Homework 2 - Playing with Numbers

Jee Whan Choi & Chris Misa

April 12, 2019

DUE DATE: 11:59 PM 4/19/2019

The objective of today's homework is to understand how pointer and arrays work. The homework has two parts

- Given a list of integers, find the n^{th} largest integer in the list.
- Given a matrix of numbers stored as a 2-D array, find the n^{th} largest integer in each *column*.

Follow the instructions below to get started:

- In your class repo, create the directory structure `homeworks/hw02/`. The rest of this assignment should be completed in the `hw02` directory which is where the grader will look for your final solution.
- Once the tarball has been extracted, you should see three files: this pdf, `hw02_nth_largest_skeleton.c`, and `hw02_nth_largest_matrix_skeleton.c`.
- Compile the code using `gcc hw02_nth_largest_skeleton.c` which will generate the `a.out` executable.
- Study the code to see what is happening. This is **important** since you will use some or all of these functions to implement the homework.
- Do the same for `hw02_nth_largest_matrix_skeleton.c`.
- As before, rename your files to `hw02_nth_largest_answer.c` and `hw02_nth_largest_matrix_answer.c` before committing them to the repository.

hw02_nth_largest_skeleton.c This program will take in one integer as input - n - and you will print the nth largest integer in a given list of integers. This list will be loaded from a file, whose name is #defined at the top of the program. In this program, we have already taken care of input processing - if you enter no value or more than one value, it returns an error. If you've entered just one value, the main function will call the *find_nth* function, which you will implement. We have also taken care of reading the input file for the list of intgers. This function is *load_data* and it will return an array of integers - *int_array* - and its size - *array_size*.

Things to note:

- The *find_nth* function should return the index into the array for the nth largest value. If n is larger than the number of integers in the list, it should return -1, and it should return -2 for all other errors.
- For any outcome that is **not 1) the nth largest number** or **2) n that is larger than the array size**, the program should print "Some error!" That is, only output of this program should be 1) "The nth value is <answer>", 2) "n is too large" and 3) "Some error!" For example, there is no such thing as 0th largest element, so an entry of 0 for n should print "Some error!" This excludes code written by the instructors (e.g., error messages from the functions *arg_test* and *load_data*)
- All answer and error messages should be printed between `---- Answer ----` and `--------`. Copy the strings from the skeleton code to get the exact string values.
- The index should be for the **first occurrence** of the nth largest value.
- For this part of the homework, you are allowed to use functions provided in the standard C library.
- The grading will be done with a different set of integers so make sure your program can handle lists of arbitrary size.
- There are two places that you need to insert your code - `// Inser your code here (1)` and `// Inser your code here (2)`. In the first one, you should write code to figure out what n is from the input (i.e., argv), and make sure the input is valid (i.e., an integer >= 1), and in the second one, you should write the code to implement *find_nth*. You need not worry about printing the answer - the code for checking the return value of *find_nth* has already been provided.

hw02_nth_largest_matrix_skeleton.c As before, this program will take in one integer as input - n - and you will print the n^{th} largest integer in each column of a 2D array of integers. This 2D array will be loaded from a file, whose name is #defined at the top of the program and will always be **square** (i.e., number of rows is equal to the number of columns). We have already taken care of input processing - if you enter no value or more than one value, it returns an error. If you've entered just one value, the main function will call the *find_nth* function, which you will implement. We have also taken care of reading the input file for the 2D array of intgers. This function is *load_data* and it will return a 2D array of integers - *int_array* - and its sizes - *row* and *col*.

- The *find_nth* function should return an array of numbers (via *ret_array*), whose size is equal to the number of columns in the 2D array. Each element in the array should be the n^{th} largest integer in each column. For example, the element in *ret_array*[0] should be the n^{th} largest value in column 0 of the 2D array.

- If n is larger than the number of rows in the 2D array, it should return -1 (for every element of *ret_array*). It should return -2 for any other error. -2 should appear for columns that had an error, but may not necessarily have -2 on every element of *ret_array*. For example, if column 1 had an error, *ret_array[1]* should have -2, but if you were able to find the n^{th} largest value for column 3, *ret_array[3]* should have that value.

- For this part of the homework, you are **NOT** allowed to use functions provided in the standard C library, or any other libraries, including third party ones.

- You are also **NOT** allowed to create new arrays or data structures to temporarily store any part of the 2D array. That is, all processing must be done in-place. You should also **not** use *ret_array* as a temporary storage.

- The grading will be done with a different sets of integers so make sure your program can handle 2D arrays of arbitrary size (but still square).

- There are two places that you need to insert your code - `// Inser your code here (1)` and `// Inser your code here (2)`. In the first one, you should write code to figure out what n is from the input (i.e., argv), and make sure the input is valid (i.e., an integer >= 1)

and print "Some error!" if it is not valid, and and in the second one, you should implement the *find_nth* function.

- If the function is implemented properly, it will return an array with the correct values (whether it is the n^{th} largest value, -1, or -2) that will be printed by the provided skeleton code.

- Note that in the first exercise, you returned the **index** into the array for the n^{th} largest value. Here, you are returning the n^{th} largest value itself (and not the index) in *ret_array*.

- For this program, the only ouptut should be 1) content of *ret_array* (printed by the provided skeleton code) and 2) "Some error!" All answer and error messages should be printed between `---- Answer ----` and `--------`. Copy the strings from the skeleton code to get the exact string values. This excludes the code provided by the instructors as before.

Note that you may choose not to include the extra information printed by the skeleton code in your final solution. These functions were provided just to help you get started in the right (or wrong) direction.

When you have completed your assignment, verify it on ix-dev and then rename the file to `hw02_nth_largest_answer.c` and `hw02_nth_largest_matrix_answer.c`, then add, commit, and push the file to your Bitbucket repository.

Extra credit will be given if you can find clever ways to modularize your code (i.e., instead of going through your entire 2D array to find the answer, send each column or row of the 2D array to a function to find the n^{th} largest value one column or row at a time).

Have fun with your assignment and don't hesitate to post questions on Piazza if something is ambiguous.