Practical Exercise - Walkthrough

When trying to run first time, you'll notice that the workflow fails with "Object reference not set to an instance of an object." message when executing the Assign.

-
    - This is because the *index* variable is not initialized. While a **Int32** or **String** get created with default values even if they are not explicitly specified, a **GenericValue** variable is not created implicitly as there is no way to know if you'd like to have an integer (so a 0 default value), a String (so a "" default value) etc.
    - Either change the variable type to an Int32, or set a Default value of 0. Let's go with the latter.
- If you ran it again, you would notice that the first issue is solved, but now it is failing after a few iterations, with the message "Index was outside the bounds of the array.".
    - This is because at that moment, the index variable value is 5, so it's trying to access the filesArray index / position number 5, which doesn't exist in this case. The array has 5 elements, starting from index 0 to index 4. The index 5 is outside of the array limits.
    - This is due to a logical error in the algorithm as the loop is iterating more times than needed.
    - We should only let the index get to a value of maximum 4 when it enters the *Sequence*, so the easiest way is to remove the = from the expression, leaving it as *index < filesArray.Count.*
- If we check the logs, we can see a contradiction between the messages for file 5.txt, one stating that the content will not be added and other mentioning that the append has been finalised.
- Also, the requirements state that if the reading of the file fails (as is the case for 5), the **Append** line should not be executed.
- Third, if we look at the consolidated file, the content is wrongly added, as some lines are duplicated.
    - This is because the text variable has a **While** scope, so it's being kept from iteration to iteration. That means when it is not set with the file text value, because of the **Read Text File** exception (e.g. 5.txt), then it keeps the previous value (e.g. the 4.txt text).

    - As we can easily fix this by changing the scope of text from **While** to **Process** one file, this fixes just part of the problem and still doesn't respect the requirement.

- The easiest way to fix the functionality is by putting the content of both **Try Catch** blocks into only one.

    - This way, if the first activity fails, the execution jumps to the exception handling **Catch** block, so the second activity is not executed anymore.

    - Drag and drop the **Append Line** and **Log Message** activities to the first **Try** block, right after the **Read Text File**. The **Append Line** depends on **Read Text File** so they need to be in the same **Try Catch** block to work correctly.

    - Delete the second **Try Catch** block.