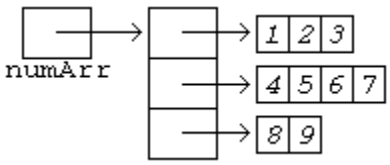
ENGI4892 – Data Structures
Assignment 1
Due: May 30, 2017 (at 1PM)

Guidelines:

- All code is to be properly formatted, indented and commented, and where appropriate, pre/post-conditions must be provided.
- For code submissions, include .h and .cpp files; for non-code submissions, include either a text document, or in the case of drawings, etc, a picture of the item. No paper copies are to be submitted.
- See the link "Submission Details for All Assignments" on the course webpage for more information.

1. (75 marks) A *ragged array* is one where the number of elements in each row are not necessarily the same (ie, there are a set number of rows, but the number of columns in each row is variable). For example, the following is a ragged array of integers with 3 rows contains 3,4,2 columns, respectively:



- For this assignment, you will be creating an array of `struct` containing the following information:
  - A pointer to an array of integers (unknown size initially)
  - An integer representing the number of elements in the array (size)
  - An integer representing the minimum value in the array (to be calculated)
  - An integer representing the maximum value in the array (to be calculated)
  - A double representing the average value in the array (to be calculated)

  In other words, your row array will contain, at each index, a `struct` as defined above.

- You will have a data file in the following format:

```
4
3 10 20 15
5 56 34 45 21 45
4 45 67 24 56
1 59
```

  The number on the first line indicates the number of rows, and in each row thereafter the first number indicates the number of columns in the *ragged array* (ie, the size of the array). For the example above, there are 4 rows, containing a `struct` with internal array of integer of size 3,5,4,1 columns, respectively.

Write a C++ program that will read from a given data file (in the format above) and create a *ragged array* of `struct` (also defined above). Integers should be read from each row of the file and stored inside the `struct`, in a dynamically allocated array of integers. Also store the number of elements in the `size` variable inside the struct.

Write three functions with signatures as follows (note that the return type and parameter list are left out; you must figure them out based on the pre/post-conditions listed):

```
// pre-condition: Pointer to an array of 'n' integers
// post-condition: Calculates the minimum element in the array (array is unchanged)
… minimum( … )

// pre-condition: Pointer to an array of 'n' integers
// post-condition: Calculates the maximum element in the array (array is unchanged)
… maximum ( … )

// pre-condition: Pointer to an array of 'n' integers
// post-condition: Calculates the average of the elements in the array (array is unchanged)
… average ( … )
```

After reading all data into the *ragged array*, traverse the array, and for each struct calculate the `minimum`, `maximum` and `average` (using the functions above), and store these inside the `struct` in the appropriate variables.

Finally, traverse the *ragged array* again, and print out its entire contents (integer data inside the array, size of the array, and the minimum, maximum and average values of the array).

2. (25 marks) Time Complexity theory:

   a) Show that $(n+1)^5$ is $O(n^5)$.
   b) Show that 2^{n+1} is $O(2^n)$.
   c) Show that n is O(nlogn).
   d) Show that $\sum_{i=1}^{n} i^2$ is $O(n^3)$.
   e) Show that $7n^2 + 5n + 3$ is not O(n).