CISPROG-2 Final Project - Erich Finkle - Spring 2025

Project Title:
Teacher Simulator

Author:
Erich Finkle - Programmer

Description:
This project is a C++ text-based game where the player fills the role of a high school teacher, choosing actions each day that affect their morale, fatigue, and reputation. To mix things up, random events can occur, which also affect fatigue, morale, and reputation. The game ends if morale or reputation reaches zero, causing the teacher to quit or be fired, respectively.

Creation process:
This project was built using OnlineGDB's hosted C++ development environment and compiler. It's a C++ text-based game where the player fills the role of a high school teacher, choosing actions each day that affect their morale, fatigue, and reputation.

To mix things up, random events can occur, which also affect fatigue, morale, and reputation. The game ends if morale or reputation reaches zero, causing the teacher to quit or be fired, respectively.

This project started off as a big mess, to be honest. I had my class definitions and everything all crammed into main.cpp just to get the ideas out of my head, and it wasn't until about 6 hours of work into the project that I separated things into the 3 files that are used now.

Files:
The game is composed of three files:

- main.cpp (The core game loop and user name input)
- Teacher.h (Header file containing all data for the Teacher)
- Event.h (Header file containing all data for the Events)

Development Notes:
Since I'm still pretty new to coding, it took me a long time to figure out how to get some things to work while others went surprisingly smoothly. I had to consult with Christian a couple times to figure out where I was getting weird errors that I couldn't figure out, like the first time with an error in my switch cases causing the Day/AM/PM cycle to not loop properly. I later found out this was entirely due to me forgetting how switch worked, and using break and continue incorrectly, because of course it was. It didn't help that I've also been teaching Python to my high school students this year, and my brain was struggling to differentiate on more than one occasion.

That aside, on the whole the project went pretty quickly and I'm happy with how it turned out. It's not particularly complex, I suppose, but it's just complex enough for me to be proud of it. It also marks the last project for completing my AS in CIS Programming, so... hooray!

Known issues:
As of now, the only problem I couldn't figure out how to fix was how to handle intentionally messed up input for the Day/AM/PM cycle. If the user enters a hybrid input that starts with a number, like **3asd**, the program reads the number, pulls the correct action, then automatically says they've made an invalid input for the next action. I tried a few things with getline, but wasn't able to figure it out. So, this still exists, but as long as the user enters a number that's expected, it works without a hitch. I wasn't able to cause any crashes in my tests either, so I'm happy about that.

Conclusion and Future Ideas:
I could have made things more complex or incorporated some other modifiers to the game, like having to track individual students' progress, dealing with other factors to the job, or earning a paycheck every month and budgeting my spending (so as not to go broke on my pitiful teacher salary!), but these felt more like they went beyond the scope of what I was going for in my project and thus I decided not to include them.

If I were to modify this project in the future, I'd want to work with someone who knew about UI design to have a lot more features incorporated into the game. For example, I could have the teacher actually walk around the campus during downtime, take attendance, attend meetings, or track student grades to make sure their students aren't all failing to encourage wise choices about how they run the class. Those other features, though, I feel would be a bit overbearing with plain text output since it would just be a lot of printing on the screen.

With a codeveloper capable of making an actual graphic UI, I could focus on coding the core of the game like I have here, and let them handle the visual interpretations, which I might enjoy, but for now, I've been satisfied with my work.

Cheers to everyone for a great semester, and best of luck in your future classes!