **Teach array declaration and initialization (10 minutes):**
  - Show side-by-side Python and Java syntax:
    - Python: `credits = ["Actor 1", "Actor 2"]`
    - Java: `String[] credits = {"Actor 1", "Actor 2"};`
  - Emphasize: Java requires the type (`String[]`), uses curly braces, and ends with a semicolon
  - Have students repeat the syntax aloud to build muscle memory
- **Teach the for loop structure (8 minutes):**
  - Write a Python for loop: `for name in credits:`
  - Contrast with Java: `for (int i = 0; i < credits.length; i++)`
  - Explain each part: initialization (`int i = 0`), condition (`i < credits.length`), increment (`i++`)
  - Show how to access array elements: `credits[i]`
- **Introduce System.out.println() and Thread.sleep() (8 minutes):**
  - Show: `System.out.println(credits[i]);` replaces Python’s `print(credits[i])`
  - Introduce the delay: `Thread.sleep(1000);` (1000 milliseconds = 1 second)
  - Explain that Thread.sleep() throws an exception, so it must be wrapped in try-catch
  - Show a simple try-catch block (do not require deep understanding of exceptions yet; focus on structure)
- **Live coding demonstration (5 minutes):**
  - Type out a complete working program on screen, narrating each step
  - Compile and run it; show the output with delays
  - Point out: “Notice how this works just like the JavaScript version, but the syntax is Java”
- **Common misconception to address:**
  - “I can declare an array like in Python without a type.” → Reinforce: “In Java, every variable must have a type. Arrays are no exception. `String[]` tells Java that this is an array of text.”
  - “I don’t need a semicolon at the end of statements.” → Show compiler error and correct it. “In Java, semicolons are required.”

Guided Practice

- **Behavioral expectations:** Students work on their own computers or in pairs (if equipment allows). Raise hand if stuck; teacher circulates to check progress.
- **Scaffolded practice with examples:**
    1. **Task 1 (Easy):** Given a partial program with a pre-filled array of 3 credits, students add System.out.println() statements inside the for loop to print each credit. (Focuses on loop syntax and output.)
    2. **Task 2 (Medium):** Students create their own array of 5 credits (e.g., movie title, director, producer, cinematographer, composer) and write the for loop to print them. (Focuses on array initialization and loop structure.)
    3. **Task 3 (Hard):** Students add a `Thread.sleep(1500);` call inside the loop, wrapped in a try-catch block. Teacher provides a template for the try-catch. (Focuses on handling exceptions and adding delays.)
- **Scaffolded questions to guide students:**
    - "What type of data are you storing? What should go in the square brackets?"
    - "How many items are in your array? What should the condition in the for loop be?"
    - "How do you access the first item? The second? The last?"
    - "What does `credits.length` give you?"
- **Monitoring student performance:**
    - Circulate and ask students to explain their code aloud: "Tell me what this for loop does."
    - Check for common syntax errors: missing semicolons, incorrect array initialization, off-by-one loop errors
    - Use a quick formative check: "Thumbs up if your code compiles, thumbs sideways if you have one error, thumbs down if you need help."

Independent Practice

- **Behavioral expectations:** Students work individually. They may reference their guided practice work and a provided syntax cheat sheet. No collaboration during this phase.
- **Assignment:** Students will create a complete Java program titled `MovieCredits.java` that:
    1. Declares and initializes a String array with at least 5 movie credits (they may use a real movie or invent one)
    2. Uses a for loop to iterate through the array
    3. Prints each credit to the screen with a 1-2 second delay between outputs
    4. Compiles and runs without errors

5. Includes a single-line comment explaining what the program does

- **Submission:** Students save their file and run it once to show the output. Teacher verifies compilation and execution.

Differentiation

Below Grade Level

- **Scaffolds:**
  - Provide a code template with blanks to fill in:

```
public class MovieCredits {
public static void main(String[] args) throws
InterruptedException {
String[] credits = {_____, _____, _____};
for (int i = 0; i < credits.length; i++) {
  System.out.println(_____);
  Thread.sleep(_____);
}
}
}
```

  - Provide a syntax reference card with examples of array declaration, for loops, and System.out.println()
  - Reduce the number of credits required to 3 instead of 5
  - Offer pre-written try-catch block; students do not need to write it themselves
  - Pair with a peer mentor during guided practice for verbal explanation of code

On Grade Level

- **Core expectation:** Students write the complete program from scratch using only the syntax rules taught in the lesson. They include proper array initialization, a correct for loop, System.out.println() statements, and Thread.sleep() with a try-catch block. The program compiles and runs without errors.

Above Grade Level

- **Extensions (increased cognitive demand):**
  - **Task 1 (Application):** Modify the program to accept user input for the delay time. “Ask the user: ‘How many seconds should the delay be?’ and use that value in Thread.sleep().” (Requires understanding of Scanner and input handling.)
  - **Task 2 (Analysis):** “Rewrite your program using a while loop instead of a for loop. Explain why you chose the initialization, condition, and increment

the way you did." (Requires understanding of loop equivalence and decision-making.)
    - **Task 3 (Synthesis):** "Create a second array for descriptions of each credit (e.g., an array of roles). Modify your program to print each credit alongside its description with a delay." (Requires managing multiple arrays and nested or paired output logic.)

Closing

- **Summary activity (5 minutes):**
    - Ask 3-4 students to share one line of code from their program and explain what it does
    - Ask the class: "What was the biggest difference between writing this in Python and Java?"
    - Recap: "You've now written a Java program with arrays and loops. These are building blocks for everything you'll do in AP Computer Science A."

Extension / Above-Grade Challenge

Challenge: Animated Credits Roll

Students will enhance their program to simulate a scrolling movie credits roll:

1. Add a loop that prints each credit multiple times, moving it across the screen using spaces (or ANSI escape codes for more advanced students)
2. Experiment with different delay times to create a smooth scrolling effect
3. Add color to the output (using ANSI color codes) for different types of credits (actors, directors, etc.)

Example (simple version):

```
for (int j = 0; j < 5; j++) {
  System.out.println(credits[i]);
  Thread.sleep(500);
}
```

This extension moves beyond the basic objective into creative application and exploration of output formatting.

Homework

Homework Assignment: "Adapt Your Program"

Students will modify their `MovieCredits.java` program to:

1. Add at least 2 more credits to their array (total of 7+)
2. Change the delay to a different value (e.g., 500 milliseconds or 3 seconds) and explain why they chose that value

3. Add a comment at the top of the code with the movie title and a brief description of what the program does

Reflection question (written, 3-5 sentences): ”Describe one challenge you faced when writing this Java program and how you solved it. How is this challenge different from (or similar to) challenges you faced in Python?”

Due: Next class period. Students will be prepared to share their modified program and reflection.

Standards Aligned

This lesson aligns with the **AP Computer Science A** framework, which emphasizes:

- Understanding and implementing fundamental programming constructs (variables, arrays, control structures)
- Writing syntactically correct Java code
- Understanding the relationship between code and its output

While specific AP CSA standard codes were not available through the standards database, this lesson directly supports the foundational competencies required for the AP Computer Science A exam, specifically:

- **Unit 1: Primitive Types** (variable declaration and initialization)
- **Unit 2: Using Objects** (understanding object-oriented syntax and method calls like System.out.println())
- **Unit 3: Boolean Expressions and if Statements** (preparation for conditional logic in loops)
- **Unit 4: Iteration** (for loops and loop control)