CS261: HOMEWORK 4
Due 06/07/2017 by 12pm

Submit three files via the TEACH website:

`https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth`

General Instructions

In this assignment, you will implement a "to-do list" application with a Heap-based Priority Queue. The application allows the user to manage their prioritized to-do lists using console commands for operations such as adding new tasks, retrieving, or removing the highest priority task. Additionally, users can save the list to a file or load the list from a file. Refer to Worksheets 33 and 34 for more details regarding the heap implementation of the priority queue interface and the heapsort algorithm.

- Complete the implementation of the heap-based priority queue in file `dynArray.c`.
- Implement functions for saving and loading a to-do list to and from a file. These functions are in the files `toDoList.c` and `toDoList.h`.
- Complete functions `_buildHeap()` and `sortHeap()` in file `dynArray.c`, and in this way implement the heap sort algorithm.
- Implement the print function in `toDoList.c` which prints the to-do list in the priority order. We have provided a helper function `copyDynArr()` in file `dynArray.c`, which will be necessary to implement the print function.

Provided Files:

- `dynArray.c` — Implementation of dynamic array and heap-based priority queue. You will finish the functions for heap-based priority queue in this file.
- `dynArray.h` — Header for `dynArray.c`. This file should not be changed.
- `toDoList.c` — Implementation of functions specialized for a to-do list application, such as `saveList()` and `loadList()`. You will finish these functions.
- `toDoList.h` — Header for `toDoList.c`. This file should not be changed.
- `type.h` — Header file for `Task` structure. This file should not be changed.
- `main.c` — Controls the interactions between the user and the program. This file should not be changed. You can use it to test your functions.
- `Makefile` — The program's makefile.
- `todo.txt` — An example of a file that contains a to-do list, which was saved by function `saveList()`. Your function `saveList()` should save a to-do list in a file with the same format. You can also use this file to test your function `loadList()`.

- `program_demo.txt` — Examples of command lines showing how a user can interact with the program. This file is provided for your reference.
- `main2.c` — Tests the heap sort algorithm. You will need to change the `Makefile` and function names in `main.c` and `main2.c` accordingly to switch between testing your implementations of the to-do list application and the heap sort algorithm.

Scoring:

1) `addHeap` 20
2) `_adjustHeap` 25
3) `getMinHeap` 5
4) `removeHeap` 10
5) `_buildHeap` 5
6) `sortHeap` 5
7) `saveList` 10
8) `loadList` 10
9) `printList` 5
10) `answer.txt` 5

What to turn in

You will submit the following completed files:

1) `dynArray.c`
2) `toDoList.c`
3) `answer.txt` [print out of your `printList` function]

Please use this file-naming convention. Make sure your code compiles using the provided makefile on the ENGR server. Zero tolerance for compiling errors. Design a number of test examples to thoroughly check for any errors in your code. If you have any questions regarding HW4, please email `cs261-001-sp17@engr.orst.edu`.