Summer 2013 - CS 2050
Pre-lab 3

Objectives:

- Implement selection sort on an array of structures
- Use malloc and free

Definitions needed at the top of your program (**Bold only**):

```
#include <stdlib.h>
#include <ctype.h>
#include <time.h>
#include <string.h>

#define INIT_SIZE 8

typedef struct Array_ {

     int data;

}DynamicArray;

typedef struct ArrayInfo_ {

     unsigned int max_size;
     unsigned int curr_idx;

}DynamicArrayInfo;


DynamicArrayInfo info;
// NO OTHER GLOBAL ARRAYS OR VARIABLES ALLOWED
```

Inside main:
     Will assign max_size to INIT_SIZE and assign curr_idx to zero. Next call allocate on a locally created pointer of type DynamicArray. After successful allocation, populate the newly allocated array with randomly generated integer values from 1 to 100 upto max_size from info. Note: make sure to update the curr_idx inside of info after each new addition to the allocated array. Then call sort, and then destroy the allocated memory. At the end display infos curr_idx and max_size they should be the same.

void void allocate(DynamicArray ** v);
     Will allocate memory using malloc to assign to v using max_size from info.

void destroy(DynamicArray **v);
     Will free the allocated memory assigned to v.

int sort(DynamicArray *v, unsigned int size);

Will sort the v in descending order using a **selection sort using pointer notation.**

Sample Output

```
$ ./a.out
Final Stats from info [last index: 8 current max size: 8]

$ valgrind --tool=memcheck ./a.out
```

Cool tool for checking for memory leaks. Make sure to run this after the program runs fine. It'll catch possible location of memory leaks.

Output Information from Valgrind

```
==29643== Memcheck, a memory error detector
==29643== Copyright (C) 2002-2012, and GNU GPL'd, by Julian Seward et al.
==29643== Using Valgrind-3.8.1 and LibVEX; rerun with -h for copyright info
==29643== Command: ./a.out 8
==29643==
Final Stats from info [last index: 8 current max size: 8]
==29643==
==29643== HEAP SUMMARY:
==29643==     in use at exit: 0 bytes in 0 blocks
==29643==   total heap usage: 1 allocs, 1 frees, 32 bytes allocated
==29643==
==29643== All heap blocks were freed -- no leaks are possible
==29643==
==29643== For counts of detected and suppressed errors, rerun with: -v
==29643== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 6 from 6)
```