CSE 2421 – Systems 1

Spring Semester 2024

Programming Assignment #4

- **This assignment is worth 10pts.**
- **You must upload the zipped solution folder to Carmen, as solution.zip.**
- **The deadline for this assignment is March 1st 11:59pm ET.**
- **Deductions for late submissions apply.**
- **The main topic of this assignment is pointers, dynamic allocation, and structure handling.**

Preliminary Instructions:

- Download the entire folder a4 from Carmen.
- You can debug and do the preliminary implementation of your code in your own laptop or desktop, but the final testing of your code must be done and work in stdlinux or coelinux.
- After downloading the folder a4, copy the entire folder to your favorite OSU linux cluster (stdlinux or coelinux).

Instructions:

You are given two text files, input01.txt and input02.txt. Below we show the contents of file input01.txt

```
5
course01 6 80 80 80 80 80 90
course10 4 10 10 10 70
course35 3 80 70 90
course02 4 90 90 90 100
course20 4 60 80 70 90
```

The structure of this file is as follows:

- The first line states the number of courses to read.
- Next, you will read the information of as many as requested.
- For each course, you will read the course name, a number NC representing the number of grades, to read next, and finally NC grades (int).

You must complete the implementation of three functions:

- read_from_file
- free_structure
- sort

Your implementation should follow the below description:

[3 pts] `read_from_file` must read and load all of the information of the input file to the variable `info` of type `t_course_info**` in the main function. See the definition of `struct s_course_info` at the top of the source file to understand how to dynamically allocate all the required fields. This function should use the provide routine `malloc_wrapper`, which receives the same parameter as `malloc`. Internally, the `malloc_wrapper` will update a global variable `allocated_total` with the number of requested bytes. This variable will tell you how much dynamic memory has been allocated.

[3 pts] `free_structure` must free the entire `info` data structure. Do not use the regular `free` routine. Instead, use the `free_wrapper` function provided. This function, in addition to the pointer to release, must also receive the number of bytes you allocated for the pointer. Every call to this function will decrement `allocated_total` by the number of bytes given. Then, near the end of the program, this variable will report if you have any memory pending to be freed. Your goal is to make this variable to be zero. If all pointers are freed correctly, and you know exactly how much memory you should be releasing (freeing) for each field and pointer, you should end up with exactly zero bytes allocated after calling `free_structure`.

[2pts] Finally, implement the function `sort`, which must sort the elements of `info` by the course name, in non-descending lexicographic order (i.e., from 'a' to 'z'). Printing your data structure after the call to the sort function must show the courses being sorted.

[2 pts] Use of reasonable comments and descriptive variable names.

What and where to upload:
Upload via Assignments->Assignment 4 in Carmen.
Place your file courses.c and a screenshot showing your code working, either in stdlinux or coelinux, in a folder "solution". Zip the folder and upload it to Carmen.