Object Oriented Programming Lab

Lab 08 **Marks 10**

Instructions

Work on this homework individually. Absolutely NO collaboration is allowed. Any traces of plagiarism would result in ZERO marks in this homework and possible disciplinary action. Task should be coded in **C++**. You are strictly ***NOT ALLOWED*** to include any additional data-members/functions/constructors in your class. Write the ***main*** function first and keep testing the functionality of each function once created.

Due Date

Upload the solution *(source code **.cpp** file only)* labeled with your complete roll number in capital letters e.g., BITF21M000 till 05:00PM Wednesday, April 12, 2023, in course's *Google* classroom.

ADT: Collection

Write a class named **Collection** that can hold a group of **negative integers** and **zero** as the *default value*.

1. This class should have two private data members:
   a. An *integer pointer* called **data** that holds an array of integers allocated dynamically based on the specified size.
   b. An *integer* called **size** represents the number of elements in the array.

2. Implement the following **constructors** and a **destructor** for the **Collection** class:
   a. A constructor that accepts an integer argument to represent the **size** of the array and initializes it as an *empty collection*, meaning all elements in the array are set to *zero*.
   b. An additional constructor that receives an **array of integers** and its **size** as arguments and uses the array to initialize a collection object.
   c. A copy constructor that initializes a collection object with an existing object.
   d. A destructor that frees any memory resources occupied by the collection object.

3. Implement the following **non-static member functions** and **operators** for the **Collection** class:
   - **getSize()**: Returns the size of the collection.
   - **setElement(int i, int k)**: Inserts a new integer **k** at index **i** in the collection, if possible. Otherwise, give an appropriate error message.
   - **countElement(int key)**: Accepts an integer **key** as an argument and returns the total occurrences of it in the collection.
   - **Assignment operator (=)**: Copies the data of the right-hand-side object to the left-hand-side object. If the size of the left-hand-side object is different from the right-hand-side object, reallocate the memory for the left-hand-side object based on the size of the right-hand-side object and then copy the data. Don't forget to update the size of the left-hand-side object.
   - **getSubCollection(int start_index, int end_index)**: This member function takes two integer parameters **start_index** and **end_index** as arguments and returns a new Collection that contains all the values in the left-hand-side object from **start_index** to **end_index**, both inclusive. If the requested sub-collection cannot be created, it displays an appropriate error message and returns a Collection object consisting of NULL with its size set to 0.
   - **Stream insertion operator (<<)**: Allows the user to input data for the collection.
   - **Stream extraction operator (>>)**: Displays the contents of the collection on the screen.
   - **Addition operator (+)**: Performs the addition of two collections (left-hand-side and right-hand-side) and returns the result. If the two collections have different sizes, it displays an appropriate error message and returns a Collection object consisting of NULL with its size set to 0. Otherwise, the function adds the corresponding elements of both collections and stores the result in a new Collection object, which it returns.
   - **Subscript ([])** for non-const objects. If the **index** is out of bounds, it displays an appropriate error message and returns 99, Without exiting the program.
   - **Subscript ([])** for const objects. If the **index** is out of bounds, it displays an appropriate error message and returns 99, Without exiting the program.
   - **Unary minus operator (–)**: Returns *true* if all the elements of the collection are negative or zero, *false* otherwise.

4. Once you have written the class, write the **main** function and test its functionality by creating some objects of **Collection**.

☺ ☺ ☺ **BEST OF LUCK** ☺ ☺ ☺