Data Structures and Algorithms Lab

Lab 02 **Marks 05**

Instructions

Work on this lab individually. You can use your books, notes, handouts etc. but you are not allowed to borrow anything from your peer student.

Marking Criteria

Show your work to the instructor before leaving the lab to get some or full credit.

What you have to do

Program the following task in your C++ compiler and then compile and execute them. *Write the **main** function first and keep on testing the functionality of each function once created.*

ADT: Collection

Write a class named **Collection** for which each object can hold **positive numbers** and **zero** as a default value.

1. The class should have following **two private data members**.
    1. An **integer pointer** named **data** that holds an **array of integers** allocated dynamically according to the **size.**
    2. An **integer** named **size** that holds the **size of the array** *(amount of memory allocated to data)*.
2. Provide the implementation of following **constructors** and a **destructor**
    1. A **constructor** which accepts an **integer** as argument to represent the **size of an array** and initializes it to the so-called "empty collection," i.e., a collection whose array representation **contain all zeroes**.
    2. An additional **constructor** that receives an **array of integers** and the **size of that array** as its arguments and uses the array to initialize a **collection object**.
    3. A **copy constructor** to initialize a collection object with already existing object.
    4. A **destructor** to **free any memory resources** occupied by the **collection** object.
3. Provide the implementation of following member functions and operators
    1. **getSize** returns the size of collection.
    2. **printCollection** to display the contents of **data** on the screen of a **collection**.
    3. **getCollection** to take input from user for the **data** of a **collection** as per its **size**.
    4. **insertElement** that **inserts** a new integer **k** at index **i** *(both passed as argument)* into a **collection**, if possible, otherwise give an appropriate error message.
    5. **findElement** accepts an integer **key** as argument and return **true**, if the key element exists in the collection, **false** otherwise if the key does not exist.
    6. **Assignment (=) operator** which copies the data of right-hand side object to the left-hand side object. If the size of left-hand side object is different from right-hand side object. Free any previous memory resources occupied by left-hand side object and reallocate the memory to it according to the size of right-hand side object. Eventually copy the data and return the result as per the standard implementation of assignment operator. Don't forget to update the size of left-hand side object.
    7. **Addition (+) binary operator** which perform the addition of two collections *(right-hand side from left-hand side)* and **return** a new *Collection* containing the result of addition. This addition should only be performed, if both objects have same size, give appropriate error message otherwise and return a new *Collection* with **size** and **data** set to **-1** and **NULL** respectively.
    8. **Comparison (==) operator** that determines whether two collections *(left-hand and right-hand)* are **equal** or **not**. The operator should return **true** if both the collections are equal, **false** otherwise.

In ***main*** function, create few objects of ***Collection*** class and demonstrate the working of each function clearly.

☺ ☺ ☺ **BEST OF LUCK** ☺ ☺ ☺