Problem Set 4 Exercise #20: Customer Spending

Reference: Week 11 notes

Learning objective: Array of Structures; Sorting

Estimated completion time: 70 minutes

Problem statement:

In today's world, many applications use database technology to manage a large amount of data in an organized way.

For example, an online shopping web application may store data about its customers, the items being sold, and the orders placed by the customers on the items. Such data can be queried for many practical purposes, such as identifying which customer has not been making purchases recently and find out which item is the top seller in the past week.

In this exercise, you are to write a program to read in some data, store them into arrays of structure variables and answer simple queries on the data.

More specifically, you are given the following structure type definitions:

```
typedef struct {
  int id;
  char name[MAX_LENGTH+1];
} customer_t;
```

```
typedef struct {
  int cusID;
  char category[MAX_LENGTH+1];
  int spending;
} record_t;
```

The variables of the structure type **`customer_t`** are used to store the ids and the names of the customers. For example, {111, "Mary"} refers to a customer whose id is 111 and name is Mary.

In addition, the variables of the structure type **`record_t`** are used to store how much a customer has spent in a particular category of products. For example, {111, "Cosmetics", 500} refers to a customer whose id is 111 and this customer has spent 500 dollars on Cosmetics.

The input to your program consists of multiple parts as shown on the right:

- The number of customers
- The id and the name of each customer
- The number of spending records
- The customer ID, the category and the spending in each record
- A name

```
3
111 Mary
222 Peter
333 Lucy
15
111 Toys 400
222 Cosmetics 300
111 Cosmetics 500
(12 more records...)
Mary
```

Given these inputs, your program should 1) find the spending records of the customer with the given name, 2) sort the records in descending order of the spending, and 3) print only the category and the spending in these spending records.

For example, in the above sample inputs, the given name is Mary. Your program should print `Cosmetics 500` and `Toys 400`, among other spending records for Mary.

Note that the record for Cosmetics must be printed before the one for Toys since the spending for Cosmetic is higher (500 vs 400). Moreover, the second record in the list {222 Cosmetics, 300} is not printed since that record is for customer 222, whose name is Peter.

If 1) a customer of the given name cannot be found, or 2) a customer of the given name can be found but there is no spending record for that customer, your program should print a message to indicate that there is no record for the given name (check sample run #2).

Write a program **spending.c** for the above task. Your program should contain the following functions:

- **`find_record()`** that takes an array of **`customer_t`** variables, an array of **`record_t`** variables, a customer name as parameters, returns the spending records of this particular customer.
- **`sort_record()`** that sorts an array of **`record_t`** variables in descending order of spending.

You may assume that:

1) a name or a category consists of alphabet only and its maximum length is 50.
2) the maximum number of customers is 20.
3) the maximum number of spending records is 100.
4) each customer has a unique id and name.
5) the **`cusID`** in **`record_t`** and **`id`** in **`customer_t`** refer to the same customer, if they have the same value.

Sample run #1:

```
Enter the number of customers: 3
Enter customers:
111 Mary
222 Peter
333 Lucy
Enter the number of records: 15
Enter records:
111 Cosmetics 500
222 Cosmetics 300
111 Toys 400
333 Bags 100
```

```
111 Books 300
222 Toys 600
222 Music 200
333 Cosmetics 200
111 Flowers 200
222 Electronics 500
111 Bags 100
222 Flowers 100
333 Electronics 300
333 Books 400
222 Shoes 400
Enter name: Mary
The records for Mary are as follows:
Cosmetics 500
Toys 400
Books 300
Flowers 200
Bags 100
```

Sample run #2:

```
Enter the number of customers: 2
Enter customers:
111 Mary
222 Peter
Enter the number of records: 1
Enter records:
111 Cosmetics 500
Enter name: Peter
No record can be found for Peter
```