Sabancı University

Faculty of Engineering and Natural Sciences

CS204 Advanced Programming

Fall 2019-2020

Homework 2 – Customer Carts

Due: **04/10/2019, 11:55pm**

PLEASE NOTE:

Your program should be a robust one such that you have to consider all relevant programmer mistakes and extreme cases; you are expected to take actions accordingly!

You HAVE TO write down the code on your own.

You CANNOT HELP any friend while coding.

Plagiarism will not be tolerated!

Introduction

The aim of this homework is to get familiar with basic manipulation with pointers and simple/doubly linked lists, such as: finding (searching for) a node in a list, dynamically adding/deleting a node in a list, creating new list, printing a list, printing a certain node in a list, as well as cooperation of both of those kinds of lists when building a more complex frameworks of linked lists of (doubly) linked list. Your job here is to implement a system for an E-commerce Store. The system will store customer's carts and the items inside. The following operations must be implemented:

1. Add item to the carts of customers,
2. Remove item from customer's cart,
3. Add a customer,
4. Remove a customer,

as well as extra operations given below. Since neither the number of customers, nor the items they will purchase are known beforehand, we will have to manage this data structure dynamically. Thus, you will dynamically allocate (this is covered in the lectures) and free memory (this is in the slides and will be covered on Monday) during runtime, in accordance to your needs. This means that you are going to use the `new` statement when allocating a block of memory from the heap and the `delete` statement when freeing a memory block you won't need anymore. Pay attention to avoid memory leaks (unused memory which has not been freed) as well as accessing already freed (deleted) memory chunks.

