**Electronics Store**

**Problem Statement:**

Leonard owns an electronics store. His employees need an application to help them manage all the products. Each product will be stored in the application in the following way: ID (unique identifier), type (e.g. printer, laptop, speaker), model, manufacturer, price, date (when the product arrived at the store) and quantity. The employees need the application in order to help them with the following functionalities:

a. Each time a new stock of products arrives they need to be able to add the new products.

b. The employees want to be able to update an existing product by modifying the price or its quantity.

c. The application must allow them to delete an existing product.

d. Leonard’s employees would like to be able to filter the products by a given property such as manufacturer, price or date.

e. The store’s employees would like to be able to sort the products by different criteria such as by price or by date (both ascending and descending).

**Functionalities:**

**F1:** Console

**F2:** Add product in file

**F3:** Show all products

**F4:** Update product date and quantity

**F5:** Remove product

**F6:** Filter products

**F7:** Sort products

**Iterations:**

**First iteration:** F1, F2, F3, F4, F5

**Final iterations:** F6, F7

**Specifications:**

**Console:**

/\*

\* Method to run the program

\* Prints the console

\*/

**int** **run**();

**Controller:**

/\*

\* Controller method that adds a product to the repository

\* Returns 0

\*/

**int** **addProduct**(**char**\* type, **char**\* model, **char**\* manufacturer, **int** price, **char**\* date, **int** quantity);

/\*

\* Controller method to get all the products from the repository

\* The products will be placed at the result address

\*/

**void** **getAllProducts**(Products \*result);

/\*

\* Controller method to update the price of a product in the repository

\*/

**int** **updatePrice**(**int** id, **int** price);

/\*

\* Controller method to update the quantity of a product in the repository

\*/

**int** **updateQuantity**(**int** id, **int** quantity);

/\*

\* Controller method to filter all products by manufacturer

\* The resulting list will be put at the result address

\*/

**void** **filterByManufacturer**(Products \*result ,**char**\* manufacturer);

/\*

\* Controller method to filter all products by price

\* The resulting list will be put at the result address

\*/

**void** **filterByPrice**(Products \*result ,**int** price);

/\*

\* Controller method to filter all products by date

\* The resulting list will be put at the result address

\*/

**void** **filterByDate**(Products \*result ,**char**\* date);

/\*

\* Controller method to sort the list of products by price

\* The resulting list will be put at the result address

\*/

**void** **sortByPrice**(Products \*result, **int** direction);

/\*

\* Controller method to sort the list of products by date

\* The resulting list will be put at the result address

\*/

**void** **sortByDate**(Products \*result, **int** direction);

/\*

\* Controller method to delete a product from the repository

\* Returns the result of the deleteProductInRepo function

\*/

**int** **deleteProduct**(**int** id);

**Repository:**

/\*

\* ASC and DSC constants represent the sorting direction passed to the sort functions

\*/

**#define** ASC 1

**#define** DSC -1

/\*

\* The struct that holds the array of products and the size of the list

\*/

**typedef** **struct**{

Product products[100];

**int** size;

}Products;

/\*

\* Method to initialize the repository

\* Reads the products from the file and loads them into memory

\*/

**void** **initRepo**(**char**\* filename);

/\*

\* Method that adds a new product to the repository

\*/

**int** **addProductInRepo**(

**char**\* type,

**char**\* model,

**char**\* manufacturer,

**int** price,

**char**\* date,

**int** quantity);

/\*

\* Method that copies the repository to the result address

\*/

**void** **getAllProductsFromRepo**(Products \*result);

/\*

\* Method that updates the price of a product in the repository

\* Returns 1 if id was not found

\* Returns 0 if the price was updated successfully

\*/

**int** **updatePriceInRepo**(**int** id, **int** price);

/\*

\* Method that updates the quantity of a product in the repository

\* Returns 1 if id was not found

\* Returns 0 if the quantity was updated successfully

\*/

**int** **updateQuantityInRepo**(**int** id, **int** quantity);

/\*

\* Method that deletes a product from the repository

\* Returns 1 if id was not found

\* Returns 0 if the product was deleted successfully

\*/

**int** **deleteProductInRepo**(**int** id);

/\*

\* Method that filters all products by manufacturer

\* The resulting list will be placed at the result address

\*/

**void** **repo\_filterByManufacturer**(Products\* result, **char**\* manufacturer);

/\*

\* Method that filters all products by price

\* The resulting list will be placed at the result address

\*/

**void** **repo\_filterByPrice**(Products \*result ,**int** price);

/\*

\* Method that filters all products by date

\* The resulting list will be placed at the result address

\*/

**void** **repo\_filterByDate**(Products \*result ,**char**\* date);

/\*

\* Method that sorts all products by price

\* The resulting list will be placed at the result address

\*/

**void** **repo\_sortByPrice**(Products \*result, **int** direction);

/\*

\* Method that sorts all products by date

\* The resulting list will be placed at the result address

\*/

**void** **repo\_sortByDate**(Products \*result, **int** direction);

**Domain:**

/\*

\* The structure of a product, containing id, type, model, manufacturer, price, date and quantity

\*/

**typedef** **struct** {

**int** id;

**char** type[30];

**char** model[30];

**char** manufacturer[30];

**int** price;

**char** date[30];

**int** quantity;

}Product;

**Mihai Costiug**

**Group 912**