Teacher Simulator

Erich Finkle

CISPROG-2 Final Project

Chaffey College, Spring 2025

Introduction

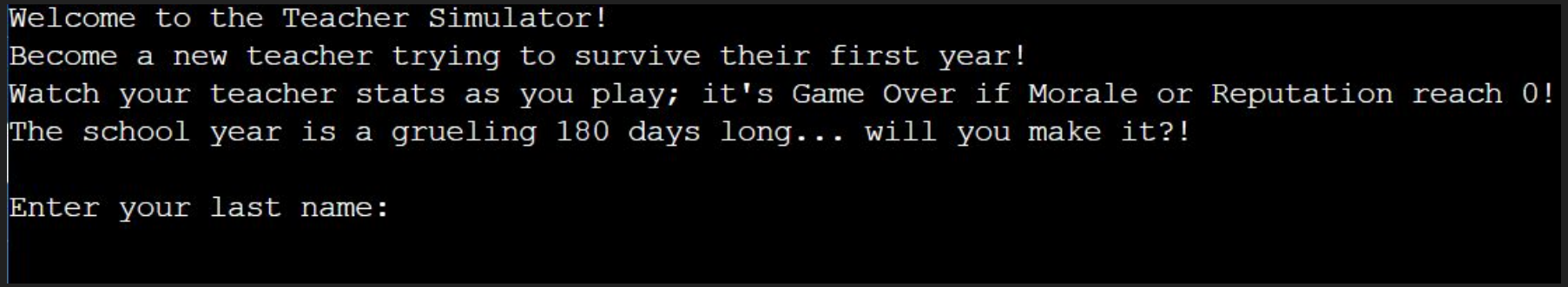
Hello, I’m Erich Finkle, a high school English and Computer Science teacher, and esports coach! I created this project as a simple game that used some of the concepts we learned in CISPROG-2 this year to simulate my everyday life as an educator.

Project Implementation

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.

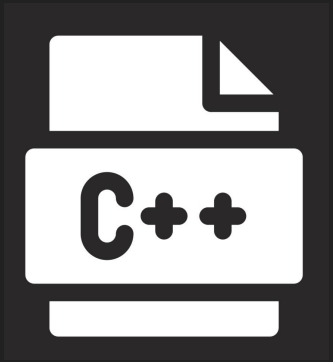
To mix things up, random events can occur, which also affect fatigue, morale, and reputation. The game ends if morale or reputation reaches zero, causing the teacher to quit or be fired, respectively.



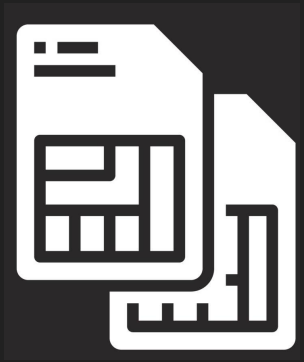
Project Implementation (cont.)

This project started off as a big mess, to be honest.

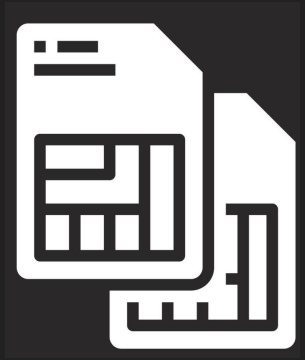
I had my class definitions and everything all crammed into main.cpp just to get the ideas out of my head, and it wasn't until about 6 hours of work into the project that I separated things into the 3 files that are used now.



main.cpp
Main game loop and user name entry.



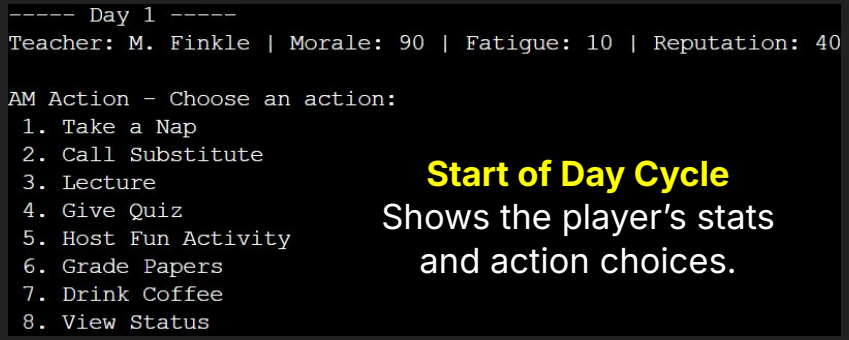
Teacher.h
Data and methods for the Teacher class.