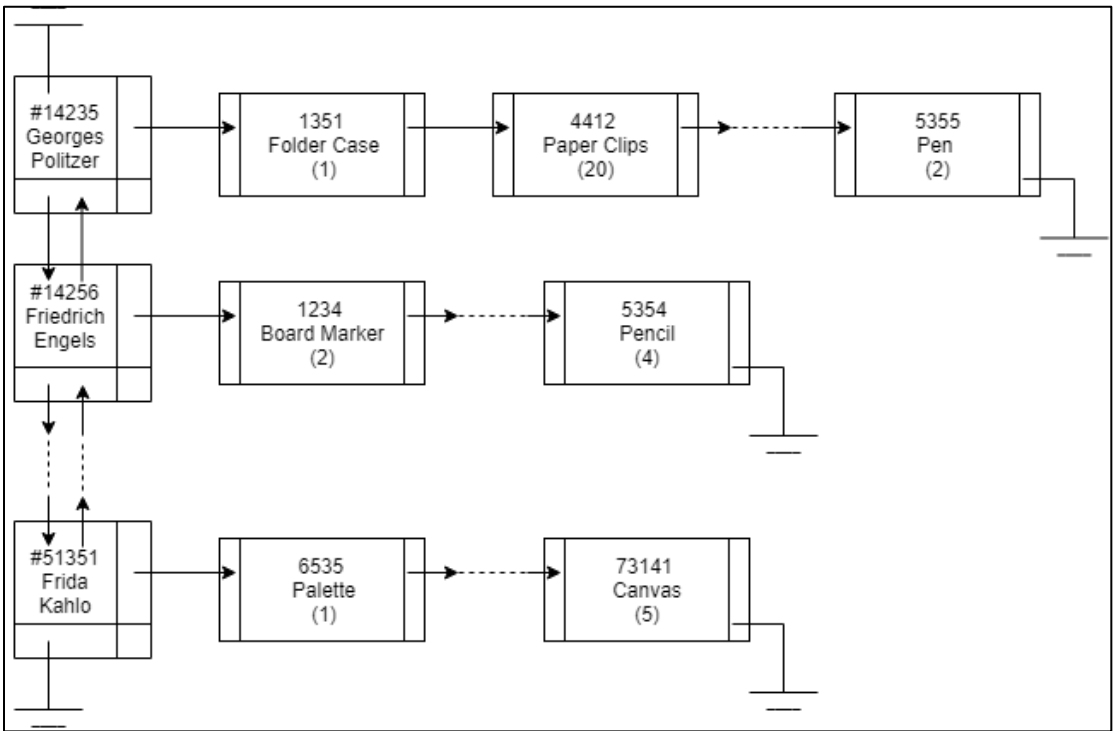
Input file

The input file is shown below: A record for a customer starts with # proceeded by his/her unique ID (e.g., #189754). After a comma, the customer's full name is given (e.g. Georges Politzer). Then after a colon (:), the item(s) in her/his cart are given, each in a single row, until a full stop is reached. The items are separated by a comma. Each item starts with a unique integer ID (e.g., 5355). After an empty space, the name of item is given (e.g. Pen). In the end, the amount the customer wants to buy from that item is given in parenthesis (e.g. (2)).

```
#14235, Georges Politzer:   5355 Pen (2),
                            1351 Folder Case (1),
                            4412 Paper Clips (20).

#51234, Ludwig Andreas Feuerbach:   1234 Board Marker (4),
                                    1351 Folder Case (2),
                                    1452 Color Sheet (50),
                                    5355 Pen (1).

#51351, Frida Kahlo:    73141 Canvas (5),
                        6535 Palette (1),
                        6536 Flat Brush 2 (1),
                        6537 Flat Brush 6 (1),
                        6542 Round Brush 12 (1),
                        7412 Linseed Oil (1).
```

Data structure

In the beginning, you are asked to extract the data from the input file and construct a data structure consist of single linked lists of products in the cart on a doubly linked lists of customers, as it is illustrated in the figure below.



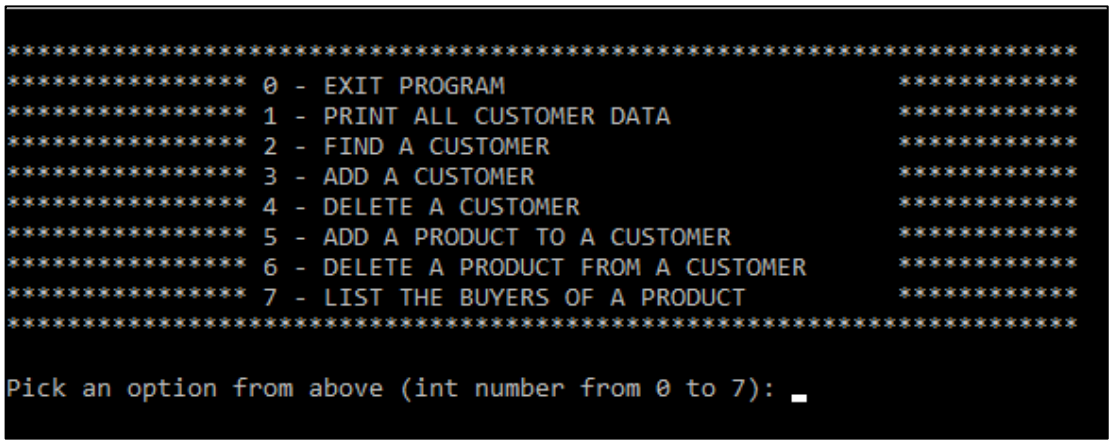
For each **Product node,** you are asked to store its **ID (int)**, **product's name (string)** and **amount (int)** of it. For each customer, you should store **customer's unique ID (int)** and **his/her full name (string)**.

Both the customer's linked list and product linked list for each customer **should always be sorted** (say in ascending order) with respect to the IDs of either customers' or products'. Note that there can be multiple customers with the same name and surname, but all customers will have a different ID. Knowing that there might be thousands of customers, maintaining a sorted doubly linked list for customers at any stage would improve performance when searching for and/or editing certain customer's data. We expect from you to utilize this fact when writing your assignment. **In addition, a doubly linked list will be useful when you are requested to print the customers in reverse order.**

