|  |  |  |
| --- | --- | --- |
| Vocational School of Coding and Innovations | | |
| **Theme:** Interoperable Console - Entrepreneur's Choice for Restocking, Evaluation and Maintenance | | |
| **Galaxy Team:**  Emil Ribarev  Martin Atanasov  Vasil Stanchev  Plamen Petkov |  | **Mentor:**  Panayot Tsenkov |

Table of content

[1. Authors and consultant 3](#_Toc39173246)

[2. Resume 4](#_Toc39173247)

[3. Diagram 5](#_Toc39173248)

[4. Description of the main functions 6](#_Toc39173249)

[4.1. showMenu() 6](#_Toc39173250)

[4.2. showAllProducts() 7](#_Toc39173251)

[4.3. showCreateOrderMenu() 7](#_Toc39173252)

[4.4. createOrder() 8](#_Toc39173253)

[4.5. showDeleteMenu() 8](#_Toc39173254)

[4.6. deleteProduct() 9](#_Toc39173255)

[4.7. showUpdateOrderMenu() 10](#_Toc39173256)

[4.8. showRestockMenu() 11](#_Toc39173257)

[4.9. showFlavourStockMenu() 12](#_Toc39173258)

[4.10. showContainerStockMenu() 12](#_Toc39173259)

[4.11. showFlavourRestockMenu() 13](#_Toc39173260)

[4.12. showContainerRestockMenu() 14](#_Toc39173261)

[4.13. runAutoRestock() 15](#_Toc39173262)

# Authors and consultant

Name: Martin Atanasov

Е-mail: mvatanasov18@codingburgas.bg

Name: Emil Ribarev

Е-mail: eiribarev@codingburgas.bg

Name: Vasil Stanchev

Е-mail: VYStanchev18@codingburgas.bg

Name: Plamen Petkov

Е-mail: PCPetkov18@codingburgas.bg

Name: Panayot Tsenkov

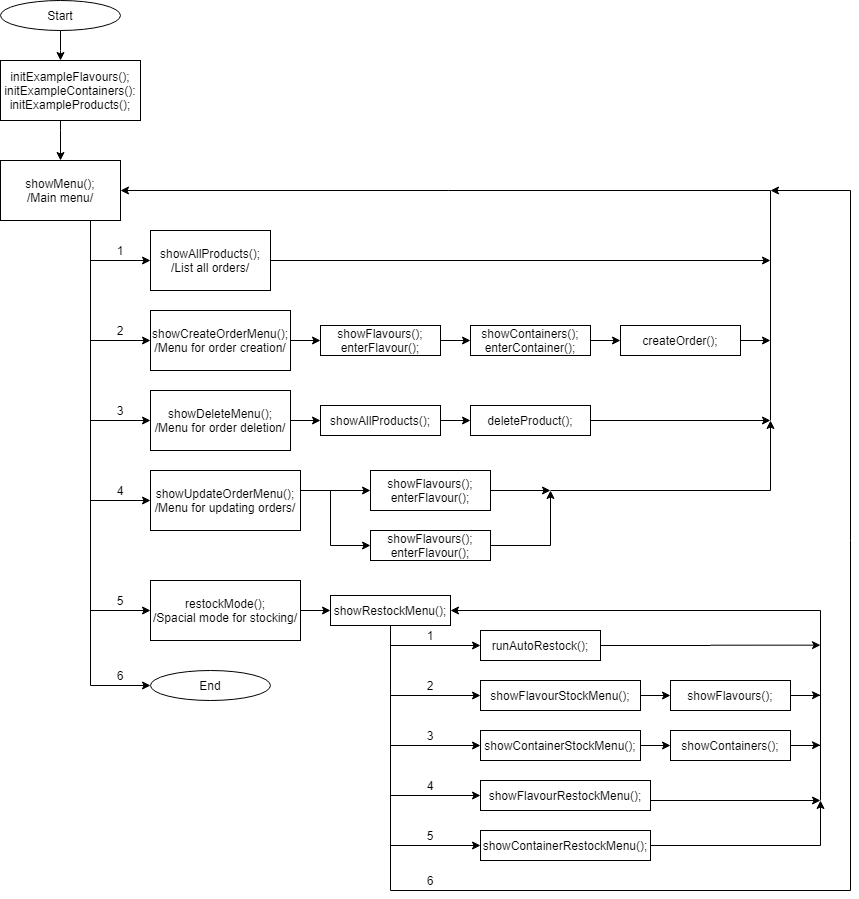
E-mail: ptsenkov@codingburgas.bg

# Resume

Everything in the past few years has been digitalized. But the ice creams were not quite digitalized. So there the idea was born. The team decided to make a C++ program and after weeks, and weeks of putting effort into writing THE ULTIMATE program they present to you THE ICE CREAM PROGRAM. It can be used in every ice cream shop. Unless you do not know English. It is very simple to use. When you start the program in the console it is presented a greeting message and a menu. In the menu there are 6 options:

