Object Oriented Programming Lab

Lab 08 **Marks 10**

Instructions

- Work in this lab individually.
- You can use your books, notes, handouts etc. but you are not allowed to borrow anything from your peer student.
- Make sure to follow the best coding practices.
- Include comments to explain the logic where necessary.
- *You are strictly **NOT ALLOWED** to include any additional data-members/functions/constructors in your class.*
- Test your program thoroughly with various inputs to ensure proper functionality and error handling.
- Show your work to the instructor before leaving the lab to get some or full credit.

ADT: GenericArray

Write a generic class named **GenericArray** capable of managing arrays of any valid data type (e.g., int, float, char). The class should provide functionality for creating, manipulating, and comparing arrays, ensuring robust error handling and memory management.

1. The class should have the following two private data members.

- A generic pointer called **data** that holds an array of given data type allocated dynamically according to the specified size.
- An integer called **size** represents the number of elements in the array.

2. Implement the following constructors and a destructor:

- A default constructor that allocates an array of size 5 and initializes it to the so-called "empty array," i.e., an array whose array representation contains all zeroes.
- A constructor that accepts an integer as argument to represent the size of an array and initializes it to the so-called "empty array," i.e., an array whose array representation contains all zeroes.
- A copy constructor initializes an array object with an already existing object.
- A destructor to free any memory resources occupied by the array object.

3. Implement the following non-static member functions and operators:

- **getSize()**: Returns the size of array.
- **setElement(int index, T value)**: Inserts a new element at the specified index in the array. If the index is out of bounds, display an appropriate error message.
- **countElement(T key)**: Accepts a key and counts the total occurrences of it in the array.
- **operator=**: Copies the data of the right-hand-side object to the left-hand-side object. If the sizes are different, reallocate memory for the left-hand-side object based on the size of the right-hand-side object.
- **getSubArray(int start_index, int end_index)**: Returns a new array containing values from start_index to end_index, inclusive. If the requested sub-collection cannot be created, the function should throw an exception with an appropriate error message.
- **operator+**: Appends the contents of the right-hand-side array to the end of the left-hand-side array, returning the result.
- **operator<<**: Displays the contents of the array.
- **operator>>**: Allows input of data into the array.
- **operator==**: Determines whether two arrays are equal or not. The operator should return *true* if both arrays (left-hand-side and right-hand-side) are equal, and *false* otherwise.
- **Subscript ([])** for non-const objects. If the index is out of bounds, throw an exception with an appropriate error message.
- **Subscript ([])** for const objects. If the index is out of bounds, throw an exception with an appropriate error message.

4. Test the functionality of the **GenericArray** class by creating objects of different data types and performing various operations on them.