Department of Computer Science
Graduate Exam in Programming
Fall 2021

You must write **Java** or **C++** on the first page of your answers!! You may answer the questions using either Java or C++, but you must use the same language for **each** question on the exam.

1. (**2 pts**) Write a function named `validate` that takes an array of integers and a size as an argument. The function should return true if ALL of the numbers are between 0 and 100 (inclusive). If ANY of the numbers in the array is less than 0 or greater than 100, the function should return false.

   For example, for the array containing {0,3,8,100} with size 4 it should return true. For the array containing {0,99,101,77,87} with size 5 it should return false (because 101 is greater than 100).

The next question uses the following class, which implements a linked list:

```
// C++
class List {
private:
  struct Node {
      string value;
      Node *next;
  };
  //points to the first node
  Node *head;

public:
    List() {head=NULL;}
    void removeLast();
  };
```

```
// Java
class List {
  private class Node {
      String value;
      Node next;
  }

  //refers to the first node
  private Node head;

  public List() {head = null;}
  public void removeLast() {…}
};
```

2. (**2 pts**) Define the public member function `removeLast()` that removes the *last* element from the linked list. If the list is empty, the function should do nothing (it should not cause an error, it should not output an error message).

3. (**2 pts**) Write a **recursive** function named countV that takes three arguments: an array of integers and its size and an integer v, and returns a count of the number of times v appears in the list.
   For example, if a contains {9,7,3,4,7,1} then countV(a,6,7) returns 2.
   Do not use loops, extra parameters, or global or static variables.

4. (**4 pts**) Declare and implement a class to represent a list of integers, called IntList. Use an **array of integers** of size 100 to store the values in the list. Use an integer variable named **num** to store the number of elements currently in the list.

   Include the following functions in your class:

   - a no-argument constructor that sets up an empty list.
   - a void function add(x) that adds a new value, x, to the end of the list. If adding the element would exceed the list's capacity, this function should output an error message.
   - a function sort(a,size) that takes an array of integers (a) and its size and rearranges the items in the array so that they are in ascending order. You may use any algorithm to sort the array. It should **not** change the order of the elements in the list member.
   - A function median() that returns the median value of the list of integers. If the values are in order from smallest to largest, the median is the middle value. If the list contains an even number of values, the median is the average of the two middle values. If the list is empty, the median is 0. This function should not change the order of the elements in the list member. For example, if the list contains{3,5,8,4,9,6,7} the median is 6. If the list contains {6,7,8,9,1,2,3,4} the median is 5. If the list contains {7,8,9,10,1,2,3,4} the median is 5.5.