1. List all orders
2. Create order
3. Delete order
4. Update order
5. Restock the shop
6. Exit

# Diagram



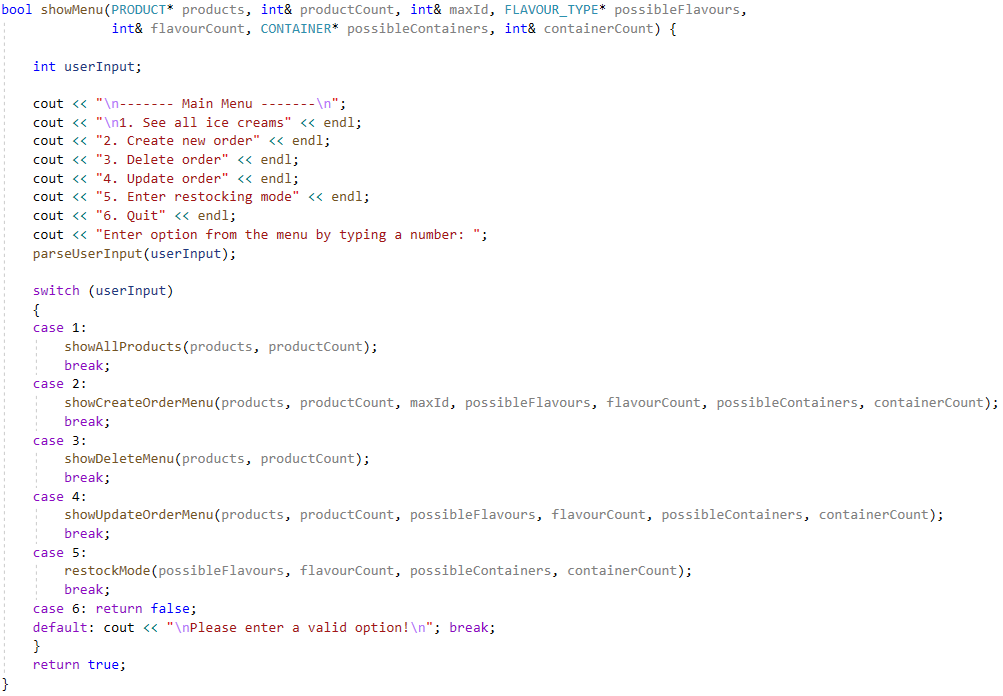
# Description of the main functions

## showMenu()

This is the main menu function of our program. It is responsible for calling the main functions of the different modules of the ICECREAM.

It takes as arguments a pointer to the first element of an array of the ordered products as well as its size, a reference to an integer where the largest unique order ID is stored, a pointer to the first element of an array of possible flavours, its size, and a pointer to the first element of an array of possible ice cream containers and its size.

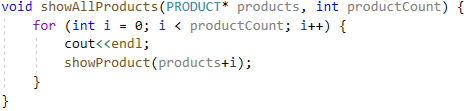
It returns true if the main menu should be shown again and false otherwise.



## showAllProducts()

This function lists all product orders in the system. It is implemented by calling the showProduct() function.

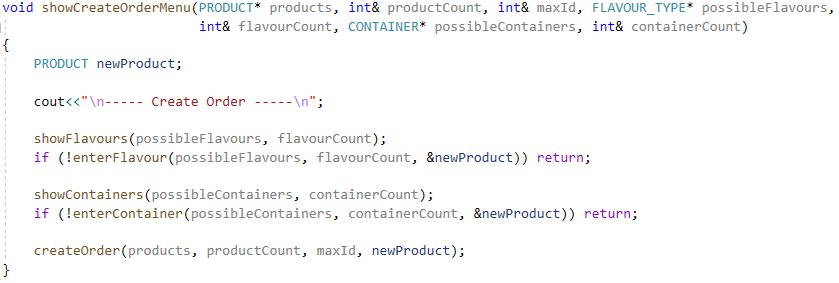
It takes a pointer to the first element of an array of the ordered products as well as its size.



