CSE 1320: Intermediate Programming
University of Texas at Arlington
Spring 2021
Dr. Alex Dillhoff

Assignment 6

This assignment focuses on qsort and linked lists.

Your code must compile without any warnings or errors and run without segmentation faults to receive credit. Additionally, any allocated memory must be freed. Points will be taken for inconsistent formatting.

1. Build a monster database that allows the user to view, sort, and save creature information. When viewing and sorting creature data, the data should be dynamically allocated based on the file entries. After printing the requested data to `stdout`, the allocated memory should be released.

   When adding a creature, save the data in CSV format to the database file as the last entry.

   For sorting data, a second submenu should ask the user which stat they want to sort by. Sorting should be done by passing the relevant comparison function to `qsort`. Data should be sorted in descending order only (greatest to least). For sorting strings, you can use the result of `strcmp`.

   Data Format

   ```
   typedef struct {
       int hp;
       int ac;
       int speed;
       char *name;
       char *type;
   } Creature;
   ```

   Other Requirements

   - The CSV file (optional) must be input as a command line argument when running the program. If a CSV file is not given, create a new one for the user.
   - Allocated memory must be freed before the program terminates.
   - Any function declarations, `struct` declarations, and library includes should be in a corresponding header file.
   - Save your files as `problem1.h` and `problem1.c`.

   Example Run

```
----------------------
|    CREATURE DB     |
----------------------
| 1. Add Creature    |
| 2. View Creatures  |
| 3. Sort Creatures  |
| 4. Exit            |
----------------------
> 1
Name: Bandit
HP: 11
AC: 12
Speed: 30
Type: Humanoid
Saved to creatures.csv!

// Viewing
NAME          HP      AC      SPEED    TYPE
Bandit        11      12      30       Humanoid
Chain Devil   85      16      30       Fiend

// Sorting
Select sort criteria (1. Name, 2. HP, 3. AC, 4. Speed, 5. Type)
> 3
NAME          HP      AC      SPEED    TYPE
Chain Devil   85      16      30       Fiend
Bandit        11      12      30       Humanoid
```

2. Create a program that reads in creature data in CSV format and allows the user to search by type. The Creature data should be stored in your program using a linked list. When searching by type, your program should group all creatures of that type together in a separate linked list. The order does not matter. After grouping, the program should traverse through the linked list recursively, printing the name and stats of each creature. It should print the result as seen in the example run.

   Data Format

```c
typedef struct {
    int hp;
    int ac;
    int speed;
    char *name;
    char *type;
} Creature;
```

   Other Requirements

   - The CSV file must be input as a command line argument when running the program.
   - Nodes must be allocated for each creature read in the CSV file.

- Allocated memory must be freed before the program terminates.
- Any function declarations, `struct` declarations, and library includes should be in a corresponding header file.
- Save your files as `problem2.h` and `problem2.c`.

Example Run

```
Enter type: Dragon
NAME                      HP      AC
Adult Copper Dragon       184     18
Adult Gold Dragon         256     19
Adult Green Dragon        207     19
Adult Red Dragon          256     19
```

Create a `zip` file using the name template `LASTNAME_ID_A6.zip` which includes the all required code files. Submit the `zip` file through Canvas.