The main menu

Below we give to you the node structure for the doubly and the single linked list. We also show to you a print screen of the main menu, as well as a part of code of the main menu. It should give you a clue how to proceed with each menu option and the corresponding function that implements the option. Of course, you preserve the freedom to add/delete/change functions as you wish, given that the output is the desired one.

```
struct Product {
	int prod_id;
	string prod_name;
	int amount;
	Product * next;
};
struct Customer{
	int cust_id;
	string cust_name;
	Customer * prev;
	Customer * next;
	Product * prod;
};
```




```
while (true){
        cout << endl;
        cout << "**************************************************************************" << endl
                << "********** 0 - EXIT PROGRAM                              **********" << endl
                << "********** 1 - PRINT ALL CUSTOMER DATA                   **********" << endl
                << "********** 2 - FIND A CUSTOMER                           **********" << endl
                << "********** 3 - ADD A CUSTOMER                            **********" << endl
                << "********** 4 - DELETE A CUSTOMER                         **********" << endl
                << "********** 5 - ADD A PRODUCT TO A CUSTOMER               **********" << endl
                << "********** 6 - DELETE A PRODUCT FROM A CUSTOMER          **********" << endl
                << "********** 7 - LIST THE BUYERS OF A PRODUCT              **********" << endl
                << "**************************************************************************" << endl;
        cout << endl;
        int option;
        cout << "Pick an option from above (int number from 0 to 7): ";
        cin >> option;
        switch (option)
        {
        case 0:
                cout << "PROGRAM EXITING ... " << endl;
                system("pause");
                exit(0);
        case 1:
                print_customer_data(head);
                break;
        case 2:
                find_customer(head);
                break;
        case 3:
                add_customer(head);
                break;
        case 4:
                delete_customer(head);
                break;
        case 5:
                add_product(head);
                break;
        case 6:
                delete_product(head);
                break;
        case 7:
                list_product_owners(head);
                break;
        default:
                cout << "INVALID OPTION!!! Try again" << endl;
        }
}
```

Below we describe each of the main menu options (with option 0 being obvious and already implemented).

1) Print All Customer Data

It prints all the recorded customers and their corresponding products in the cart to. The output should be as shown below

```
**************************************************************************
********** 0 - EXIT PROGRAM                                     **********
********** 1 - PRINT ALL CUSTOMER DATA                          **********
********** 2 - FIND A CUSTOMER                                  **********
********** 3 - ADD A CUSTOMER                                   **********
********** 4 - DELETE A CUSTOMER                                **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                      **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER                 **********
********** 7 - LIST THE BUYERS OF A PRODUCT                     **********
**************************************************************************

Pick an option from above (int number from 0 to 7): 1
ID: 14235 Name: Georges Politzer
Items in the cart:
Item ID: 1351 Item name: Folder Case.  Amount: 1
Item ID: 4412 Item name: Paper Clips.  Amount: 20
Item ID: 5355 Item name: Pen.  Amount: 2

ID: 14256 Name: Friedrich Engels
Items in the cart:
Item ID: 1234 Item name: Board Marker.  Amount: 2
Item ID: 2345 Item name: Scissors.  Amount: 1
Item ID: 4123 Item name: Dustless Chalk.  Amount: 5
Item ID: 4412 Item name: Multipurpose A4 Paper.  Amount: 100
Item ID: 5354 Item name: Pencil.  Amount: 4

ID: 42345 Name: Arthur Schopenhauer
Items in the cart:
Item ID: 1452 Item name: Color Sheet.  Amount: 20
Item ID: 2344 Item name: Pin.  Amount: 20
Item ID: 4123 Item name: Dustless Chalk.  Amount: 2
Item ID: 5353 Item name: Highlighter.  Amount: 4
```

2) Find A Customer

This feature should allow the user to search for and find a customer (if he/she exists). The search should be done using either the customer's ID or her/his name.

If a customer is not found, a proper text is printed and the program should send us back to the main menu, as it is shown bellow. Note: Search will be case sensitive, which means Arthur and ARTHUR is not the same.

```
**********************************************************************
********** 0 - EXIT PROGRAM                                  **********
********** 1 - PRINT ALL CUSTOMER DATA                       **********
********** 2 - FIND A CUSTOMER                               **********
********** 3 - ADD A CUSTOMER                                **********
********** 4 - DELETE A CUSTOMER                             **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                   **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER              **********
********** 7 - LIST THE BUYERS OF A PRODUCT                  **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 2
Enter customer's ID or name and surname: 131415
CUSTOMER DOESN'T EXIST. GOING BACK TO MAIN MENU
```

