PHYS 352 – Assignment 2

Due: Tues., Jan 25

Submit code solutions for each of the problems below. Your C source files should be named "assignment2_X.c", where "X" corresponds to the question numbers. Include your name enclosed in C comment tags (ie: `/*YourName*/`) at the top of each program. Create a zip archive containing all of your code, name it "assignment2_YourLastName.zip" (with the appropriate name replacement) and copy it to your `/projects/e20271/student/[netID]/homework` directory by 2 PM on Tuesday. Work on this assignment individually.

1. **Pointers & Structs : temperature sort (10 pt.)**

   This problem combines aspects of the temperature conversion and quicksort routines that were reviewed in class.

   (a) Write a program that reads in the date and the average daily temperature from the 2012 Chicago .csv data. Use `fscanf` to store the average temperature directly into an `float` array. You can ignore the low and high temperatures or scan them into arrays/variables as well. Scan the date into three utility integer variables, `day`, `month`, `year`. Encode the date as a single integer as follows :

   ```
   date = (1000000*day) + (10000*month) + year;
   ```

   and save this in an array.

   (b) Create an interface and an implementation file for a modified quicksort routine that synchronously sorts the date and temperature arrays according to increasing temperature. The interface should be :

   ```
   void swapD( int * ptr1, int * ptr2 );
   void swapT( float * ptr1, float * ptr2 );
   void qsortTD( float * leftT, float * rightT,
                 int * leftD, int * rightD );
   ```

   You are to provide the implementation.

   (c) Use the `qsortTD` function to synchorously sort the average temperature and date arrays. Then loop over the arrays and output the temperature-sorted day, month, year, daily average to a file ("chicago_tempsSorted_2012.csv"). You can use the following to decode the date back into day, month, year :

   ```
       day   = date/1000000;
       month = (date/10000)%100;
       year  = date%10000;
   ```

   The first few lines of the sorted output should resemble :

   ```
   11 2 2012 15.000000
   20 1 2012 16.000000
   19 1 2012 16.000000
   14 1 2012 18.000000
   18 1 2012 20.000000
   ```

```
13 1 2012 20.000000
3 1 2012 21.000000
...
```

(d) Note that although the relationship between the date and average temperature is properly maintined, our approach is rather clunky in that we have to swap two sets of pointers. We would have been better off using *structs* ... try reimplementing with a `struct` and a single `swap` function that operates on pointer to that `struct`.