Class Examples

Lecture 6

📝 Ex 1: If Statements

- **Files**: Lecture6Examples.java (method: ifStatementExamples)
- **Topic**: Basic if statement for conditional execution.
- **Learning Goals**:
  - Understand the basic structure of an if statement.
  - Use comparison operators (>=, <, ==) to create conditions.
  - Write code that only runs when a specific condition is true.

🤔 Ex 2: Conditions

- **Files**: Lecture6Examples.java (method: conditionExamples)
- **Topic**: Exploring different types of comparison operators.
- **Learning Goals**:
  - Reinforce the use of various comparison operators (>=, !=, ==).
  - Use the .equals() method for accurate String comparison.
  - Understand that conditions evaluate to a boolean (true or false) value.

🖇️ Ex 3: Multiple Ifs

- **Files**: Lecture6Examples.java (method: multipleIfsExample)
- **Topic**: Using sequential, independent if statements.
- **Learning Goals**:
  - Understand that multiple, separate if statements are checked independently.
  - Recognize that more than one if block can execute if their conditions are all met.
  - Differentiate this structure from an if-else if chain.

⛓️ Ex 4: If-Else-If Chains

- **Files**: Lecture6Examples.java (method: ifElseIfElseExamples)
- **Topic**: Handling multiple, mutually exclusive conditions.
- **Learning Goals**:
  - Create if-else if-else structures to handle a series of distinct cases.
  - Understand that only one block in the entire chain will be executed.
  - Use a final else block to catch any cases not covered by the preceding if or else if conditions.

🔤 Ex 5: String Comparison

- **Files**: Lecture6Examples.java (method: stringComparisonExamples)
- **Topic**: Correctly comparing String objects.
- **Learning Goals**:
  - Use .equals() for case-sensitive String comparisons.
  - Use .equalsIgnoreCase() to compare two Strings while ignoring differences in capitalization.

- Understand why the == operator should not be used to compare String content.

🧠 Ex 6: Logical Operators

- **Files**: Lecture6Examples.java (method: logicalOperatorsExamples)
- **Topic**: Combining multiple boolean conditions.
- **Learning Goals**:
  - Use the && (AND) operator to require multiple conditions to be true.
  - Use the || (OR) operator to check if at least one condition is true.
  - Use the ! (NOT) operator to invert the value of a boolean condition.

🔀 Ex 7: Switch Statements

- **Files**: Lecture6Examples.java (method: switchStatementExamples)
- **Topic**: Efficiently handling multiple distinct values for a variable.
- **Learning Goals**:
  - Write a switch statement as a cleaner alternative to long if-else if chains.
  - Use the break statement to exit a switch block and prevent "fall-through".
  - Group multiple cases to execute the same code block.

🔢 Ex 8: Increment & Decrement

- **Files**: Lecture6Examples.java (method: incrementDecrementExamples)
- **Topic**: Shorthand operators for modifying numeric values.
- **Learning Goals**:
  - Understand the difference between post-increment (x++) and pre-increment (++x).
  - Use compound assignment operators (+=, -=) as a concise way to update variable values.

🔁 Ex 9: For Loops

- **Files**: Lecture6Examples.java (method: forLoopExamples)
- **Topic**: Iterating a specific number of times.
- **Learning Goals**:
  - Construct a for loop with its three parts: initialization, condition, and update.
  - Iterate through arrays and Strings by index.
  - Modify array elements within a loop.

✨ Ex 10: Foreach Loops

- **Files**: Lecture6Examples.java (method: foreachLoopExample)
- **Topic**: A simpler syntax for iterating over collections.
- **Learning Goals**:
  - Use the foreach (enhanced for) loop to easily access every element in an array without using an index.
  - Recognize when a foreach loop is more readable than a traditional for loop.

⏳ Ex 11: While Loops

- **Files**: Lecture6Examples.java (method: whileLoopExamples)

- **Topic**: Looping as long as a condition remains true.
- **Learning Goals**:
  - Write a while loop that continues until its condition becomes false.
  - Understand the importance of updating the loop control variable inside the loop to avoid infinite loops.

✅ Ex 12: Do-While Loops

- **Files**: Lecture6Examples.java (method: doWhileLoopExample)
- **Topic**: A loop that always executes at least once.
- **Learning Goals**:
  - Understand that a do-while loop checks its condition *after* the loop body executes.
  - Use a do-while loop for scenarios like input validation where an initial action is always required.

🛑 Ex 13: Break Statement

- **Files**: Lecture6Examples.java (method: breakExamples)
- **Topic**: Exiting a loop or switch statement prematurely.
- **Learning Goals**:
  - Use the break keyword to immediately terminate a loop's execution.
  - Apply break to stop iterating once a desired condition has been met.

🌀 Ex 14: Nested Loops

- **Files**: Lecture6Examples.java (method: nestedLoopExample)
- **Topic**: Placing one loop inside another.
- **Learning Goals**:
  - Understand how inner and outer loops interact.
  - Use nested loops to create patterns (like squares or triangles) and iterate over 2D structures.

🛠️ Ex 15: Helper Methods

- **Files**: Lecture6Examples.java (method: methodExamples)
- **Topic**: Using methods to organize and reuse code.
- **Learning Goals**:
  - Understand how to call a method and pass arguments to it.
  - Analyze methods that perform specific tasks like iterating through a String or modifying an array.
  - See an example of method chaining (.toLowerCase().equals()).