If a search is done giving the name&surname of the customer we want to find, you should take care to give it as a single input (e.g. Frida Kahlo in this case should be taken as a single string). (Hint: use getline(cin, name)). If the customer is found, his/her information is printed in the following manner

```
**********************************************************************
********** 0 - EXIT PROGRAM                                  **********
********** 1 - PRINT ALL CUSTOMER DATA                       **********
********** 2 - FIND A CUSTOMER                               **********
********** 3 - ADD A CUSTOMER                                **********
********** 4 - DELETE A CUSTOMER                             **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                   **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER              **********
********** 7 - LIST THE BUYERS OF A PRODUCT                  **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 2
Enter customer's ID or name and surname: Frida Kahlo
Customer found!
ID: 51351 Name: Frida Kahlo
Item ID: 6535 Item name: Palette.  Amount: 1
Item ID: 6536 Item name: Flat Brush 2.  Amount: 1
Item ID: 6537 Item name: Flat Brush 6.  Amount: 1
Item ID: 6542 Item name: Round Brush 12.  Amount: 1
Item ID: 7412 Item name: Linseed Oil.  Amount: 1
Item ID: 73141 Item name: Canvas.  Amount: 5
```

Since there might be 2 customer with the same name, if user searches with name and surname, you should print all the customers with the same name and surname.

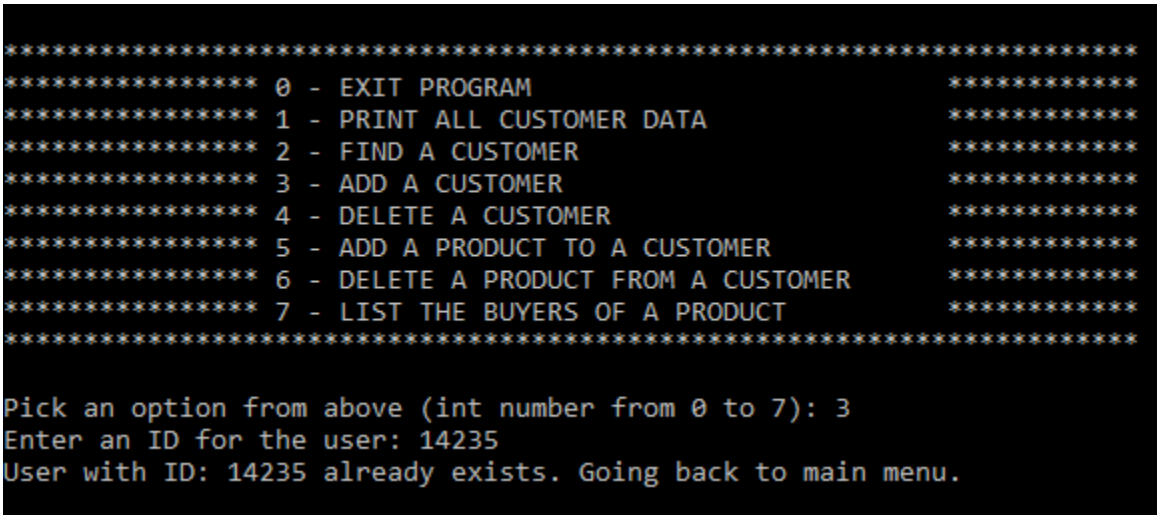
```
**********************************************************************
********** 0 - EXIT PROGRAM                                  **********
********** 1 - PRINT ALL CUSTOMER DATA                       **********
********** 2 - FIND A CUSTOMER                               **********
********** 3 - ADD A CUSTOMER                                **********
********** 4 - DELETE A CUSTOMER                             **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                   **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER              **********
********** 7 - LIST THE BUYERS OF A PRODUCT                  **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 2
Enter customer's ID or name and surname: Georges Politzer
Customer found!
ID: 14235 Name: Georges Politzer
Item ID: 1351 Item name: Folder Case.  Amount: 1
Item ID: 4412 Item name: Paper Clips.  Amount: 20
Item ID: 5355 Item name: Pen.  Amount: 2
Customer found!
ID: 113527 Name: Georges Politzer
Item ID: 5355 Item name: Pen.  Amount: 2
```

Remember that search by name is case sensitive.

