Eric Sehun Oh
CIS 22B

Homework 3 Report: ***Pointers and Dynamic Allocation of Memory***

Data Organization

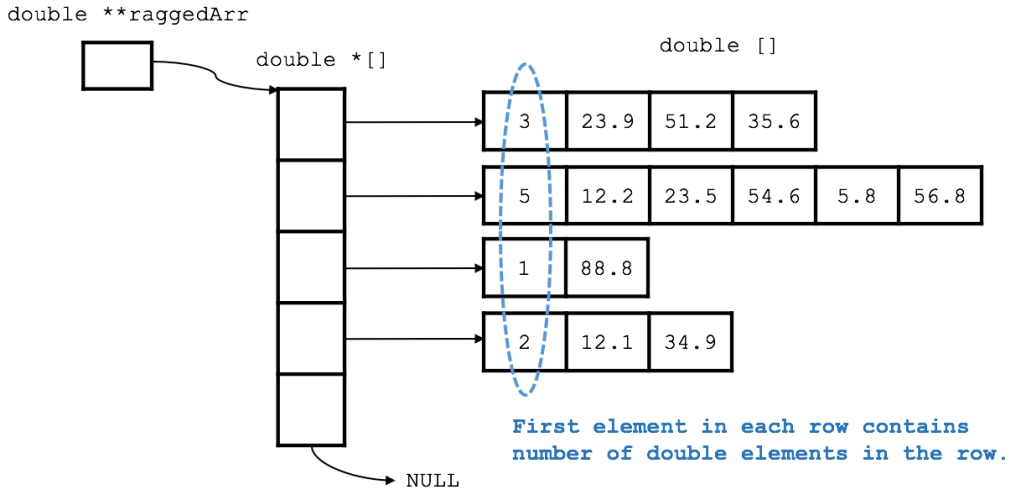

figure refers to example ragged.txt file for sake of demonstration

`double **raggedArr` is a pointer to an array of double pointers. Each double pointer in the double pointer array points to a double array filled with values, except the last pointer which is set to `NULL` in order to be used as a stop condition in other functions. Each double array is formatted as follows: first element in the array denoting the number of double values, each consequent element being a double value that needs to be sorted.

Structure & Functions

```
void insertionSortRow(double *row);
     // Sorts the row in descending order
void insertionSortArray(double **arr);
     // Sorts the ragged array from longest length to shortest length
void writeRaggedArray(double **arr);
     // Prints ragged array to screen with proper formatting
void freeArray(double **arr);
     // Frees all dynamically allocated arrays
```

// Pseudocode
// Read in first line and store number into `int` variable `num_rows`
// Set `raggedArr` to point to dynamically allocated double pointer array with `num_rows+1` elements,
// Read in first element of line and store into `int` variable `num_elems`
// Dynamically allocate double array with `num_elems+1` elements, store that variable as the first element and store the next `num_elem` elements
// Call function `insertionSortRow` on the row
// After `num_rows` rows have been read in call `insertionSortArray` on the `raggedArr`
// Write the output by calling the function `writeRaggedArray`
// Free dynamically allocated memory by calling the function `freeArray`