UCSD 167113 Data Structure And Algorithms in C/C++
HW3
Cheng FEI

Source Code:

```
//
//  main.c
//  quicksort-cars
//
//  Created by Cheng FEI on 2022/10/8.
//

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "sort.h"

#define MAX_STRING_LENGTH 256

// Car structure with maker's brand (char *), model name (char *), and mpg (int).
typedef struct Car_ {
    char make[MAX_STRING_LENGTH];
    char model[MAX_STRING_LENGTH];
    int mpg;
} Car;

// Function forward declaration.
int compareCarsByMakeThenModel(const void *, const void *);
int compareCarsByDescendingMPG(const void *, const void *);
int compareCarsByMakeThenDescendingMPG(const void *, const void *);

int main(int argc, const char * argv[]) {
    // Initialize array of Car objects.
    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 }
    };

    // Calculate the length of cars array.
    int carSize = sizeof(Car);
    int arrLength = sizeof(cars) / sizeof(Car);

    // Task1: Display make, model and MPG of cars in original unsorted order.
    printf("Task1: Output the cars in original unsorted order.\n\n");
    for (int i = 0; i < arrLength; i++) {
        printf("Car #%d: (make) %s, (model) %s, (mpg) %d\n", i+1, cars[i].make, cars[i].model,
cars[i].mpg);
    }
    printf("--------------------------------------------------\n\n");

    // Taks2: Display cars sorted by make then model.
    printf("Task2: Output cars sorted by make then model.\n\n");
    if (qksort(cars, arrLength, carSize, 0, arrLength-1, compareCarsByMakeThenModel) < 0)
printf("Quick sort fails!");
    for (int i = 0; i < arrLength; i++) {
        printf("Car #%d: (make) %s, (model) %s, (mpg) %d\n", i+1, cars[i].make, cars[i].model,
cars[i].mpg);
    }
    printf("--------------------------------------------------\n\n");

    // Taks3: Display cars sorted by descending MPG.
    printf("Task3: Output cars sorted by descending MPG.\n\n");
```

```
    if (qksort(cars, arrLength, carSize, 0, arrLength-1, compareCarsByDescendingMPG) < 0)
printf("Quick sort fails!");
    for (int i = 0; i < arrLength; i++) {
        printf("Car #%d: (make) %s, (model) %s, (mpg) %d\n", i+1, cars[i].make, cars[i].model,
cars[i].mpg);
    }
    printf("-------------------------------------------------\n\n");

    // Taks4: Display cars sorted by make then descending MPG.
    printf("Task4: Output cars sorted by make then descending MPG.\n\n");
    if (qksort(cars, arrLength, carSize, 0, arrLength-1, compareCarsByMakeThenDescendingMPG) < 0)
printf("Quick sort fails!");
    for (int i = 0; i < arrLength; i++) {
        printf("Car #%d: (make) %s, (model) %s, (mpg) %d\n", i+1, cars[i].make, cars[i].model,
cars[i].mpg);
    }
    printf("-------------------------------------------------\n\n");

    return EXIT_SUCCESS;
}

/*
 Compare Car objects by their makes then models.
 Params: pKey1 -- pointer to void object with car1's info.
 Params: pKey2 -- pointer to void object with car2's info.
 Return: 0 -- if both makes and models of cars are the same.
         1 -- if car1's make is larger than car2's make, or
              if makes of both cars are the same, but car1's model is larger than car2's model.
         -1 -- if car1's make is smaller than car2's make, or
               if makes of both cars are the same, but car1's model is smaller than car2's model.
 */
int compareCarsByMakeThenModel(const void *pKey1, const void *pKey2) {
    // pointer to Car objects to store cars info from pKey1 and pKey2.
    Car *pCar1 = (Car *) pKey1;
    Car *pCar2 = (Car *) pKey2;

    // Compare make and model separately using built-in strcmp function.
    int makeCmpResult = strcmp(pCar1->make, pCar2->make);
    int modelCmpResult = strcmp(pCar1->model, pCar2->model);

    // Compare make of cars first.
    if (makeCmpResult > 0) return 1;
    else if (makeCmpResult < 0) return -1;
    else {
        // If cars' makes are the same, compare their models next.
        if (modelCmpResult > 0) return 1;
        if (modelCmpResult < 0) return -1;
    }

    // If both make and model of cars are the same, return 0.
    return 0;
}

/*
 Compare Car objects by descending MPG.
 Params: pKey1 -- pointer to void object with car1's info.
 Params: pKey2 -- pointer to void object with car2's info.
 Return: 0 -- if MPGs of both cars are the same.
