CSE3150: Homework 3 – Arrays, Pointers, Const, and Exceptions

Overview

In this assignment, you will write a C++ program that manages a simple list of students and their GPAs. You will use arrays, pointers, references, const correctness, type casting, control flow, and exceptions. This assignment will also introduce the `cstring` library for working with C-style strings. The function `std::strcpy(dest, src)` copies a C-string from `src` into `dest`, including the terminating null character `\0`. The function `strlen(str)` returns the length of a C-string (not including the null terminator).

New Concepts

nullptr

`nullptr` is a special keyword in C++ that means ”this pointer points to nothing.” It is the safe replacement for the old `NULL`. You can initialize pointers with `nullptr` when they do not yet point to anything. Example:

```
double* gpaPtr = nullptr;
if (gpaPtr != nullptr) {
    *gpaPtr = 3.7;
}
```

Exceptions with Strings

Exceptions in C++ can be any type. In this assignment, you will throw exceptions as strings and catch them with `catch(const char* msg)`.

```
try {
    double avg = averageGPA(gpas, size);
} catch (const char* msg) {
    std::cout << msg << std::endl;
}
```

Command Line Arguments

Your program will accept the student list capacity from the command line. C++ provides `argc` (argument count) and `argv` (argument values). `argv` is an array of C-style strings, so you must convert the string to an integer with `std::stoi`.

```
int main(int argc, char* argv[]) {
    if (argc < 2) {
        std::cout << "Usage: ./program <capacity>" << std::endl;
        return 1;
    }
    int capacity = std::stoi(argv[1]);
    // now use capacity instead of hardcoded value
}
```

Specifications

1. Store student names and GPAs in two parallel arrays of size `capacity`.
2. Implement the following functions:
   (a) `void addStudent(char* name, double gpa, char* names[], double gpas[], int& size, int capacity);`
   (b) `void updateGPA(double* gpaPtr, double newGpa);`
   (c) `void printStudent(const char* name, const double& gpa);`
   (d) `double averageGPA(const double gpas[], int size);`
3. In `main()`, implement a menu with `switch` and `if/else`. Always use braces for control flow statements.
4. Use `static_cast<int>` when printing the average GPA to show the rounded integer value.
5. Catch exceptions for adding when full or computing average with no students, and print the error string.

Deliverables

- `starter_main.cpp`: starter file with TODOs.
- `solution.cpp`: complete solution.
- `README.md`: short explanation of how you used pointers, const, references, casting, exceptions, and control flow.
- Push to your GitHub under cse3150_week_3_hw and make sure the tests pass