OOP Course and Laboratory
Catalin Boja
catalin.boja@ie.ase.ro

Assignment 4

Implement a C++ class Book that uses dynamic memory (int*), constructors, destructor, copy constructor, static members, and safe getters/setters. No operator overloading is required.

Notes for the Assignment

- **You can use only pointers and classic C array manipulation** (**NO** STL containers like vector).
- **You can use only iostream, string.h and stdio.h libraries.**
- **Don't change given methods definition**
- **NO shallow copy**
- For char array processing you can use ONLY *string.h* functions (like strncpy, strlen, strcmp, etc). Because is vulnerable to different buffer overflow attacks, **we don't use** strcpy(). Use strcpy_s().
- The solution must be modularized by using functions.
- Manage memory carefully, especially when dealing with dynamic arrays.
- Solutions that use advanced C/C++ syntax will not be considered for evaluation (like dynamic cast, other libraries, etc)
- Required time to code it, from 80 minutes to 240 minutes
- Please don't waste your time by coding this with ChatGPT or other code generation tools. You gain nothing. Try to code it, even if you don't know how to solve everything or don't have time to finish it. Coding is learned by writing a lot of bad code. Copy & paste does not help. Use Google Search and other tools to learn how to do on your own different implementations.

Requirements:

1. All attributes are private and are accessible by public accessor methods
2. Implement:

Attributes:

- static integer nextId; – counts how many Book objects have been created.
- integer id; – unique id of the book, automatically assigned based on nextId. No setter.
- std::string title; – title of the book.
- int* chapterPageCounts; – dynamic array of chars; number of pages for each chapter.
- Integer chapterCount; – number of chapters.

Constructors

1. **Default constructor**
   - Sets title to "Unknown".
   - Sets chapterPageCounts to nullptr.
   - Sets chapterCount to 0.

    - Assigns a new unique id based on nextId.
2. **Parameterized constructor**
3. Book(const std::string& title, const int* chapterPageCounts, int chapterCount);
    - Sets the book title.
    - Allocates dynamic memory for chapterPageCounts and copies all chapter pages.
    - Sets chapterCount.
    - Assigns a new unique id.
4. **Copy constructor**
5. Book(const Book& other);
    - Performs a **deep copy** of other, including its dynamic array.
    - Does **not** share the same chapterPageCounts pointer.
6. **Destructor**
7. ~Book();
    - Releases dynamically allocated memory for chapterPageCounts.

Getters

- int getId() const; – returns the unique id.
- std::string getTitle() const;
- int getChapterCount() const;
- int getChapterPages(int index) const;
    - Returns the number of pages of chapter at index (0-based).
    - If index is invalid, return -1.
- int getTotalPages() const;
    - Returns the sum of all chapterPageCounts.
- double getAverageChapterPages() const;
    - Returns the average number of pages per chapter (0 if chapterCount == 0).

Setters

- void setTitle(const std::string& title);
- void setChapterPageCounts(const int* chapterPageCounts, int chapterCount);
    - Must:
        - Delete old dynamic array (if any).
        - Allocate new memory and copy the values (deep copy).
        - Update chapterCount.

Processing Methods

- bool hasShortChapter(int maxPages) const;
    - Returns true if there is at least one chapter with pages <= maxPages.
- int getIndexOfFirstChapterWithPages(int pages) const;
    - Returns index of the first chapter that has exactly pages pages, or -1 if none.

Static Method

- static int getNumberOfBooksCreated();
    - Returns the value of nextId.

Private Helper

- void copyChapters(const int* chapterPageCounts, int chapterCount);
    - Allocates and copies the dynamic array. Used by constructors and setter.

Extra processing methods

- bool hasShortChapter(int maxPages) const;
- int getIndexOfFirstChapterWithPages(int pages) const;
- bool isLongerThan(const Book& other) const;

- int getNumberOfChaptersWithinInterval(int minPages, int maxPages) const;
- bool canBeReadInDays(int days, int maxPagesPerDay) const;