Homework #3

In this assignment you will be using quicksort to sort an array of car objects by various criteria.

Define a struct Car as follows:

```
#define MAX_STRING_LENGTH

typedef struct Car_ {
        char make [MAX_STRING_LENGTH];
        char model [MAX_STRING_LENGTH];
        int mpg;        /* Miles per gallon */
} Car;
```

a) **(2 points)** Implement a function called *compareCarsByMakeThenModel* that can be passed as an argument to the compare parameter of the qksort function from the book. *compareCarsByMakeThenModel* should return a value that will cause qksort to sort an array of cars in ascending order (from smallest to largest) by make and, when two cars have the same make, in ascending order by model.

hw3 > hw3.cpp > main()

```cpp
#include <iostream>
#include <stdio.h>
#include <string.h>
#include "qksort.h"

#define MAX_STRING_LENGTH 100

typedef struct Car_ {
    char make [MAX_STRING_LENGTH];
    char model [MAX_STRING_LENGTH];
    int mpg;    /* Miles per gallon */
} Car;

int compareCarsByMakeThenModel(const void *car1, const void *car2) {
    const Car *c1 = (const Car *)car1;
    const Car *c2 = (const Car *)car2;

    // Compare by make
    int compare = strcmp(c1->make, c2->make);          // strcmp compares two strings alphabetically
    if (compare != 0) {                                // if two strings aren't equivalent values
        return compare;                                // < 0 means c1 is alphabetically smaller
    }

    // return model if make is same
    return strcmp(c1->model, c2->model);
}
```

b) **(2 points)** Implement a function called *compareCarsByDescendingMPG* that can be passed as an argument to the compare parameter of the qksort function from the book. *compareCarByDescendingMPG* should return a value that will cause qksort to sort an array of cars in descending order (from largest to smallest) by mpg.

```
int compareCarsByDescendingMPG(const void *car1, const void *car2) {
    const Car *c1 = (const Car *)car1;
    const Car *c2 = (const Car *)car2;

    // compare mpg if make is same
    if (c1->mpg < c2->mpg) {                        // same as strcmp but for integers
        return 1;                                   // except <0 means c2 is smaller
    } else if (c1->mpg > c2->mpg) {
        return -1;                                  // results flipped since desc instead of asc
    } else{
        return 0;
    }
}
```

c) **(2 points)** Implement a function called *compareCarsByMakeThenDescendingMPG* that can be passed as an argument to the compare parameter of the qksort function from the book. *compareCarsByMakeThenDescendingMPG* should return a value that will cause qksort to sort an array of cars in ascending order by make and, when two cars have the same make, in descending order by mpg.

```
int compareCarsByMakeThenDescendingMPG(const void *car1, const void *car2) {
    const Car *c1 = (const Car *)car1;
    const Car *c2 = (const Car *)car2;

    // Compare by make
    int compare = strcmp(c1->make, c2->make);           // strcmp compares two strings alphabetically
    if (compare != 0) {                                 // if two strings aren't equivalent values
        return compare;                                 // < 0 means c1 is alphabetically smaller
    }

    // If make is same, compare mpg (desc)
    return compareCarsByDescendingMPG(car1, car2);
}
```

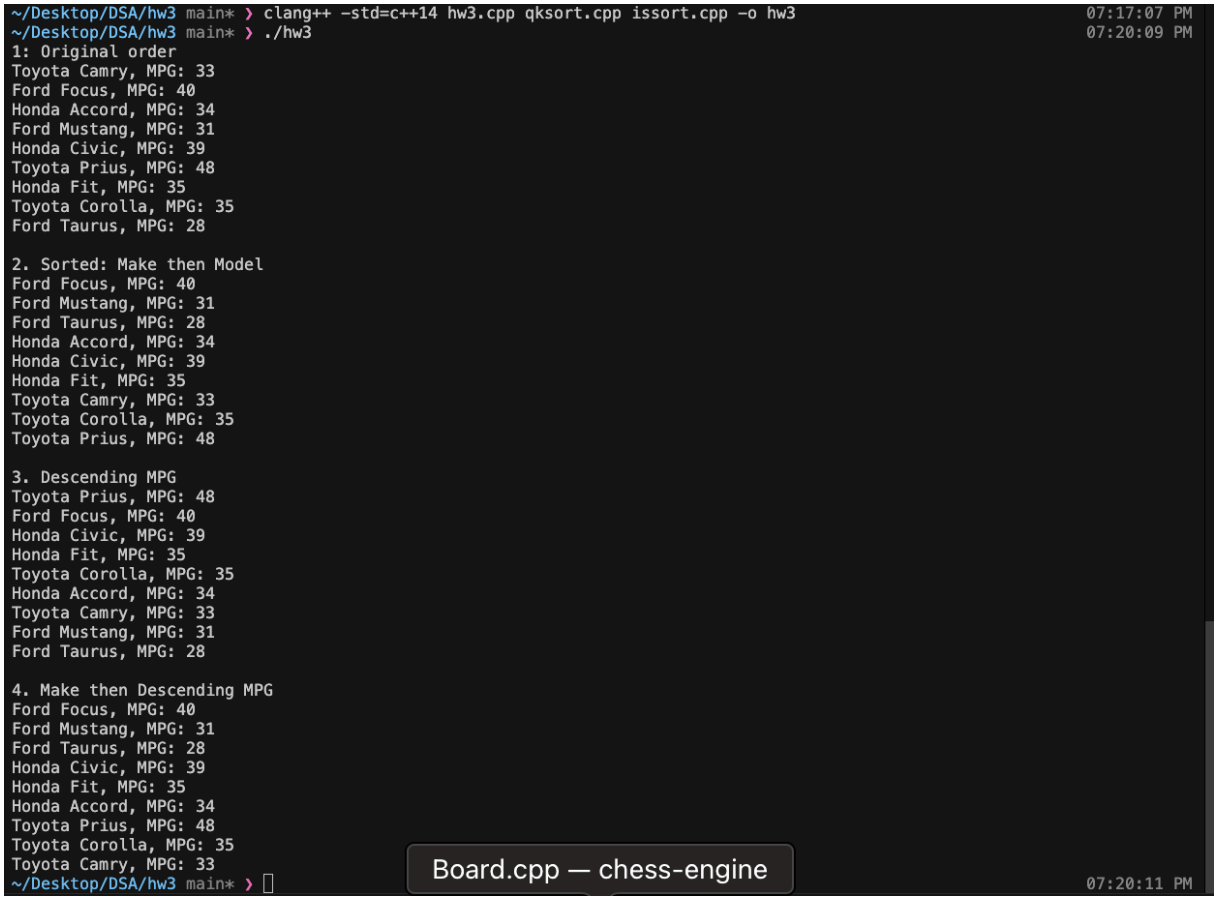
d) **(3 points)** Write a program that tests your functions from parts a-c with the following array of cars:

```
Car cars[] = {
{“Toyota”, “Camry”, 33},
{“Ford”, “Focus”, 40},
{“Honda”, “Accord”, 34},
{“Ford”, “Mustang”, 31},
{“Honda”, “Civic”, 39},
{“Toyota”, “Prius”, 48},
{“Honda”, “Fit”, 35},
{“Toyota”, “Corolla”, 35},
{“Ford”, “Taurus”, 28}
};
```

