Retraining Scheduler – Design Explanation & Instructions

Design Choices

The Retraining Scheduler uses a backtracking algorithm to optimally assign training talks into morning and afternoon sessions, this ensures optimal utilization of available time, the code also has fall back Greedy algorithm (`sorts talks in descending order of duration`) this is faster but not optimal.

Each track contains a morning session (9:00 AM – 12:00 PM), a lunch break (12:00 PM), an afternoon session (1:00 PM – no later than 5:00 PM),
and ends with a mandatory sharing session starting after 4:00 PM but no later than 5:00 PM.

Backtracking ensures maximum time utilization within each session. Talks are never repeated across tracks. A helper method calculates
session start times, ensuring the final schedule is formatted correctly with AM/PM times.

Assumptions

- Talk titles are assumed to be free of numbers and validated accordingly.
- The keyword "lightning" is interpreted as a 5-minute session.
- Lunch is always fixed at 12:00 PM, regardless of the actual end time of the morning session.
- Each talk can only appear once across all tracks.
- The scheduling respects the maximum time allowed per session (180 minutes for morning, 240 for afternoon).

Instructions to Run the Application

1. Ensure you have .NET 8.0 SDK or later installed.
2. Run the project

Visual Studio:

1. Open the RetrainingSchedular solution.
2. Set RetrainingSchedular as the startup project.
3. Press F5 (Debug) or Ctrl + F5 (No Debug).

Command Line:

1. Open a terminal in the project root directory.
2. Run: ***dotnet run --project RetrainingSchedular***

3. You will be prompted to choose between default talks or entering your own.
   - Type '***1***' to use the default list of talks.
   - Type '***2***' to manually enter talks in the format: \`Title, Duration\`.

By: Eduke Ohien Stephen

- You may use the word '***lightning***' instead of a numeric duration to represent a 5-minute talk.
- Type 'done' when finished.

4. The application will output the complete schedule grouped by track with start times and durations.

Running Unit Tests

The project includes unit tests using the xUnit framework. There are two test files

1. HelpersTest.cs *# Unit tests for Helpers*
2. SchedularTest.cs *# Unit tests for Schedular logic*

To run the tests:

***visual Studio* -** open the project in visual studio and run the test project

CLI* -** Execute the following command from the solution root: ***dotnet test

By: Eduke Ohien Stephen