Assignment 2

COMPSCI 3318 - Software Engineering in Industry

TrekCon - Event Scheduling

Luke and Ben from **Basement Engineering Entertainment** are riding the success of their wildly popular mobile game:

Trek Wars

They are preparing for their first ever "TrekCon" - an 16-day event filled with game demonstrations, panel discussions, merchandise stalls and much more. They have recently just finalised the events that will make up the convention, but are yet to determine exactly when the convention will be held.

They have asked you to implement a scheduling application that can help them understand and validate their event schedule.

Input Data

You are provided with two pieces of input data on the command line:

- A path to a CSV containing information on each event on arg[1]
- A start date in ISO date format (2024-08-30) on arg[2]

The CSV of event data contains a single row per event, consisting of the following columns:

1. Event ID
2. Event name
3. Event day (start from 1)
4. Start time (HH:mm, 24-hour time)
5. Duration (HH:mm)

Scheduling Rules

The scheduling of events must follow these rules:

1. No two events shall overlap.
2. If the opening day falls on a Saturday or Sunday, the convention shall be rescheduled to start on the following Monday.
3. If the final day falls on a Saturday or Sunday - it shall be moved to the following Monday.
4. All events should be run on their scheduled day, unless the event falls on a Saturday or Sunday - in which case, that event must be rescheduled to a weekday.
5. Events must be scheduled to start and finish between 9am and 5pm (local time).
6. There must be a 1-hour lunch break during the day - starting no earlier than 12 noon, and finishing no later than 2pm.

Output Format

Your application should exit with a return code of zero and output to standard output in CSV format, with the following columns:

1. Event ID
2. Event name
3. Event date (yyyy-MM-DD)
4. Start time (HH:mm, 24-hour time)

These will be cross referenced against the input data to ensure it meets the scheduling rules.

Output Validation

To validate your output, submit your executable to Maptek Titan. Titan is an automated testing platform that will execute your application against a number of tests.

https://titan.maptek.net/

You will each be provided a login, and can submit to Titan as many times as you wish. To upload your entry and any dependencies, you can submit a ZIP with the name based on your entry, i.e <*your_application_name*>.exe.zip. The executable must be available in the top level directory once it has been unpacked.

A number of different tests will be provided, some public and some hidden.

- **Public** tests will show you the output of the first failed test (if any).
- **Private** tests will not show any errors from failed tests, and will instead report the number of tests passed.

These tests will be used to determine the effectiveness of your application, and as such a portion of your marks will be derived from the Titan tests.

MyUni Submission

Please provide all documentation in a single PDF. Each section of the PDF should be clearly marked with the corresponding deliverable number and title.

All code should be placed into a **zip** and submitted through MyUni. It is recommended you do all work in a single solution, making use of individual projects for each API and each scheduling application implementation.

Please ensure projects are sensibly named (CalendarV2 is an acceptable library name for the API as part of Deliverable 3).

Resources

Luke and Ben have provided you with a Visual Studio solution containing their own C# “Calendar” library. This solution uses dotnet 8 (required: dotnet 8 SDK). The documentation for the Calendar library can be found using the Object Browser inside Visual Studio, or by opening the XML file in the Calendar folder of the provided zip.

A copy of the events CSV has already been included, but you can view a web copy below. To change the arguments passed at runtime, edit the “launchSettings.json” file in the Properties folder.

Link to library solution: EventScheduler.zip
Link to data: TrekCon events

Deliverable 1

Luke and Ben have asked you to implement your scheduling application using the functions provided in their Calendar library. No changes to the library are permitted at this time, but any supporting functions are welcome.

Once your application is ready, submit your executable to the Titan testing service which will validate your scheduling output for correctness.

Grading: 8 marks

Deliverable 2

In delivering the application to *Luke & Ben*, you raised with them that the Calendar API was difficult to work with. Provide a ½ - 1 page description of the issues you’ve identified with their API.

Grading: 2 marks

Deliverable 2.1

As part of your feedback, provide an API consisting of class names and method stubs to better solve this problem (1 - 2 pages). Provide justification for your changes, and why they represent good design principles.

Grading: 3 marks

Deliverable 3

Implement your new API - using good configuration management strategies.

Adjust your original scheduling implementation to use your new library. Submit the resulting application to Titan to validate the results.

Grading: 5 marks

Deliverable 3.1

Provide a half page reflection on how the improved API:

- Represents good Software Engineering practices
- Improved the development of the scheduling application
  - i.e fast to develop, less error prone
- Will be easier for consumption by other developers in comparison to the original API

Grading: 2 marks