## showCreateOrderMenu()

This is the menu for creating orders. It calls helper functions to present the user with a choice of flavour and container and to create the order in the system.

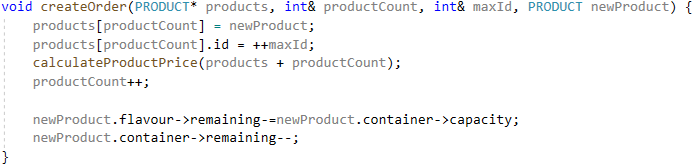
It takes as arguments a pointer to the first element of an array of the ordered products as well as its size, a reference to an integer where the largest unique order ID is stored, a pointer to the first element of an array of possible flavours, its size, and a pointer to the first element of an array of possible ice cream containers and its size.



## createOrder()

This function is responsible for adding a new product order entry in the system. It assigns a unique ID to the order and automatically calculates its price.

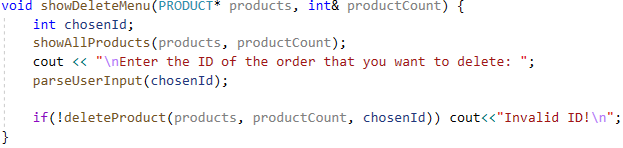
It takes as arguments a pointer to the first element of an array of the ordered products, a reference to its size, a reference to the largest generated ID, and an instance of a PRODUCT structure, representing the new order to be added.



## showDeleteMenu()

This is the menu for deleting orders. It presents the user with a list of all orders and prompts them to choose which one to delete. Then it calls the deleteOrder() function to remove it from the system.

It takes as arguments a pointer to the first element of an array of the ordered products and a reference to its size.

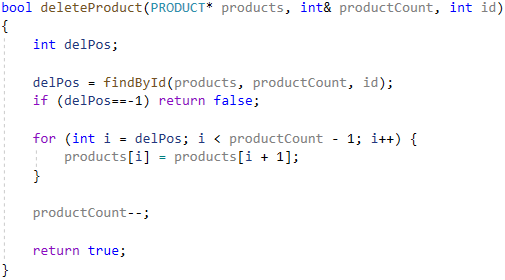


## deleteProduct()

This function removes an order from the system

It takes as arguments a pointer to the first element of an array of the ordered products, a reference to its size and the ID of the order to delete.

The function returns false if no such order exists and true if the operation was successful.



## showUpdateOrderMenu()

This is the menu for updating orders. It lists all orders and prompts the user to choose one by its ID. Then it shows the chosen order and lets the user choose whether they want to change the flavour or the container and then it calls the appropriate helper functions to update accordingly.

It takes as arguments a pointer to the first element of an array of the ordered products as well as its size, a pointer to the first element of an array of possible flavours, its size, and a pointer to the first element of an array of possible ice cream containers and its size.

