Lab #11

CS-2050

April 19, 2024

1 Requirements

In this lab, you are tasked with implementing functions for the purpose of sorting and efficiently searching an array of structs for a car dealership database.

You **may not** use standard library implementations of sorting algorithms or search algorithms for this lab. You must write the implementations of these features yourself.

Note the complexity requirement specified on each function. If your implementation of a function does not meet the complexity requirement specified, you cannot get credit for that function. (Yes, you are free to implement a faster than $O(n^2)$ sorting algorithm if you so desire).

1.1 makeDatabase

```
// O(n^2)

Database * makeDatabase(Car *cars, int size)
```

Info: This function takes an array of Cars, and the size of the array. It creates a Database from the given array, which can be efficiently searched by *both* **price** and **SKU**. You may assume that there will be no duplicates. It returns a pointer to the database on success, or NULL on failure.

1.2 printSKU_Sorted

```
// O(n)

void printSKU_Sorted(Database *db)
```

Info: This function takes a database, and prints the SKUs of all the cars in ascending order (from smallest to largest). The SKUs must be all on one line, separated by commas, and followed by a newline.

```
Example:
Database = { 10057, 10081, 20099 }
// output of printSKU_Sorted
"SKUs: 10057, 10081, 20099"
```

1.3 printPriceSorted

```
// O(n)

void printPriceSorted(Database *db)
```

Info: This function takes a database, and prints the prices of all the cars in ascending order (from smallest to largest). The prices must be all on one line, preceded by a $, separated by commas, and followed by a newline.

```
Example:
Database = { 23185.99, 54899.00, 101233.50 }
// output of printSKU_Sorted
"prices: $23185.99, $54899.00, $101233.50"
```

1.4 getPN_FromSKU

```
// O(log(n))

unsigned long long getPN_FromSKU(Database *db, int SKU)
```

Info: This function takes a Car Database, and returns the OEM_PN assocaited with the car that has the provided SKU. If the car is not found, it returns -1 (which is the same as `ULLONG_MAX`, the compiler will understand what you mean when you return -1 from a function returning unsigned).

1.5 getSKU_FromPrice

```
// O(log(n))

int getSKU_FromPrice(Database *db, double price)
```

Info: This function takes a Car Database, and returns the SKU assocaited with the car that has the provided price. If the car is not found, it returns -1.

1.6 destroy

```
// O(1)

void destroy(Database *db)
```

Info: This function takes a Database, and frees all memory allocated to it.

Grading: 33 points

1. Write required *makeDatabase* function
   * 20 points
2. Write required *printSKU_Sorted* function
   * 1 points
3. Write required *printPriceSorted* function
   * 1 points
4. Write required *getPN_FromSKU* function
   * 5 points
5. Write required *getSKU_FromPrice* function
   * 5 points
6. Write required *destroy* function
   * 1 points

Notice:

1. All of your lab submissions **must** include documentation to receive full points.
2. All of your lab submissions must compile under GCC using the $-Wall$, $-Werror$, and $-Wpedantic$ flags to be considered for a grade.
3. You are expected to provide proper documentation in every lab submission, in the form of code comments. For an example of proper lab documentation and a clear description of our expectations, see the lab intro PDF.