CS 2400 Daily Agenda (60 points)

Due: Wednesday April 23, 2025 (11:59 PM) **NO LATE SUBMISSION**

Modify the Appointment class designed in project 6 by overloading the "==" operator. The operator should compare two appointments and return true if the first appointment has the same title as the second appointment. The operator should treat uppercase and lowercase letters equally.

Write a C++ program to manage a single day agenda using the *Appointment* class from project 6. Appointments for the day are stored in a file called "**agenda.txt**". The appointment data is stored in the file using the following format:

```
"title|year|month|day|time(standard)|duration"
```

Example:

```
" Meeting with Bob | 2019 |4 |29 |8:30 aM  | 15"
```

The file might contain empty lines or no lines at all (no appointments). See the supplied sample file in the assignment's repository.

Write a C++ program that uses the Appointment class to manage the daily calendar. Your program should start by reading all the appointments from the data file and process command line arguments as follows:

- `./a.out -ps`
  - Print my daily schedule in **order** by starting time using standard time format. The appointments must be displayed in a table format with proper labels.

- `./a.out -p "time"`
  - Print all appointments at specified military time.

- `./a.out -a "Appointment data"`
  - Add an appointment given in the format: `"title|year|month|day|time|duration".` Time is given in standard time and duration is given in minutes. Leading and trailing spaces must be removed.
    Example: `" Meeting with Sue | 2019 |4 |29 |8:30 aM  |15 "`

- `./a.out -dt "title"`
  - Delete all appointments that match a title exactly (with leading and trailing spaces removed). It should treat uppercase and lowercase characters the same.

- `./a.out -dm "time"`
  - Delete a all appointments that match the starting military time exactly.

CS 2400 Daily Agenda (60 points)

Due: Wednesday April 23, 2025 (11:59 PM) **NO LATE SUBMISSION**

If the daily calendar data is updated (add, delete) then all the data must be stored back in the same data file using the same format. You should be able to run the program again and see the changes reflected in the new daily agenda.

Error checking:

- Your program must handle data files that include empty lines or no lines at all.
- Report an error if the number of command line arguments is invalid
- Report an error if an option is not valid

Grading:

Programs that contain syntax errors will earn zero points.
Programs that do not include functions will also earn zero points.
Programs that use global variables other than constants will earn zero points.

You may use any function or library discussed in class or in the chapters we covered from your textbook. Do not use any other libraries or functions.

Your grade will be determine using the following criteria:

- Correctness (55 points)
  - (5 points) overloaded "<" operator
  - (10 points) `-ps` option
  - (5 points) `-p` option
  - (5 points) `-a` option
  - (10 points) `-dt` option
  - (10 points) `-dm` option
  - (5 points) Data file is updated
  - (5 points) error checking

- Style & Documentation (5 points)

Follow the coding style outline on GitHub:
https://github.com/nasseef/cs/blob/master/docs/coding-style.md

Submit a link to your repository on Canvas.

Please note, you will not be able to submit your link after the due date.