Gebze Technical University
Department of Computer Engineering
CSE 241/501
Object Oriented Programming / Programming
Fall 2024
Homework # 2
Due date Jan 2 2025

ANSI Terminal-Based Spreadsheet Program - Enhanced Version

Project Overview

This project extends the previous terminal-based spreadsheet program with additional requirements to demonstrate modern C++ programming skills. The focus is on dynamic memory management, smart pointers, exception handling, namespaces, templates, and a robust class hierarchy. Other core requirements remain the same as the original project.

New Project Requirements

1. **Dynamic Memory and Smart Pointers**
  - Avoid STL containers for spreadsheet data storage.
  - Implement your own 2D dynamically allocated array to manage spreadsheet data.
  - Use smart pointers for memory management.
2. **Abstract Cell Class**
  - Abstract Base Class: `Cell` with pure virtual functions.
  - Derived Classes:
    - `FormulaCell`: Represents cells containing formulas (e.g., =A1 + B2).
    - `ValueCell`: A base class for specific value types:
      - `IntValueCell`: Integer values.
      - `StringValueCell`: String values.
      - `DoubleValueCell`: Floating-point values.
3. **Namespaces**
  - Define logical namespaces to organize code, such as `Spreadsheet` and `Utils`.
4. **Exception Handling**
  - Use C++ exceptions for handling errors like invalid formulas, out-of-bound references, and file operation failures.
5. **Templates**

  - Use templates for reusable components, such as a `Range` class or `Grid` template.
6. **Formula Parsing**
  - Support operators (+, -, *, /) and functions (SUM, AVER, STDDEV, MAX, MIN).
7. **File Operations**
  - Save and load spreadsheet data in CSV format.
8. **Visual Interface**
  - Maintain the ANSI terminal interface similar to VisiCalc.

Submission Requirements

- Source Code: All source files, including the provided AnsiTerminal.h and AnsiTerminal.cpp files.
- Include a header file and a CPP file for each class.
- Documentation PDF:
  - UML Diagram of the class structure.
  - Description of implemented features and any missing parts.
  - Declaration of AI assistance, if applicable.
  - A User Manual explaining the usage of the program with examples.
- Do not use any functions from the standard C library (like **`printf`**), do not use C arrays. For math functions you may use standard C functions.
- Use all the OOP techniques that we have learned in the lectures, C++11 features
- Do not forget to indent your code and provide meaningful comments.
- We will provide a number of CSV files to test your program
- You should submit your work to the Teams page using the instructions from the TAs.
- You will demo your homework online