Your test program should do the following:

1. Output (displaying make, model, and MPG) the cars in original unsorted order
2. Output the cars sorted (using qksort from the book) by make then model.
3. Output the cars sorted (using qksort from the book) by descending MPG.
4. Output the cars sorted (using qksort from the book) by make then descending MPG.

```
void displayCars(const Car cars[], int numCars) {
    for (int i = 0; i < numCars; i++) {
        printf("%s %s, MPG: %d\n", cars[i].make, cars[i].model, cars[i].mpg);
    }
}

int main() {
    Car cars[] = {
        {"Toyota", "Camry", 33},
        {"Ford", "Focus", 40},
        {"Honda", "Accord", 34},
        {"Ford", "Mustang", 31},
        {"Honda", "Civic", 39},
        {"Toyota", "Prius", 48},
        {"Honda", "Fit", 35},
        {"Toyota", "Corolla", 35},
        {"Ford", "Taurus", 28}
    };

    int numCars = sizeof(cars)/sizeof(cars[0]);

    // Output (displaying make, model, and MPG) the cars in original unsorted order
    std::cout << "1: Original order" << std::endl;;
    displayCars(cars,numCars);

    // Output the cars sorted (using qksort from the book) by make then model.
    std::cout << "\n2. Sorted: Make then Model" << std::endl;
    qksort(cars, numCars, sizeof(Car), 0, numCars - 1, compareCarsByMakeThenModel);
    displayCars(cars,numCars);

    // Output the cars sorted (using qksort from the book) by descending MPG.
    std::cout << "\n3. Descending MPG" << std::endl;;
    qksort(cars, numCars, sizeof(Car), 0, numCars - 1, compareCarsByDescendingMPG);
    displayCars(cars,numCars);

    // Output the cars sorted (using qksort from the book) by make then descending MPG.
    std::cout << "\n4. Make then Descending MPG" << std::endl;;
    qksort(cars, numCars, sizeof(Car), 0, numCars - 1, compareCarsByMakeThenDescendingMPG);
    displayCars(cars,numCars);

    return 0;
}
```


~/Desktop/DSA/hw3 main* ❯ clang++ -std=c++14 hw3.cpp qksort.cpp issort.cpp -o hw3 07:17:07 PM
~/Desktop/DSA/hw3 main* ❯ ./hw3 07:20:09 PM
1: Original order
Toyota Camry, MPG: 33
Ford Focus, MPG: 40
Honda Accord, MPG: 34
Ford Mustang, MPG: 31
Honda Civic, MPG: 39
Toyota Prius, MPG: 48
Honda Fit, MPG: 35
Toyota Corolla, MPG: 35
Ford Taurus, MPG: 28
2. Sorted: Make then Model
Ford Focus, MPG: 40
Ford Mustang, MPG: 31
Ford Taurus, MPG: 28
Honda Accord, MPG: 34
Honda Civic, MPG: 39
Honda Fit, MPG: 35
Toyota Camry, MPG: 33
Toyota Corolla, MPG: 35
Toyota Prius, MPG: 48
3. Descending MPG
Toyota Prius, MPG: 48
Ford Focus, MPG: 40
Honda Civic, MPG: 39
Honda Fit, MPG: 35
Toyota Corolla, MPG: 35
Honda Accord, MPG: 34
Toyota Camry, MPG: 33
Ford Mustang, MPG: 31
Ford Taurus, MPG: 28
4. Make then Descending MPG
Ford Focus, MPG: 40
Ford Mustang, MPG: 31
Ford Taurus, MPG: 28
Honda Civic, MPG: 39
Honda Fit, MPG: 35
Honda Accord, MPG: 34
Toyota Prius, MPG: 48
Toyota Corolla, MPG: 35
Toyota Camry, MPG: 33
~/Desktop/DSA/hw3 main* ❯
Board.cpp — chess-engine
07:20:11 PM