**E- COMMERCE SYSTEM PROJECT REPORT**

**Course:** Data Structures / CSE 2025.1

**Lecturer:** Murat Can Ganiz

**Semester:** 2021 Spring

**Date:** 12/04/2021

**Group Members:**

Ahmet Faruk Güzel – 150119659

Hüseyin Kerem Mican – 150119629

Muhammet Eren Atala – 150119904

**Titles:**

1. Introduction
2. Detailed Explanation of Program
3. Conclusion

**Introduction**

In this project the E - Commerce System program has been created successfully. The program provides an ordered structure to users that able them to use basic shopping system.

E-Commerce System program needs to run with stable data structure. When the program runs, user will get a menu interface with necessary options. Add a customer, Add Basket, Remove Customer, List Customers’ Specific Product, List Total Shopping Amount of Each Customer and Exit are the six options of the menu. Users may add a new customer, add a new basket or remove a customer by typing relevant number of the option. Also, they may list the customer who bought specific product or total shopping amounts of each customer in alphabetical order. And they may use the exit instruction by typing the number of that option. Adding customer works with name and surname which cannot be same as the other customer in alphabetical order, adding basket requires customer id and product id and the remove customer function needs name and surname to remove the customer and his\her basket from program. Additionally, listing specific product of customers runs with id of the product. It allows user to see which customer has bought the entered product by product’s id. In the option of listing total shopping amounts of each customer, users may see all customers with their number of products in their basket. Lastly, by calling exit option, user may stop the action of program.

All in all, a proper example of an E – Commence System program has been created with a complete data structure. In this title basics and logic of the program and has been explained. In the next part program’s code and main structure will be explained in more detailed way.

**Detailed Explanation of Program**

In this project there are three structs as **CustomerNode, ProductNode** and **BasketNode**. They are based on stack type and all functions has been built supporting that type. The functions will be explained and it will be shown by figures in following explanations. All functions work successfully and they do not come up with any errors.

**newCustomer**: Returns a struct type customerNode. It has an algorithm to create a new customer node in memory. This function’s parameters are id (created by program), name and surname (taken from user input).

**isCustomerEmpty**: The function checks given linked list is empty or not. If it is empty, it returns true. It has a struct parameter to check the linked list.

**pushCustomer:** The function pushes new created customer node which comes from newCustomer to the linked list.It has id, name and surname.

**popCustomer:** This method pops the first element of stack type linked list, has a struct type paramater named root.

**removeCustomer:** This function able user to remove the customer by given input which includes name and surname. It takes three parameters which are rootPtr (to change our linked list), root (root of linked list), id (integer). The id is an integer found by another function through name and surname.

**newProduct:** Returns a struct type productNode. It has an algorithm to create a new product node in memory. Parameters are: id (integer, unique for every item), name (constant char pointer), category (constant char pointer) and price (integer).

**isProductEmpty:** The function checks given linked list is empty or not. If it is empty, it returns true. It has a struct parameter to check the linked list.

**pushProduct:** This function pushes new created product node, which comes from new product to the product linked list. It takes root, id, name, category and price as parameters.

**popProduct:** This method pops the first element of stack type linked list, has a struct type parameter named root.

**newBasket:** Returns a struct type basketNode. It has an algorithm to create a new basket node in memory. There is only one parameter which is the id of the basket.

**isBasketEmpty:** The function checks given linked list is empty or not. If it is empty, it returns true. It has a struct parameter to check the linked list.

**pushBasket:** This function pushes new created basket node which comes from newBasket to the linked list. It takes only root and id as parameter.

**popBasket:** This method pops the first element of stack type linked list, has a struct type parameter named root.

**CustomerSearch:** This function helps user to reach specific customer with given ID. It returns found customer and takes root as parameter beside ID.

**BasketSearch:** This function helps user to reach specific basket with given basket ID. It returns the customers found basket and takes root as parameter beside basket ID.

**ProductSearch:** This function helps user to reach specific product with given product ID. It returns the found product and takes root as parameter beside product ID.

