CSC3002 (fall 2022) Assignment 3

Problem 1 [2 pts]

Exercise 11.6:

Using the following definitions of **MAX_JUDGES** and **scores** as a starting point.

const int MAX_JUDGES = 100;
double scores[MAX_JUDGES];

Write a program that reads in gymnastics scores between 0 and 10 from a set of judges and then computes the average of the scores after eliminating both the highest and lowest scores from consideration. Your program should accept input values until the maximum number of judges is reached or the user enters a blank line. A sample run of this program might look like this:

```
9.0
9.1
9.3
9.0
8.8
9.0
```

and you will get a result indicates:

```
The average after eliminating 8.80 and 9.30 is 9.03.
```

Requirments:

Please finish **sumArray**, **findLargest** and **findSmallest** functions in the file *Gymnastics-Judge.cpp*. DO NOT modify the main() part, which is for testing unit.

Problem 2 [3 pts]

(a) Exercise 12.4:

Design and implement a class called **IntArray** that implements the following methods:

- A constructor **IntArray(n)** that creates an **IntArray** object with n elements, each of which is initialized to 0.
- A destructor that frees any heap storage allocated by the **IntArray**.
- A method **size()** that returns the number of elements in the **IntArray**.
- A method **get(k)** that returns the element at position **k**. If **k** is outside the vector bounds, **get** should call **error** with an appropriate message.
- A method **put(k, value)** that assigns **value** to the element at position **k**. As with **get**, the **put** method should call **error** if **k** is out of bounds.

Requirments:

Please finish **constructor**, **destructor**, **size**, **get** and **put** functions in the file *intarray.cpp*. **Use int[ ] instead of wrapping an exiting std::vector<int>**; DO NOT modify the main() part, which is for testing unit.

(b) Exercise 12.5:

You can make the **IntArray** class from the preceding exercise look a little more like traditional arrays by overriding the bracket-selection operator, which has the following prototype:

int & operator[](int k);

Like the **get** and **put** methods, your implementation of **operator[]** should check to make sure that the index **k** is valid. If it is, the **operator[]** method should return the element by reference so that clients can assign a new value to a selection expression.

Requirments:

Please finish **operator[]** function in the file *intarray.cpp* according to the file *intarray.h*

(c) Exercise 12.6:

Implement deep copying for the IntArray class from Problem 2(a) and (b).

Requirments:

Please finish copy constructor and assignment operator in the file *intarray.cpp*. DO NOT modify the main() part, which is for testing unit.

Requirements for Assignment

You should finish all *.cpp* files according to the problem requirements. DO NOT modify the main() part, which is for testing unit.

Please note that, the teaching assistant may ask you to explain the meaning of your program, to ensure that the codes are indeed written by yourself. Please also note that we may check whether your program is too similar to your fellow students' code using BB.

Please refer to the BB system for the assignment deadline. For each day of late submission, you will obtain late penalty in the assignment marks.

Reminder: DO NOT wait until the last minute to finish your work.