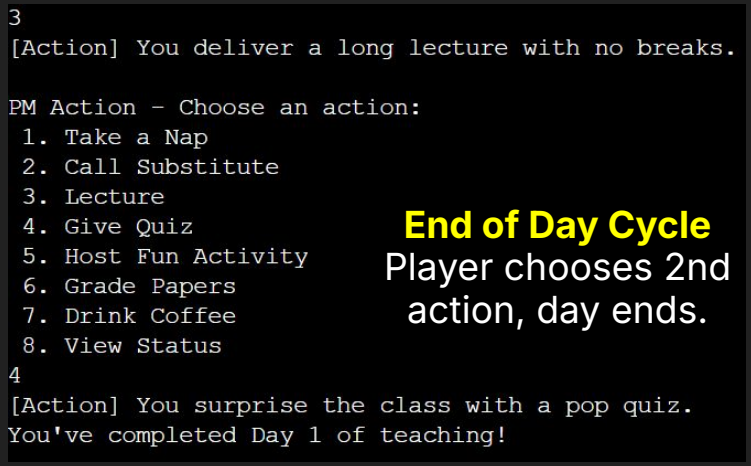
Event.h
All data related to random game events.

Project Demo

Here are some screenshots of the program at work.



Start of Day Cycle
Shows the player's stats and action choices.



End of Day Cycle
Player chooses 2nd action, day ends.

Project Demo (cont.)

Here on Day 2, we see a **random event** happen.

These can be **positive** or **negative**, affecting the teacher's Morale, Fatigue, and Reputation in different ways.

```
----- Day 2 -----
Teacher: M. Finkle | Morale: 77 | Fatigue: 30 | Reputation: 50

AM Action - Choose an action:
 1. Take a Nap
 2. Call Substitute
 3. Lecture
 4. Give Quiz
 5. Host Fun Activity
 6. Grade Papers
 7. Drink Coffee
 8. View Status
5
[Action] You lead a fun class activity.

PM Action - Choose an action:
 1. Take a Nap
 2. Call Substitute
 3. Lecture
 4. Give Quiz
 5. Host Fun Activity
 6. Grade Papers
 7. Drink Coffee
 8. View Status
6
[Action] You stay late grading assignments.
[Event] A student turned in their essay — but it's all ChatGPT.
You've completed Day 2 of teaching!
```

Project Demo (cont.)

The Events list is somewhat robust, with a total of about **25 events** that can happen.

Note the structure of each Event:

- Description
- Morale change
- Fatigue change
- Reputation change

```
// EVENT RANDOMIZER
Event getRandomEvent() {
    // Initializes the event pool. Static make ssure it only happens one time.
    static vector<Event> eventPool = {
      // Positive events that benefit the teacher.
      Event("Some free time lets everyone relax.", 10, -10, 0),
      Event("The principal praises your teaching.", 15, 0, 5),
      Event("A student thanks you after class.", 10, 0, 5),
      Event("A student aces a tough test — they thank you for preparing them.", 10, 0, 5),
      Event("The school provides free coffee in the staff lounge.", 5, -5, 0),
      Event("Your class spontaneously cleans up the room before the bell.", 8, -5, 5),
      Event("A parent sends a kind thank-you note.", 12, 0, 10),
      Event("Your lesson plan goes perfectly — every student was engaged.", 15, -5, 10),
      Event("A student actually read the syllabus. Miracles do happen.", 10, -5, 5),
      Event("Admin gave you a document camera that *almost* works. Truly blessed.", 5, -2, 3),
      Event("Your lesson only got interrupted by one announcement. New record!", 8, -3, 4),
      Event("You graded all your papers before midnight! You're tired, but happy.", 12, -8, 6),
      Event("Nobody asked, 'Can I use the restroom?' today. Are you dreaming?", 15, -5, 10),

      // Negative events that hurt the teacher.
      Event("A safety drill disrupts your lesson.", -10, 20, 0),
      Event("You get a harsh email from an angry parent.", -15, 0, -10),
      Event("Half your class didn't do the homework.", -12, 7, -8),
      Event("You forgot to submit attendance and got a warning.", -8, 0, -8),
      Event("A student crashes out during your lesson.", -12, 6, -6),
      Event("You spill coffee on your new Quiz — it's ruined.", -8, 6, 0),
      Event("An internet outage ruins your digital lesson.", -10, 0, -7),
      Event("A fight breaks out in the hallway — your class gets disrupted.", -12, 15, 0),
      Event("A student brought their emotional support lizard. It bit someone.", -15, 8, -15),
      Event("The copier jammed again. You challenged it to a duel.", -10, 10, -6),
      Event("A fire alarm test occurs. During your test. Again.", -12, 12, -6),
      Event("A student turned in their essay — but it's all ChatGPT.", -12, 5, -6)
    };
```

