Homework assignment 1.
Vending terminal software

1 Introduction

The homework assignments for the Programming course will be built around the central task of creating an information system for a company operating vending machines. In the end it will consist of the end terminal software, the central part with a database and communication modules.

As we go through the course and learn new topics you will gradually improve your software by adding new components and changing the existing ones. The assignments however will be graded independently of each other.

Vending machines (VMs) today are used to provide a broad range of operatorless services, from selling snacks, drinks or transport tickets to printing instagram pictures.

Vending machines differ in their user interface. Very simple ones have a small text display and several buttons to take user actions (e.g. choose a snack / drink, add, remove sugar etc.). We will assume however that the target VM possesses a full-size graphical screen with a touch display (so we can emulate it in a desktop application).

In this assignment you will design the terminal part that interacts with the end user.

2 Tested skills

- OOP
- Collections
- Delegates, events, lambda expressions
- Basic GUI

3 Description of tasks

We will assume the following logic of a user interacting with a vending terminal.

1. User inserts banknotes / coins (this step will be emulated by pressing buttons in a separate UI window), the VM accumulates credit and shows it on the screen
2. User selects a product
3. If the credit is high enough to sell the selected product, the latter is given to the user, the credit is decreased by the selected product price.
4. At this point the user can select another product (if the balance is still positive), insert more cash or get the change. **Only coins (1,2,5 and 10RUB) are used to return the change**.

The vending software should store the following information in a file (full state of a VM):

- List of products, for each product: name, price (can be stored as integer) and amount left. Some other attributes specific to the selected service may be added.
- Number of available bills and coins for each denomination (coins - 1,2,5,10 RUB, bills - 50,100,500,1000 RUB).

The state should be saved after each operation (inserting cash, selling a product, giving change), so that if the power fails the state can be restored to the latest version. On program startup the state should be read from a file.

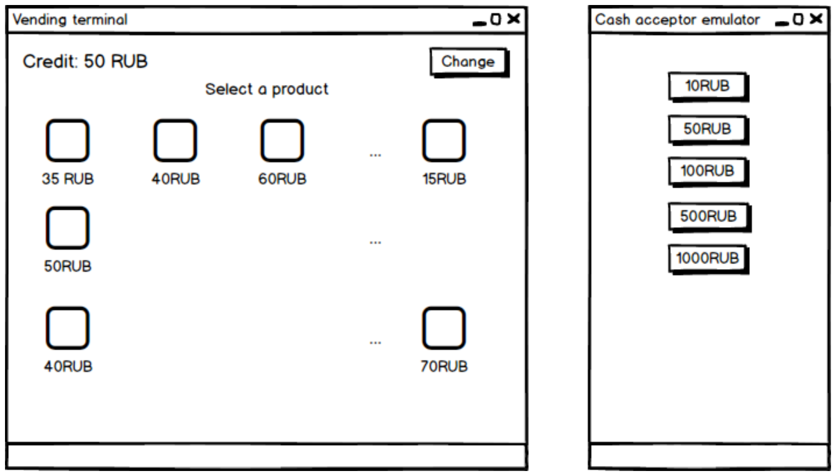
Completing the assignment

Basic part (5 points)

- Pick a service for your vending system and the corresponding products, e.g. VM for selling drinks; products (at least 5): Mineral water, Cola, Orange juice, etc.
- Create a new graphical application. The exact GUI technology is your own choice, WPF is considered as a default path.
- Add a new class that represents a single product, add all the required characteristics to it.
- Design and create a file to store the complete state of a vending machine (see above).
- Design the user interface for your VM. You are not limited to a specific design, in general the application should display the following information:
  - Range of products with prices
  - Accumulated credit
  - "Give change" button

  The number of banknotes and coins of each denomination inside the vending machine should not be shown on the screen
- Implement the key system logic. You will need a separate window with buttons that emulate banknote / coin insertions. On pressing a button the corresponding denomination counter and the overall credit value should be increased

  The following figure shows a possible design mockup. Rectangles in the main window are assumed to display a graphical icon for the corresponding product.



- Implement a simple algorithm for giving the change. It can assume an infinite number of bills/coins of each denomination available in the VM. E.g. to give 67 RUB of change, 6x10RUB coins, 1x5RUB coin and 1x2RUB coin can be returned. The number of bills and coins dispensed in the change can be printed to the console (In GUI applications console output is shown inside the "Output" panel in Visual Studio)

Additional tasks (5 points)

- (+2 points) Implement an advanced algorithm for returning the change based on the amount of banknotes and coins present in the VM. Unlike the simple algorithm, which assumes an infinite number of bills and coins this algorithm should only use the available denominations. E.g. if there are no 10RUB coins left, the VM can only give the change with 1,2 and 5 RUB coins.
- (+1 point) Activate only those of the products that have quantity > 0 and that can be sold with full change. E.g. if the credit is 50RUB, an item costs 37RUB and there are no 1RUB coins left in the VM, prevent this item from being sold to the client (otherwise he/she will not get the full change after buying the product).
- (+2 points) Extract the business logic to a separate class (set of classes) and leave the windows responsible only for diplaying information and reading user input. A possible structure of the application with two separate classes containing the system logic is shown on pastebin (click on the link). In the example one of the classes is responsible only for managing cash, the other - only for managing products and credit. Note that these classes should not have direct references to graphical windows, use mechanisms we discussed to decouple these connections.

4 Submission

Submit your solution following these steps:

1. Delete two temporary folders - “bin” and “obj” from the project folder.
2. Add the whole solution folder to a ZIP **(not RAR or 7Z)** archive.
3. Upload the archive to the Canvas LMS in the corresponding section

5 Grading policy

The final grade for the assignment is determined by the number and quality of completed tasks. An instructor has the option to ask one or two additional questions in case of an unclear grade.

The grade can be lowered in the following cases:

- Inefficient implementation of an algorithm (-1 point)
- Poor programming style (-1 point) (ask your instructor for the definition of “poor”)