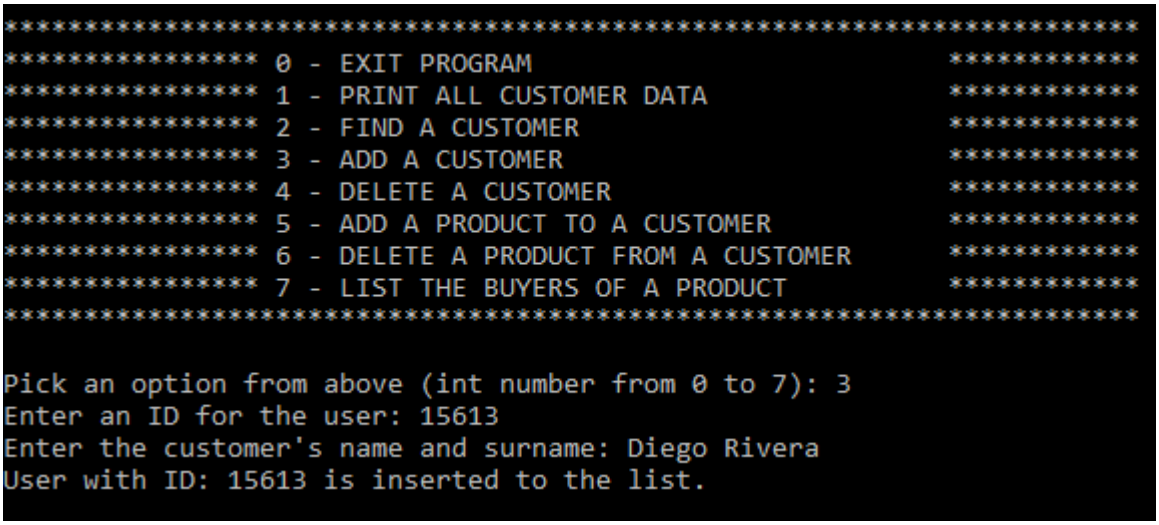
```
**********************************************************************
********** 0 - EXIT PROGRAM                                  **********
********** 1 - PRINT ALL CUSTOMER DATA                       **********
********** 2 - FIND A CUSTOMER                               **********
********** 3 - ADD A CUSTOMER                                **********
********** 4 - DELETE A CUSTOMER                             **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                   **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER              **********
********** 7 - LIST THE BUYERS OF A PRODUCT                  **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 2
Enter customer's ID or name and surname: georges politzer
CUSTOMER DOESN'T EXIST. GOING BACK TO MAIN MENU
```

3) Add A Customer

This option adds a new customer in the list. It first asks for the new customer's ID (an integer). Knowing that ID's are unique, if an already existing ID is entered, a proper message is displayed and the user is returned in the main menu.



If a unique ID is entered, the program asks for the customer's full name and surname (again, given as a single line parameter with multiple parameters). After customer is inserted to the list, a proper must be displayed.



4) Delete A Customer

Deleting is exclusively done by ID (since there might more customer with the same name and surname). If an entered ID doesn't exist a proper message is shown and program prints the main menu. If the ID is found, firstly products in the customer's cart are deleted and then his node. In the end a proper message is shown (given below). The newly freed memory should be added to the heap.

```
**********************************************************************
********** 0 - EXIT PROGRAM                                  **********
********** 1 - PRINT ALL CUSTOMER DATA                       **********
********** 2 - FIND A CUSTOMER                               **********
********** 3 - ADD A CUSTOMER                                **********
********** 4 - DELETE A CUSTOMER                             **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                   **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER              **********
********** 7 - LIST THE BUYERS OF A PRODUCT                  **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 4
Enter an ID for the customer to be deleted: 51234
Customer is deleted succesfully.
```

