CSCI 2170 Test 1 review on array

- One dimensional array
  - Read data into array and record the number of items in the array
  - Use array subscript to access array elements
  - Traversal in an array
  - Linear search in an array
  - Sorting in array
  - Insert and delete items into/from array
  - Pass 1D and 2D array to functions

- Two dimensional array
  - Use nested for loop with two dimensional array
  - be able to perform operations (i.e., computing the average) along one column of the 2 dimensional array, or along one row of the 2 dimensional array
  - be able to perform operations (i.e., search for the smallest value) across all the elements (i.e., across all the rows and columns) of the 2 dimensional array
  - look over the class notes and practice questions at the end of the class notes

Some Example test questions:

1. Write a complete C++ program to:
   a. Read float type values from a data file named "values.dat" into a one-dimensional array named "values" (the maximum capacity of the array is set to 100). The values are stored one value per line in the data file. There are an unknown number of values in the data file. Your program reads the values til it reaches the end of the data file or the maximum capacity of the array is reached.

   After the values are read and stored in the array:

   b. Sort the values in ascending order. Write a **Sort function** for this task. Call the function in the main function.
   c. Find and return the array subscript corresponding to the minimum value in this array. Write a **FindMinIndex** function for this task. Call the function in the main function and display that array subscript.

   For (b) and (c), you need to show the complete function declaration, activation and definitions

2. Add a user defined function **InsertAtFront** to Question 1 such that a user may add a new value to those currently stored in an array of float type values.
   - If the array is already full, display a message to the user stating that insertion can not be carried out.
   - If the array still has room for more values, then prompt the user to enter a new value, and store that value **at the beginning of the array, i.e., at array location having subscript 0. This means all the values that were in the array need to be shifted to make room for this new value.**

Show the complete C++ function definition for the function **InsertAtFront**.

3. Given two **parallel** one dimensional arrays in the main function as shown below: **airportCode** and **airportName**. **airportCode** contains the codes for major airports, for example, BNA is the code for the Nashville International Airport and ATL is the code for the Atlanta Airport; and **airportName** contains the names of the airports. The total number of airport codes/names is stored in variable "**numOfAirports**".

airportCode	**airportName**
BNA	Nashville Airport

ATL	Atlanta Airport
SFO	San Francisco Airport
SJC	San Jose Airport
JFK	J F Kennedy New York Airport
ORD	Chicago O'Hare Airport
LAX	Los Angeles Airport
…	…
…	…
DCA	Washington DC – National Airport

Write a C++ **user defined function** named **FindAirportName** that prompts the user to enter an airport code, and displays the corresponding airport name. The input parameters for this function include: the array **airportCode**, the array **airportNames**, and **the number of items stored in these arrays**

4. This problem has two parts. (The second part is on the back page)
The function **Insert** does not work quite the way it is supposed to.

a. Step through the code shown below and write down the output of the current version of the program.

```
#include <iostream>
using namespace std;

int Insert(int [], int, int, int);
const int SIZE = 10;   // maximum number of items to store in array

int main()
{

        int array[SIZE], value=0, position=1, aSize=4;

        for (int i=0; i<aSize; i++)   // initialize array
            array[i] = 2*i+1;

        // insert value into array at position
        aSize = Insert(array, value, position, aSize);

        // display values after the insertion
        for (int i=0; i<aSize; i++)
            cout << array[i] << " ";

        return 0;
}

// this function inserts "element" in the given "position"  in array "arr". It returns the new array size
int Insert(int arr[], int element, int position, int size)
{
        for (int i=position; i<size; i++)
            arr[i+1] = arr[i];
        arr[position]= element;
        size=size+1;
```

Show program output here:

```
    return size;
}
```

b. **How would you modify the function <u>Insert</u> to correctly insert an element into the array at position "position"?** For example, before insertion, array looks like this: 5 7 9 2. After the insertion of element 6 at position 2, array looks like this: 5 7 6 9 2

- A logging operation keeps records of 37 loggers' monthly production for purposes of analysis, using the following array structure:

```
const int NUM_LOGGERS = 37;
cons tint NUM_MONTH=12;
int logsCut[NUM_LOGGERS][NUM_MONTH];
```

For the following problems you may assume that the number of logs cut by each logger in each month is already read into the array "logsCut".

(a) Show the C++ statements to compute the yearly total for logger number 8

(b) Show the C++ statements to find the best logger (most logs cut) in May.

(c) Show the C++ statements to compute the total number of logs cut by all the loggers in the whole year.