Learning Goals

- Practice working with `structs`
- Practice applying sorting algorithms

Background

We seem to be breaking records every year for the hottest weather ever recorded. Climate scientists keep track of what are called “new normals” over multiple years so that we can better predict and prepare for conditions in the near future. The official normals are calculated for a uniform 30 year period, and consist of annual/seasonal, monthly, daily, and hourly averages and statistics of temperature, precipitation, and other climatological variables from almost 15,000 U.S. weather stations.

July is the hottest month of the year for most large US cities. Daytime temperatures above 80 degrees Fahrenheit regularly occur nearly everywhere. The exceptions are some cities along the Pacific coast.

In this problem, you will sort the average high temperature values for 10 cities, in decending order.

- Hints
  - When copying one `struct` to another, no need to assign individual elements. The entire `struct` can be assigned in one statement.
  - Even though a `void` function cannot return any values, a `return` statement can be used to terminate the function.

Getting Started

1. Log into code.cs50.io using your GitHub account.
2. Click inside the terminal window and execute `cd`.
3. Execute `wget https://cdn.cs50.net/2022/fall/labs/3/temps.zip` followed by Enter in order to download a zip called `temps.zip` in your codespace. Take care not to overlook the space between `wget` and the following URL, or any other character for that matter!
4. Now execute `unzip temps.zip` to create a folder called `temps`.
5. You no longer need the ZIP file, so you can execute `rm temps.zip` and respond with “y” followed by Enter at the prompt.
6. Finally, right-click or control-click on the `temps` folder and click “Open in CS50 Lab”. You should see the specification for this problem on the left-hand side and its distribution code on the right-hand side.

Implementation Details

The `main` function initializes the `temps` array, calls the `sort_cities` function and prints out the array in sorted order. You will use an $O(n^2)$ sorting algorithm of your choice (possibly bubble sort, selection sort, or insertion sort) to sort the array by temperature, in descending order.

Thought Question

- Which of the sorting algorithms did you choose and why?

How to Test Your Code

Your program should behave per the examples below.

```
temps/ $ ./temps
```

Average July Temperatures by City

Phoenix: 107 Las Vegas: 105 Austin: 97 Miami: 97 Denver: 90 Chicago: 85 New York: 85 Boston: 82 Los Angeles: 82 San Francisco: 66 temps/ $

No `check50` for this one!

To evaluate that the style of your code, type in the following at the `$` prompt.

```
style50 temps.c
```

How to Submit

No need to submit! This is an optional practice problem.