         1 -- if MPG of the first car is smaller than that of the second car.
         -1 -- if MPG of the first car is larger than that of the second car.
 */
int compareCarsByDescendingMPG(const void *pKey1, const void *pKey2) {
    // pointer to Car objects to store cars info from pKey1 and pKey2.
    Car *pCar1 = (Car *) pKey1;
    Car *pCar2 = (Car *) pKey2;

    if (pCar1->mpg > pCar2->mpg) return -1;
    if (pCar1->mpg < pCar2->mpg) return 1;
    return 0;
}

/*
```

```
 Compare Car objects by make then descending MPG.
 Params: pKey1 -- pointer to void object with car1's info.
 Params: pKey2 -- pointer to void object with car2's info.
 Return: 0 -- if both makes and models of cars are the same.
         1 -- if car1's make is larger than car2's make, or
              if makes of both cars are the same, but car1's model is larger than car2's model.
         -1 -- if car1's make is smaller than car2's make, or
              if makes of both cars are the same, but car1's model is smaller than car2's model.
 */
int compareCarsByMakeThenDescendingMPG(const void *pKey1, const void *pKey2) {
    // pointer to Car objects to store cars info from pKey1 and pKey2.
    Car *pCar1 = (Car *) pKey1;
    Car *pCar2 = (Car *) pKey2;

    // Compare make using built-in strcmp function.
    int makeCmpResult = strcmp(pCar1->make, pCar2->make);

    // Compare make of cars first.
    if (makeCmpResult > 0) return 1;
    else if (makeCmpResult < 0) return -1;
    else {
        // If cars' makes are the same, compare their MPGs next.
        if (pCar1->mpg > pCar2->mpg) return -1;
        if (pCar1->mpg < pCar2->mpg) return 1;
    }
    return 0;
}
```

Program Output:

```
Task1: Output the cars in original unsorted order.

Car #1: (make) Toyota, (model) Camry, (mpg) 33
Car #2: (make) Ford, (model) Focus, (mpg) 40
Car #3: (make) Honda, (model) Accord, (mpg) 34
Car #4: (make) Ford, (model) Mustang, (mpg) 31
Car #5: (make) Honda, (model) Civic, (mpg) 39
Car #6: (make) Toyota, (model) Prius, (mpg) 48
Car #7: (make) Honda, (model) Fit, (mpg) 35
Car #8: (make) Toyota, (model) Corolla, (mpg) 35
Car #9: (make) Ford, (model) Taurus, (mpg) 28
-------------------------------------------------

Task2: Output cars sorted by make then model.

Car #1: (make) Ford, (model) Focus, (mpg) 40
Car #2: (make) Ford, (model) Mustang, (mpg) 31
Car #3: (make) Ford, (model) Taurus, (mpg) 28
Car #4: (make) Honda, (model) Accord, (mpg) 34
Car #5: (make) Honda, (model) Civic, (mpg) 39
Car #6: (make) Honda, (model) Fit, (mpg) 35
Car #7: (make) Toyota, (model) Camry, (mpg) 33
Car #8: (make) Toyota, (model) Corolla, (mpg) 35
Car #9: (make) Toyota, (model) Prius, (mpg) 48
-------------------------------------------------

Task3: Output cars sorted by descending MPG.

Car #1: (make) Toyota, (model) Prius, (mpg) 48
Car #2: (make) Ford, (model) Focus, (mpg) 40
Car #3: (make) Honda, (model) Civic, (mpg) 39
Car #4: (make) Honda, (model) Fit, (mpg) 35
Car #5: (make) Toyota, (model) Corolla, (mpg) 35
Car #6: (make) Honda, (model) Accord, (mpg) 34
Car #7: (make) Toyota, (model) Camry, (mpg) 33
Car #8: (make) Ford, (model) Mustang, (mpg) 31
Car #9: (make) Ford, (model) Taurus, (mpg) 28
-------------------------------------------------

Task4: Output cars sorted by make then descending MPG.
```

```
Car #1: (make) Ford, (model) Focus, (mpg) 40
Car #2: (make) Ford, (model) Mustang, (mpg) 31
Car #3: (make) Ford, (model) Taurus, (mpg) 28
Car #4: (make) Honda, (model) Civic, (mpg) 39
Car #5: (make) Honda, (model) Fit, (mpg) 35
Car #6: (make) Honda, (model) Accord, (mpg) 34
Car #7: (make) Toyota, (model) Prius, (mpg) 48
Car #8: (make) Toyota, (model) Corolla, (mpg) 35
Car #9: (make) Toyota, (model) Camry, (mpg) 33
------------------------------------------------

Program ended with exit code: 0
```