University of Missouri – Computer Science 2050

Homework 1

In this assignment you will be in charge of creating a small system that keeps track of a parking garage. The system will use four characteristics of each car to identify it. Theses items will be stored in a structure. The system will be able to do a few different things with these identifiers, including issuing tickets and finding out if a car is parked in the garage or not. **Please read the whole document before starting your homework. It will help you in the long run.** Be sure to use comments to describe what you are trying to accomplish throughout your code.

Main Description

In main you will first call the function to **import** the cars from the input file. afterwards you will give the user some options: 1. add time to their car. 2. check the time of their car. -1. Exit. From here you will react depending on their choice. You must run these options in a loop because if they check their time and realize they need to add some they must have the option to do so. After the user is done (they enter -1) the system will run a check to **adjust** the cars that need to be given tickets. After the cars have been adjusted accordingly, the cars in the garage will be output to **the same file** the cars were imported from (so the input file becomes updated). Be sure to manage your memory well so you don’t run into problems. Good Luck!

Function Parameters

/\* Structure Variables

\*

\*

\* license\_plate - the license plate of the car

\* parking\_spot - the parking spot the car is parked in

\* num\_tickets - The number of tickets that have been issued to that car

\* time\_left - The amount of time (in minutes) remaining on the car’s parking spot

\*/

typedef struct car {

char \*license\_plate;

int parking\_spot;

int num\_tickets;

int time\_left;

} Car;

// LICENSEPLATELENGTH - the maximum size of the license plate

#define LICENSEPLATELENGTH 10

/\* Parameters

\*

\* filename - the name of the input file

\* numCars - the number of cars in the garage

\*

\* Return - An array full of cars

\*

\*/

Car\* import\_cars(char\* filename, int numCars);

/\* Parameters

\*

\* garage- the array of cars

\* numCars - the number of cars in the garage

\*

\* This function will be called after the user exits

\* the system. It will subtract 30 minutes from the

\* time\_left on each car. If the time left is <= 0 after subtraction, a

\* ticket will be added to the car and the time\_left

\* will be reset to 60.

\*

\*/

void adjust\_cars(Car\* garage, int numCars);

/\* Parameters

\*

\* garage - an array of cars

\* spot - the parking spot the car is located in

\* numCars - the number of cars in the array

\*

\* Return - this function will search the array

\* of cars by the parking spot the car is located

\* in. If it is found, the function will return the

\* index of the array that the car was found in. If

\* the car is not found the function should return -1

\*

\*/

int find\_car(Car\* garage, int spot, int numCars);

/\* Parameters

\*

\* garage - the array of cars

\* numCars- The number of cars in the garage

\*

\* description - this function allows the user

\* to add time (in minutes) for the parking spot specified.

\* It may be beneficial to use the

\* find car function inside this function.

\*

\*/

void add\_time(Car\* garage, int numCars);

/\*Parameters

\*

\* garage - the array of cars

\* numCars - the number of cars in the garage

\*

\* discription - this function will free the

\* space allocated for the cars. Be sure to free

\* all the memory that applies.

\*

\*/

void free\_garage(Car\* garage, int numCars);

/\*Parameters

\*

\* garage - the array of cars

\* numCars - the number of cars in the garage

\*

\* discription - This function will show the

\* user the information stored in the garage

\* struct on the car in the spot specified. It may be beneficial

\* to call the find\_car function inside this function

\*

\*/

void check\_car(Car\* garage, int numCars);

/\* Parameters

\*

\* filename - the name of the output file

\* garage - the array of cars

\* numCars - the number of cars in the garage

\*

\* discription - this function will output the final

\* array of cars to the file. This means after the cars

\* have been updated with the time adjustments and

\* tickets from an above function.

\*

\*/

void print\_to\_file(char\* filename, Car\* garage, int numCars);

Side Notes

* You may create your own functions besides the ones provided, **but you will only be graded for the prototypes given here.**
* Make sure to allocate everything that’s needed (be careful!).
* The C language provided a keyword called **typedef,** which you can use to give a type a new name. For instance typedef allows you to do Car x; , instead of struct Car x; using the structure defined above.
* Make sure to error check everything that you should know how to error check, and check to make sure values update as intended.
* The format of the input file is shown below and in order the categories are:
  + license\_plate
  + parking\_spot, num\_tickets, time\_left (all on the same line, in that order)

Sample Input File

fi590dz

20 2 25

57fjgmc

8 0 55

7dkgjgu

25 1 15

f9e829d

1 2 60

4jgfd81

12 2 10

t6f9dgd

21 1 45

Sample Output

[ajwvz9@tc-login HW1]$ compile hw1.c

[ajwvz9@tc-login HW1]$ ./a.out 6

Incorrect input

Correct usage ./a.out <numCars> <input/output file>

[ajwvz9@tc-login HW1]$ ./a.out 6 garage.txt

To add time enter '1'

To check time enter '2'

To exit enter '-1'

Please enter your choice: 1

Enter the parking spot of the car: 5

There is no car parked in that spot

To add time enter '1'

To check time enter '2'

To exit enter '-1'

Please enter your choice: 1

Enter the parking spot of the car: 8

Enter the amount of time you would like to add: 20

To add time enter '1'

To check time enter '2'

To exit enter '-1'

Please enter your choice: 2

Enter the spot the car is parked in: 21

License Plate: t6f9dgd

Parking Spot: 21

Number of Tickets: 1

Time Left: 45 minutes

To add time enter '1'

To check time enter '2'

To exit enter '-1'

Please enter your choice: -1

[ajwvz9@tc-login HW1]$

Sample Output file

fi590dz

20 3 60

57fjgmc

8 0 45

7dkgjgu

25 2 60

f9e829d

1 2 30

4jgfd81

12 3 60

t6f9dgd

21 1 15

Grading Criteria

import\_cars: 8

adjust\_cars: 10

find\_car: 4

add\_time: 8

free\_garage: 8

check\_car: 8

print\_to\_file: 6

main: 8

Total: 60

* The better you use comments, the higher the chance for partial credit **(this does not mean quantity over quality. Use them smartly, don’t just use them).**
* If your program fails to compile it will be an automatic 0
* If your program has a segmentation fault it will be automatic reduction of 50%.
* If your program has a memory leak it will be an automatic 25% reduction.