5) Add a Product

First a prompt is shown asking for the customer's ID. If it doesn't exist, the user is returned to the main menu. If it exists the user is asked whether she/he wants to add a product to the customer's cart by choosing (Y/y) for yes or any character for no. If yes is chosen another prompt asks from the user to give the ID of the product she/he wants to add. If it already exists, a message is displayed telling this, as it is shown below, and the amount of product will be increased. If the ID of product doesn't exist, the user is asked to also give product's name (as a single line string) and afterwards the product is added to the customer. The (Y/y) is repeated after each attempt to add an product. This is illustrated in the figure below. You can assume the inputs will be correct.

```
**********************************************************************
********** 0 - EXIT PROGRAM                                  **********
********** 1 - PRINT ALL CUSTOMER DATA                       **********
********** 2 - FIND A CUSTOMER                               **********
********** 3 - ADD A CUSTOMER                                **********
********** 4 - DELETE A CUSTOMER                             **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                   **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER              **********
********** 7 - LIST THE BUYERS OF A PRODUCT                  **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 5
Enter an ID for the Customer to add product: 14235
Would you like to add item to cart(Y/y): Y
ID of the item to add to cart: 100
Name of the product: Bookshelf
Amount: 3
Would you like to add item to cart(Y/y): Y
ID of the item to add to cart: 4412
Product already exists. How many would you like to add more: 4
Product's amount in the cart increased to: 24
Would you like to add item to cart(Y/y): No
Going back to main menu.
```

6) Delete A Product

Firstly, the user is asked to enter the ID of the customer we wish to delete a product for. If it doesn't exist, we are sent back to the main menu. If it exists, the ID of the product we wish to delete is asked. If it doesn't exist, we are returned to the main menu. Those scenarios are shown below.

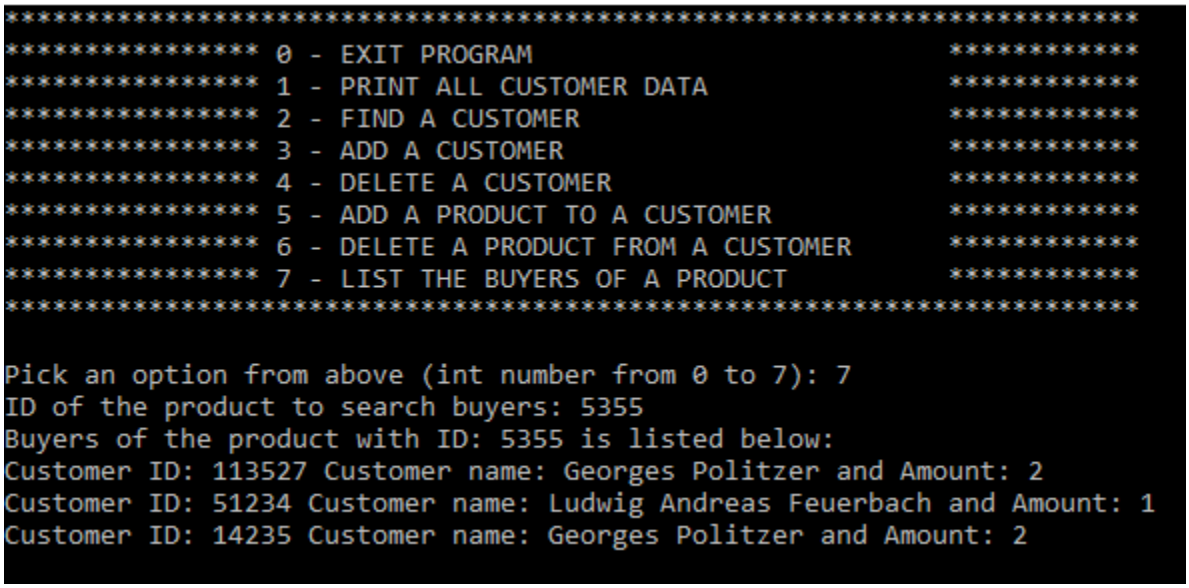
```
**********************************************************************
********** 0 - EXIT PROGRAM                                **********
********** 1 - PRINT ALL CUSTOMER DATA                     **********
********** 2 - FIND A CUSTOMER                             **********
********** 3 - ADD A CUSTOMER                              **********
********** 4 - DELETE A CUSTOMER                           **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                 **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER            **********
********** 7 - LIST THE BUYERS OF A PRODUCT                **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 6
Enter an ID for the Customer to delete product: 123
Customer with ID: 123 doesn't exists. Going back to main menu.

**********************************************************************
********** 0 - EXIT PROGRAM                                **********
********** 1 - PRINT ALL CUSTOMER DATA                     **********
********** 2 - FIND A CUSTOMER                             **********
********** 3 - ADD A CUSTOMER                              **********
********** 4 - DELETE A CUSTOMER                           **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                 **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER            **********
********** 7 - LIST THE BUYERS OF A PRODUCT                **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 6
Enter an ID for the Customer to delete product: 14235
ID of the item to delete from cart: 5
Product doesn't exists. Going back to main menu.
**********************************************************************
```