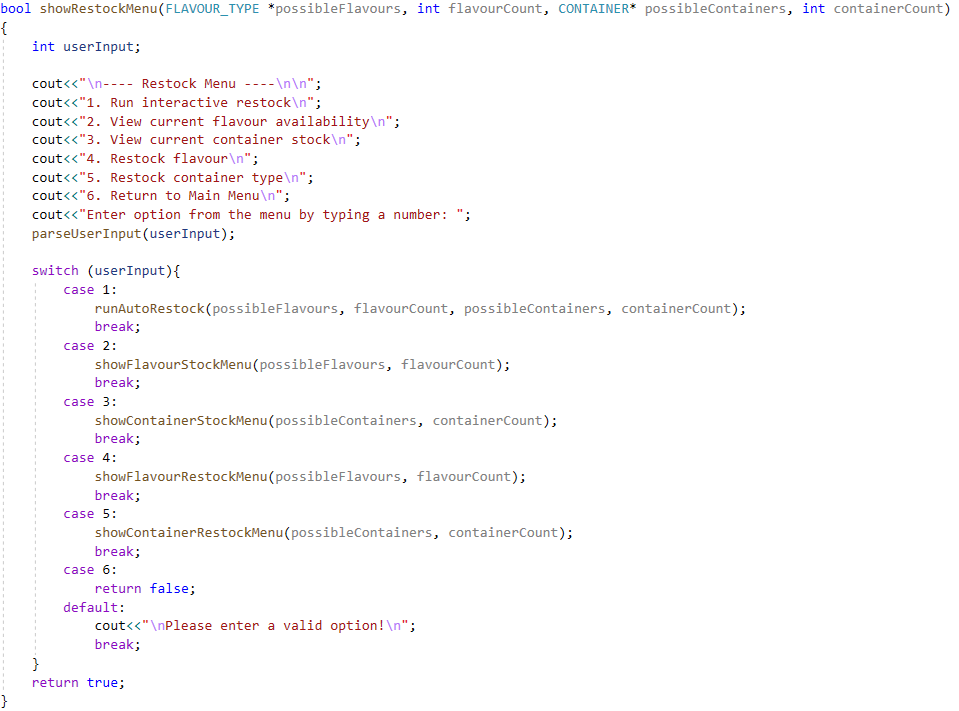


## showRestockMenu()

This is the menu which lets you access the Restock module. It is responsible for calling the functions of that module.

It takes as arguments a pointer to the first element of an array of possible flavours, its size and pointer to an array of possible containers and its size.

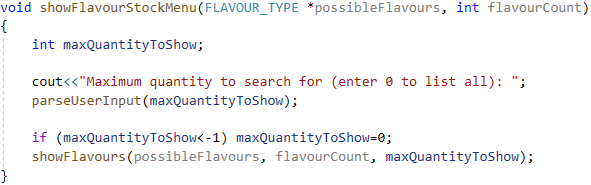
It returns true if the restocking menu should be shown again and false otherwise.



## showFlavourStockMenu()

This is the menu for viewing the flavour stock. It prompts the user for a maximum quantity and lists all flavours with less than that remaining. An exception to this is when the user enters 0. Then all flavours are listed. It is implemented by calling showFlavours().

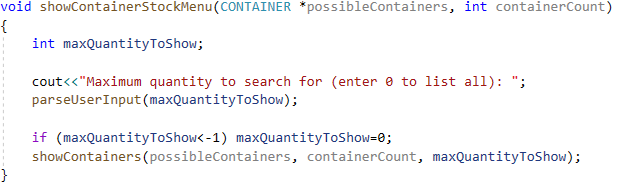
The function takes as parameters a pointer to the first element of an array of possible flavours and its size.



## showContainerStockMenu()

This is the menu for viewing the container stock. It prompts the user for a maximum quantity and lists all containers with less than that remaining. An exception to this is when the user enters 0. Then all containers are listed. It is implemented by calling showContainers().

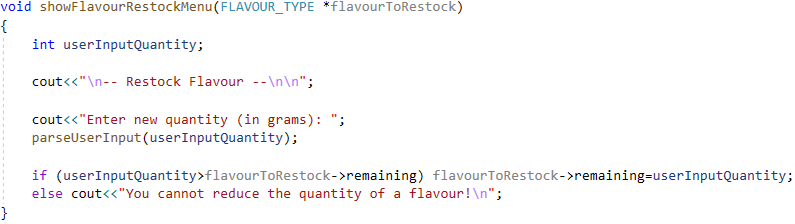
The function takes as parameters a pointer to the first element of an array of possible containers and its size.



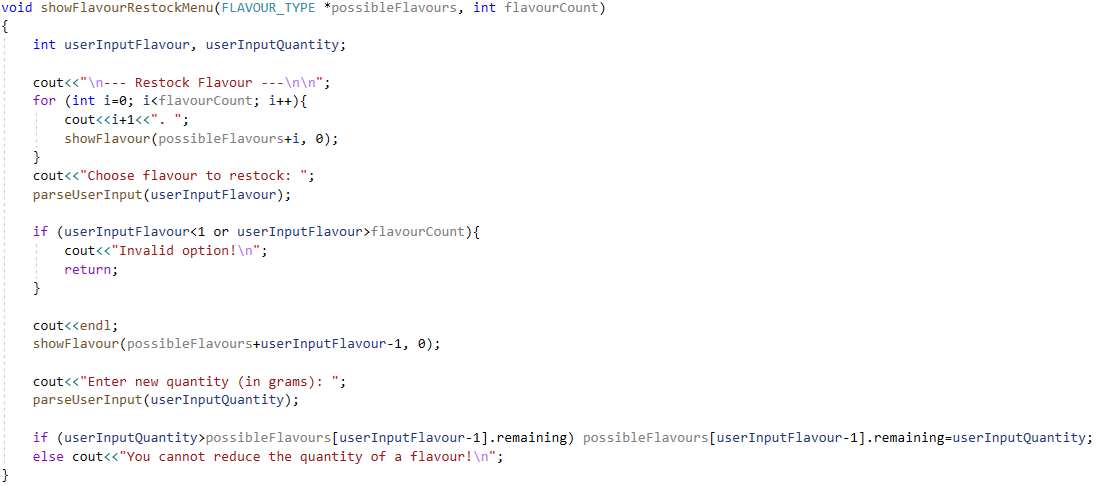
## showFlavourRestockMenu()