Project Demo (cont.)

The “teacher” is really just a class.

Data members represent stats.

Three **methods** affect those stats.

```
class Teacher {
private:
  string name;
  int morale;      // 0-100, game over if 0
  int fatigue;     // 0-100, forced to rest if 100
  int reputation;  // 0-100, fired if 0
public:
  Teacher(string n = "Teacher", int m = 90, int f = 20, int r = 40)
    : name(n), morale(m), fatigue(f), reputation(r) {}

  // STAT CHANGE METHODS
  void changeMorale(int change) {
    morale += change;
    if (morale > 100) morale = 100;
    if (morale < 0) morale = 0;
  }

  void changeFatigue(int change) {
    fatigue += change;
    if (fatigue > 100) fatigue = 100;
    if (fatigue < 0) fatigue = 0;
  }

  void changeReputation(int change) {
    reputation += change;
    if (reputation > 100) reputation = 100;
    if (reputation < 0) reputation = 0;
  }
```

Project Demo (cont.)

The teacher's Actions are handled by more methods named after the corresponding action.

The **callSubstitute** method in particular requires separate handling when called in the AM or PM parts of the day.

```
// TEACHER ACTIONS
void takeNap() {
  cout << "[Action] You sneak in a quick nap behind your desk.\n";
  changeFatigue(-8);
  changeMorale(2);
  changeReputation(-5);
} // Bad for reputation, good for mental health.

void callSubstitute(bool isMorning) {
  cout << "[Action] You call in a substitute";
  if (isMorning) {
    cout << " for the whole day.\n";
    changeFatigue(-40);
    changeMorale(5);
    changeReputation(-15);
  } // When called in the AM
  else {
    cout << " for the afternoon.\n";
    changeFatigue(-20);
    changeMorale(3);
    changeReputation(-8);
  } // When called in the PM
} // Great for health, bad for reputation.

void lecture() {
  cout << "[Action] You deliver a long lecture with no breaks.\n";
  changeFatigue(18);
  changeMorale(-7);
  changeReputation(6);
} // Tiresome, but good for reputation.
```

Project Demo (cont.)

Lastly, some **getter methods** and a **display method** finish the class.

A few other methods are used simply to return a boolean to determine whether the teacher **calls a sub due to exhaustion**, **quits due to low morale**, or **is fired due to bad reputation**.

```
// GETTER FUNCTIONS
string getName() const { return name; }
int getMorale() const { return morale; }
int getFatigue() const { return fatigue; }
int getReputation() const { return reputation; }

// DISPLAY STATS METHOD
void display() const {
  cout << "Teacher: " << name
       << " | Morale: " << morale
       << " | Fatigue: " << fatigue
       << " | Reputation: " << reputation
       << endl;
}

// STATUS CHECKS
// Checks to see if the teacher must call a sub, is fired, or has quit.
// 100 Fatigue = Burnout, 0 Reputation = Fired, 0 Morale = Quit.
bool isBurnedOut() const { return fatigue >= 100; }
bool isFired() const { return reputation <= 0; }
bool hasQuit() const { return morale <= 0; }
```