If an existing customer ID and product ID are entered, the product will be deleted, a proper message will be shown and the user will be send to the main menu, as it is shown below

```
**********************************************************************
********** 0 - EXIT PROGRAM                                **********
********** 1 - PRINT ALL CUSTOMER DATA                     **********
********** 2 - FIND A CUSTOMER                             **********
********** 3 - ADD A CUSTOMER                              **********
********** 4 - DELETE A CUSTOMER                           **********
********** 5 - ADD A PRODUCT TO A CUSTOMER                 **********
********** 6 - DELETE A PRODUCT FROM A CUSTOMER            **********
********** 7 - LIST THE BUYERS OF A PRODUCT                **********
**********************************************************************

Pick an option from above (int number from 0 to 7): 6
Enter an ID for the Customer to delete product: 14235
ID of the item to delete from cart: 5355
Product has been deleted.
Current items in the cart:
ID: 14235 Name: Georges Politzer
Item ID: 1351 Item name: Folder Case.  Amount: 1
Item ID: 4412 Item name: Paper Clips.  Amount: 20

**********************************************************************
```

7) List the Buyers of a Product

A prompt asking for product ID will be shown in the beginning. If there is a product for the entered product ID, the customers who has the product in their carts should be printed with the amounts in their cart. **Printing will be in reverse order with respect to IDs of customers**. If there is no product for the given ID, a proper message should be displayed. Those cases are illustrated below.



Some Important Rules

In order to get a full credit, your programs must be efficient and well presented, presence of any redundant computation or bad indentation, or missing, irrelevant comments are going to decrease your grades. You also have to use understandable identifier names, informative introduction and prompts. Modularity is also important; you have to use functions wherever needed and appropriate.

When we grade your homeworks we pay attention to these issues. Moreover, in order to observe the real performance of your codes, we are going to run your programs in *Release* mode and **we may test your programs with very large test cases**.

What and where to submit (PLEASE READ, IMPORTANT)

You should prepare (or at least test) your program using MS Visual Studio 2012 C++. We will use the standard C++ compiler and libraries of the abovementioned platform while testing your homework. It'd be a good idea to write your name and last name in the program (as a comment line of course).

Submissions guidelines are below. Some parts of the grading process are automatic. Students are expected to strictly follow these guidelines in order to have a smooth grading process. If you do not follow these guidelines, depending on the severity of the problem created during the grading process, 5 or more penalty points are to be deducted from the grade.
Name your cpp file that contains your program as follows:

"SUCourseUserName_YourLastname_YourName_HWnumber.cpp"

Your SUCourse user name is actually your SUNet username that is used for checking sabanciuniv e-mails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name. For example, if your SUCourse user name is valent, name is Valentina, and last name is Tereşkova, then the file name must be:

Valent_Tereskova_Valentina_hw1.cpp

Do not add any other character or phrase to the file name. Make sure that this file is the latest version of your homework program. Compress this cpp file using WINZIP or WINRAR programs. Please use "zip" compression. "rar" or another compression mechanism is NOT allowed. Our homework processing system works only with zip files. Therefore, make sure that the resulting compressed file has a zip extension. Check that your compressed file opens up correctly and it contains your cpp file.

You will receive no credits if your compressed zip file does not expand or it does not contain the correct file. The naming convention of the zip file is the same as the cpp file (except the extension of the file of course). The name of the zip file should be as follows:

SUCourseUserName_YourLastname_YourName_HWnumber.zip

For example zubzipler_Zipleroglu_Zubeyir_hw1.zip is a valid name, but

hw1_hoz_HasanOz.zip, HasanOzHoz.zip

are **NOT** valid names.

Submit via SUCourse ONLY! You will receive no credits if you submit by other means (e-mail, paper, etc.).

Successful submission is one of the requirements of the homework. If, for some reason, you cannot successfully submit your homework and we cannot grade it, your grade will be 0.

Good Luck!
CS204 Team (M. Yusa Erguven, Kamer Kaya)