This function has two overloads.

The first one has one argument – a pointer to the flavour to be restocked. It prompts the user for a new quantity and sets it if it is larger than the current one.



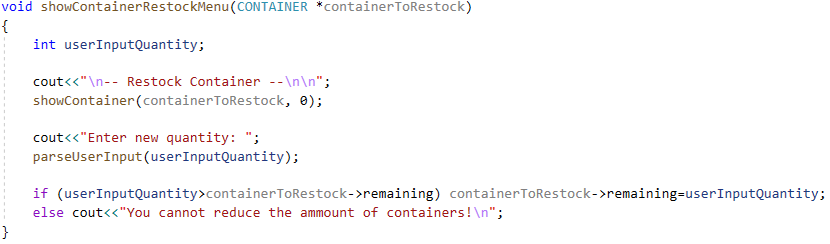
The second one has two arguments – a pointer to the first element of an array of possible flavours and its size. It presents the user with a list of flavours and prompts them to choose one, then it asks for a new quantity and sets it if it is larger than the current one.



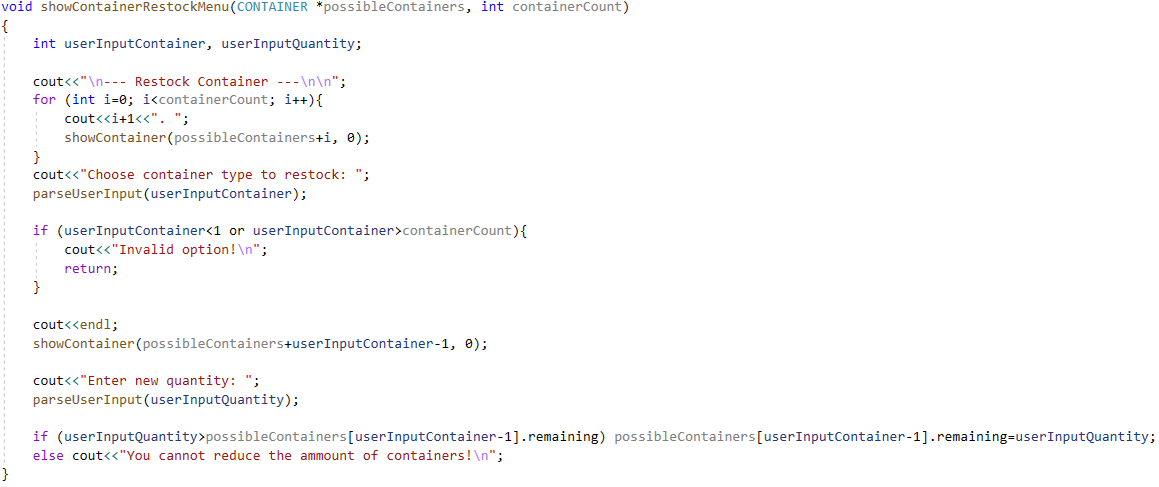
## showContainerRestockMenu()

This function has two overloads.

The first one has one argument – a pointer to the container to be restocked. It prompts the user for a new quantity and sets it if it is larger than the current one.



The second one has two arguments – a pointer to the first element of an array of possible containers and its size. It presents the user with a list of containers and prompts them to choose one, then it asks for a new quantity and sets it if it is larger than the current one.



## runAutoRestock()

This function runs an interactive restock of the flavours and containers. It starts with the flavours and prompts the user for a maximum quantity to list, then it for each flavour with less than that remaining it asks whether the user wants to restock that flavour. If the answer is yes, it calls showFlavourRestockMenu() with and continues. After that it does the same for the containers.