**createBasket:** Checks whether the basket exists or not through given basket id by program. If the customer exists it checks whether that customer has basket with given id. If basket does not exist, firstly it creates basket with given basket id then it pushes given product to the basketProductList. Then it increments the current basket amount parameter by one.

**listProduct:** Lists every single product in alphabetical order. There is one parameter named root.

**productCompare:** Compares every product read from text file given by user. It sorts them in alphabetical order using bubble sort. It has a parameter named root.

**swapNode:** This function is a part of productCompare function, helps to swap data of nodes in linked list. It takes two pointer parameters named ptr1 and ptr2 which are nodes in linked list.

**listCustomer:** Lists every single customer in numerical order. It has one parameter named root.

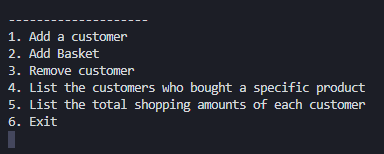
**findProduct:** Looks for specific product id given by user. It searches every basket of all customers and print customer names who bought that specific product. Takes root and productId as parameters.

**listAmount:** Prints the total amount of bought product of every single customer. If customer didn’t buy any product it shows as customer didn’t buy anything. It takes root as a parameter.

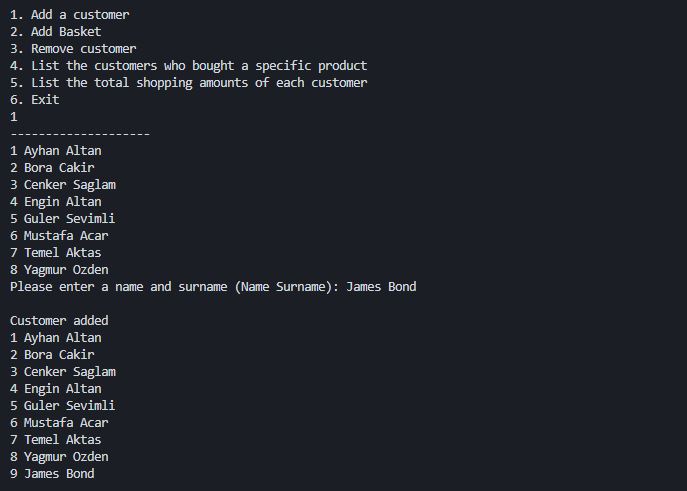
**customerSearchByName:** This function checks whether given name already exists or not. If it exists it returns false, if not, returns true. There are three parameters such as root, name and id.

**UI:** This function is user interface. There are 6 cases such as Add Customer, Add Basket, Remove Customer, List the customers who bought a specific product, List the total shopping amounts of each customer and exit.

**main:** In main function there are three while loop to read given text files and take them as input. Reads and distributes them word by word to various functions that explained previously.

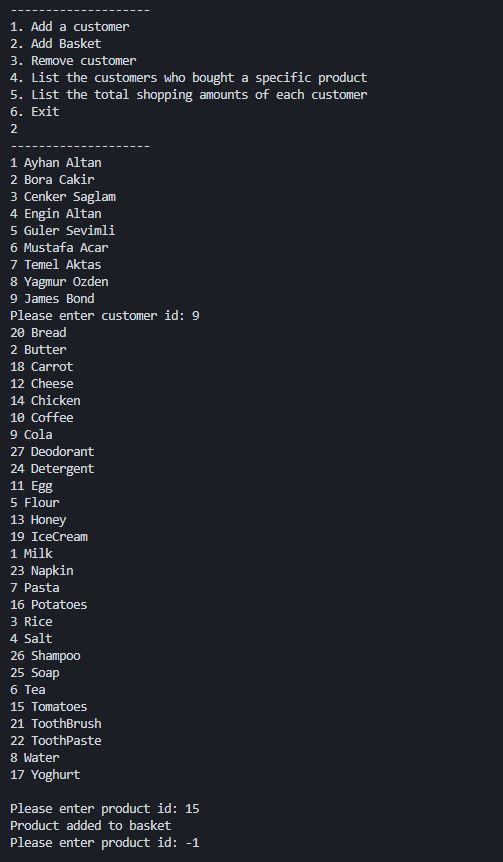


*When the user runs the program, an interface will be encountered. There will be six options to be choose.*



