COP 3223 Program #7: TA schedule

Due on Webcourses: Wednesday, December 2, 2015, 11:00 PM

Program: schedule.c

You have been asked to write a program that will sort information on when the TAs for COP 3223 are available. Your program will read in data on all TA office hour shifts, sort the data, then print out the information in a pre-specified format. You may assume that the maximum number of shifts is 100.

The shift data is stored in a data file and must be read in at the start of your program. The data in this file is organized as follows:

(1) The first line contains an integer indicating the number shifts in the file.
(2) Each additional line contains the following four items:
- A character string indicating the TA's first name
- A character string indicating the day of the week, e.g. Monday
- An integer indicating the start hour of the shift using military time which uses a 24 hour clock, e.g. midnight is 00, 10 pm is 22. Assume that all shifts start and end on the hour.
- An integer indicating the end hour of the shift using military time

You must include the following structure declaration in your code.

```
struct Shift
   {
   char name[100];
   char day_of_week[100];
   int start_hour;
   int end_hour;
   };
```

The data that you read in from the data file must be stored in an array of `Shift` structures.

Your `main()` program should do the following:
1. Call function `read_data()` to read the data into the array of structures.
2. Call function `sort_data()` to sort the data that you read in by the TA's first name.
3. Call function `print_data()` to print out the sorted data in the format specified below.

Your code must include the following functions. You may use additional functions if you desire.

Function #1

```
// Preconditions:   array of structure "Shift" to store data
// Postconditions:  number of shifts read in from data file
// Actions:         Ask user for name of input file.  Read the number
//                  of shifts, then read in the data for all
//                  of the shifts.  Return the number of shifts.
int read_data(struct Shift shift_data[]);
```

This function asks the user for the name of the input file. It opens the file and reads in the number of shifts. It then reads in the data for each shift into one element of the array shift_data[]. The function should close the data file once it is done reading in data. The function returns the number of shifts. The following is what a sample input file might look like:

```
5
Sam    Monday    16 18
Sameer Monday    12 13
Tuan   Tuesday   11 12
Yunus  Thursday  14 16
Shiva  Thursday  11 13
```

Function #2

```
// Preconditions:   array of structure "Shift"
//                  integer value indicating number of shifts
// Postconditions:  none - this function does not return anything.
// Actions:         Sort the shifts by the TA's first name.
void sort_data(struct Shift shift_data[], int num_shifts);
```

This function should sort the data in array `shift_data[]` alphabetically by the TA's first name.

Function #3

```
// Preconditions:   array of structure "Shift"
//                  integer value indicating number of shifts
// Postconditions:  none - this function does not return anything.
// Actions:         Print the sorted data in the format described below.
void print_data(struct Shift shift[], int num_shifts);
```

This function prints out a list of all of the TA shifts in the array `shift_data[]`. Each line of the output contains the TA name, day of the week, start time and end time. Note that the start and end time are printed according to a 12 hour clock with "am" or "pm" instead of according to a 24 hour clock. The columns of your output must be aligned.

For example, for the sample data file shown above, this function will print:

```
TA shifts
=========================================
    Sam     Monday   4:00 pm  to  6:00 pm
 Sameer     Monday  12:00 pm  to  1:00 pm
  Shiva   Thursday  11:00 am  to  1:00 pm
   Tuan    Tuesday  11:00 am  to 12:00 pm
  Yunus   Thursday   2:00 pm  to  4:00 pm
```

Recommendations

Start out by setting up your main program to call the three functions. Set up each function as an empty functions with simply a print statement inside indicating which function it is. Once your main program calls each function correctly, then start filling in the body of each function. Write your `read_data()` and `print_data()` functions first. Once those are working, then write the `sort_data()` function. Use print statements to keep track of what values your variables contain. You may include additional functions if you desire.

Deliverables

Please submit a file titled schedule.c containing your solution to this problems via WebCourses by Wednesday, December 2, 2015 at 11:00 PM:

The code must compile and run in Codeblocks to receive credit. All data type declarations must be at the top/start of their respective functions to receive full points.

Program: **schedule.c**