CIS 26B
Advanced C
Programming Assignments

Homework 1

100 Points

Arrays, Strings, Structures, Sorting, Pointers, and Dynamic Allocation of Memory

Project: Create and process arrays of structures

Write a program which expects the name of an input files and an output file to be given by the user. If the user does not input any names, default file names should be used, such as `in26B.txt`, `in22C.txt`, and `out.txt`. There are two input files: one contains students enrolled in CIS 26B and the other one students enrolled in CIS 22C. The input files have lines which look like this:

```
  1234 Marley, Tom
```

The number represents the student ID and it is followed by the student name. Read data from the first input file into a dynamically allocated array of STUDENT structures. Then read data from the second input file into a second dynamically array of STUDENT structures. You may assume that the maximum size of a name string is 31. The program should use either the insertion sort algorithm or the selection sort algorithm to sort the arrays in ascending order by student ID. To demonstrate that the sorting algorithm works, display the sorted arrays to the screen. Create a third array with students taking both classes. This third array should also be a dynamically allocated array sorted in ascending order. Finally, write the third array to the output file (using the same format as the input files' format). Create your own input files using the data shown on the next page. On the first line in the input file provide the number of ID/name lines. Make sure that your program does not produce memory leaks. Memory leak detection is optional (see last page). Run the program once and save the output at the end of the source file as a comment. Compress the source files, input and output files and upload the compressed file: 26B_LastName_FirstName_H1.zip

Grading

1. Read file names	– 10
2. Reading from file	– 20
3. Sorting	– 20
4. Display sorted arrays	– 10
5. Create the third array	– 20
6. Write to file	– 20

NOTE: Write a comment in the beginning of the program. Write a comment for each function. Write comments inside functions (if needed). Use proper indentation and spacing. Do not use global variables. Do not use the goto statement. Always check if opening an input file was a successful operation. Do the same for dynamic allocation of memory.

in26B.txt

```
18
1234 Marley, Tom
9002 Khuller, Samira
8372 Chen, Li
3456 Karlin, Anna
2908 Vigoda, Eric
6566 Williams, Ryan
8999 Fenner, Mia
8433 Chakrabarti, Amit
8879 Bein, Wolfgang
9865 Beame, Paul
2901 Green, Mary
1189 Shmoys, David
5445 Homer, Steve
6579 Vadhan, Salil
9123 Vianu, Victor
5567 Welch, Jennifer
6766 Hemaspaandra, Lane
4344 Kelley, Sandra
```

in22C.txt

```
15
3456 Karlin, Anna
2000 Barenboim, Leonid
6666 Forbes, Michael
8999 Fenner, Mia
8433 Chakrabarti, Amit
8800 Servedio, Rocco
9865 Beame, Paul
2001 Rossman, Marie
1111 Tan, Li-Yang
5445 Homer, Steve
6577 Green, Susan
9123 Vianu, Victor
5511 Welch, Claire
6009 Mumey, Brendan
4344 Kelley, Sandra
```

CIS 26B
Advanced C
Programming Assignments

Memory Leak Detection

It is a good habit to release the memory when it is no longer needed.

"Memory leaks are among the most difficult bugs to detect because they don't cause any outward problems until you've run out of memory and your call to **`malloc`** suddenly fails. In fact, when working with a language like C or C++ that doesn't have garbage collection, almost half your time might be spent handling correctly freeing memory. And even one mistake can be costly if your program runs for long enough and follows that branch of code."

<u>Windows, Microsoft Visual Studio:</u>
To check if memory was released properly, use CrtDumpMemoryLeaks as described below:

```
//...
  printf( _CrtDumpMemoryLeaks() ? "Memory Leak\n": "No
Memory Leak\n");
    return 0;
} // end of main()
```

`_CrtDumpMemoryLeaks` is a debug function:

. returns TRUE if a memory leak is found;
. otherwise, the function returns FALSE.

Required Header: `#include <crtdbg.h>`
See https://msdn.microsoft.com/en-us/library/e5ewb1h3(vs.80).aspx

<u>Unix</u>
VALGRIND (free download:)
Tutorial: http://www.cprogramming.com/debugging/valgrind.html