Optional Programming Makeup

*Can ONLY be used to address the “programming assignment” subminimum; cannot change the numeric grade*

Due Sunday 10 December

In brief: Create a class `PosIntSequence` that stores a sequence of nonnegative integers. It should provide the functions listed below. It should use a fixed array of ints for storage; but on deletion, it should mark the cell as invalid rather than moving elements.

The idea is that the values are always stored in the order they are presented: a value can only be added to the end of the sequence.

The class should be stored in the files `PosIntSequence.h` and `PosIntSequence.cpp` and should exhibit good C++ style.

The member variables to be used in `PosIntSequence` should include the following:

- `int capacity`: stores the current length of the array.
- `int *array`: stores the data. All data initialized to UNUSED

The functions should include:

```
constructor and destructor
void append(int element);
void remove(int element);
void increaseCapacity(int moreCapacity);
int getSize();
stream insertion operator<<
void rawDump();
```

There should also be constants

```
static const int INVALID = -1;
static const int UNUSED = -2;
```

The functions do the following:

- The unique constructor takes an integer argument that sets the value of the capacity. (All entries should be set to UNUSED.)
- append adds a value to the end of the present sequence in the first unused space. It should print an error message and do nothing if there is no room, or if the value to be added is negative.
• remove removes the first (from the beginning) occurrence of the specified value by changing the entry to INVALID. It should print an error message and do nothing if the value is absent.

• getSize returns the number of valid data values in the present sequence.

• increaseCapacity increases the length of the array by the amount specified (which should be positive). It should at the same time, compact the data to the first getSize positions.

• The stream insertion operator should print out the data sequence in a nice format.

• rawDump prints out the full contents of the array and other member variables (needed for grading) with a line-feed.

For example, starting empty, the sequence will end as 4-2-3-1 on the following.

append(4); append(1); append(2); append(3); remove(1); append(1);

1. This is work to be done individually. For what it is worth, you may use my notes and my website. You may seek (minor) assistance from me, but NOT the TAs, tutor, etc.

2. Your program MUST provide all specified functions. If unable to get a particular method to work, then it should simply print a message to the screen saying so and return. A program that does not compile will not be graded.

3. Your program MUST be robust. A program that crashes when it shouldn’t or gives wrong answers without comment will be more penalized than one that runs but skips certain cases.

To be turned in via handin.