*By typing number of Add Customer option, users may add a new customer by entering new customer’s name and surname. The list of customers will be listed before and after this process.*

*Related Functions: listCustomer, pushCustomer.*



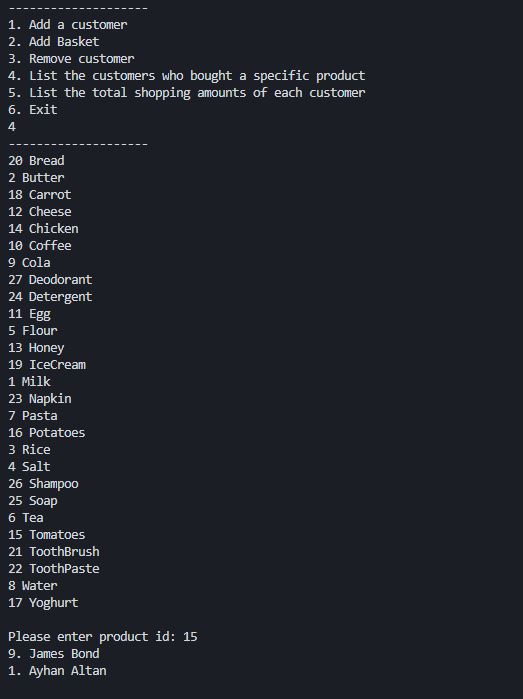
*To add a new basket user, need to choose second option. It will print the list of customers to create a basket for him/her. After entering the id of customer, selected product’s id has to be entered to add it to customer’s basket. Users may type -1 to quit this function.*

*Related Functions: listCustomer, listProduct, pushBasket.*



*To remove a customer, user needs to call third option. Selected customer may be removed by typing his/her name and surname.*

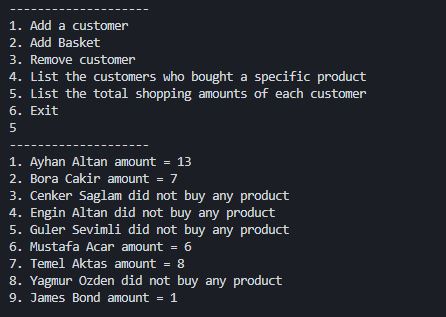
*Related Functions: listCustomer, customerSearchByName, removeCustomer*



*When the user needs to view customers, who bought a specific product, they may call the fourth option. The function will list the all products with their id and it will ask for the id of product to list who had bought the entered product. The customers that purchased the entered product will be listed after this process.*

*Related Function: listProduct, findProduct*

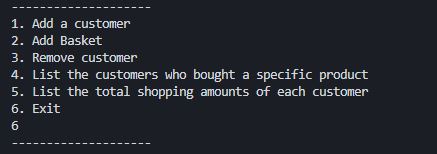
*(It is assumed that the user deleted in the figure (which is id 9) has not been deleted yet.)*



*When the list of total shopping amounts of each customer required, users may call the option 5. It will list all customers with their shopping amounts. If the customer has not bought any product, it will be shown as “Customer did not buy any product”.*

*Related Functions: listAmount*

*(It is assumed that the user deleted in the figure (which is id 9) has not been deleted yet.)*



*To terminate the program, exit option has to be called by typing number of it. This will end the process of program.*

**Conclusion**

The basic elements of a successful example of the E - Commerce System Project are explained under three headings in this report. The definition of the program has been told in introduction part. Second title includes the way of how program and functions work. In this title, the work done in functions has shown as subheads and figures. Every subhead contains the structure and logic of program. To make explanation more clear figures has shown as screenshots and they have been added with descriptions. Every action which may be done in program included in all figures. Program which has stack type data structure has been tested and debugged with different methods. As a result of this process, program did not encounter any errors or problems.