

Problem - 1

Write a program which prompts the user to enter a string and returns the count of all the characters in the string in increasing order of occurrence and lastly, prints the string with the characters sorted in ascending order. The program should first print a ">" symbol as a prompt to the user. The user then enters the string and the program prints the count results. Punctuations, whitespaces and non-printable characters are ignored while processing the input string. Assume that the length of the string is no longer than 256 characters including the null terminating character. ('\0').

The following shows an example where the user enters the string "hello world".

```
> hello world
h: 1
e: 1
w: 1
r: 1
d: 1
o: 2
l: 3
dehllloorw
```

Problem - 3

Write a program that prompts the user to enter a string and two integers, `index` and `length` and prints out a substring based on the 2 integer parameters. The program should first print a ">" symbol as a prompt to the user. All three arguments should be separated by commas. The index argument indicates the index at which the substring starts and the length dictates the length of the substring. If the length of the substring starting from the index exceeds the end of the string, then the string is printed until the end of the string. Assume that the length of the string is no longer than 256 characters including the null terminating character. ('\0').

The following example shows the behavior for the string "I love C Programming" with `index = 10` and `length = 11`.

```
> I love C Programming, 10, 11
Programming
```

Problem - 4

Write a program that records the trips made by a taxi for a taxi company. Your program should allow the user to enter a series of commands which can add taxis, indicated by a unique ID, record trips for a particular taxi in terms of miles driven, reset the mile counter for a taxi as well as print the mileage of all taxis in the database. The prompt for the program should print the character '>' At the prompt, the user will type a command followed by the set of arguments associated with the command. The interactive program should exit when the user enters the string "quit". For implementing the database and keeping track of cars/trips, we suggest you use a structure that keeps track of the car ID and miles driven associated with the car. You can assume that at any given time, there will be no more than 10 cars in the system.

Your program should accept the following commands:

- AddCar <CAR ID>: The AddCar command adds a taxi with the unique ID specified by the user. The CAR ID is a unique, non-negative integer. Only one car with a specific CAR ID should exist in the database at a given time.
- AddTrip <CAR ID> <MILES DRIVEN>: The AddTrip command records a trip made by a particular taxi. The CAR ID is a non negative unique integer and MILES DRIVEN is a non negative float. This command adds the number of miles specified to its total miles driven. Users can enter multiple trips for a particular car. If so, the mileage is added to the total mileage of the car.
- Reset <CAR ID>: The Reset command resets the miles driven counter for a particular taxi specified by the ID.
- Display: The Display command prints out all the taxis in the database alongside the total mileage.
- quit: This command quits the program.

The following is an example of execution of the program:

```
> AddCar 7
```

```
> AddCar 10
```

```
> AddCar 8
```

```
> AddCar 10
```

```
Error! Car with ID 10 already exists in the database.
```

```
> AddTrip 7 14.7
```

```
> AddTrip 8 21.9
```

```
> AddTrip 1 1.8
```

Error! Car with ID 1 doesn't exist in the database.

> AddTrip 7 5.2

> Display

7 19.9

10 0.0

8 21.9

> Reset 8

> Display

7 19.9

10 0.0

8 0.0

> quit

\$

Problem - 5

Use the gcov utility to determine the line coverage produced by executing the program that you wrote for Problem 4. Execute the program and supply it with a single input, "AddCar 1". Use gcov to compute the statement coverage and to produce the "p4.c.gcov" file. Submit the "p4.c.gcov" file.