**Building a C++ Vending Machine**

The aim of this exercise is to build a C++ application from scratch that emulates the features of a vending machine. We will cover pseudocode briefly before you get the specifications for the application, as it may be very useful to plan the project before you code it.

**Pseudocode**

Pseudocode is the process of writing out the structure and flow of your program before you start coding it properly. It is in essence descriptive text that describes the program you are going to make and how it runs. There is no set style for pseudocode you need to adhere to, but feel free to adopt one that you are comfortable with or you feel works well. You can have your pseudocode structured, or you can describe your program in a large paragraph. The purpose of this is to make you think and plan of about the overall architecture of your applications and it helps you identify areas where there could be issues, such as sanitising user input, checking and handling network messages, how you render the screen etc.

The most common types of pseudocode you will find will most likely be one of the following; descriptive paragraphs of the app, bullet points of the program and how it runs, or more code like examples like so:



**Pseudocode Alternatives**

If you don’t want to use pseudocode, there are visual alternatives that can be used to plan out your programs. These can lose some of the specifics that are easy to write with words and can take up a lot more space but are valid alternatives. The Unified Modelling Language (UML) can be useful for showing the relationship of objects within your applications and flowcharts can easily show the flow and decision making process.

Again, use what you are comfortable with, even if it is as simple as some drawings in a notebook. Planning a program before you write it can help you catch problems before they occur and can help streamline the development process.

Ok, let’s get into making the Vending Machine.

**Specification**

We need a vending machine that allows a user to purchase items and view all purchases made. The machine needs to have 10 numbered buttons. Button 1 dispenses a chocolate bar, Button 2 dispenses a Muesli Bar, Button 3 a pack of Cheese Puffs, Button 4 an Apple and Button 5 a Pack of Popcorn. Button 6 displays to the user all the sales that the application has processed since it started, and the remaining Buttons (0, 7 , 8, 9) have no effect. Each item in the machine needs to cost money that the user enters in when they make their choice. If the user inputs an insufficient amount of money, the machine should state they have insufficient funds and allow them to top up until they have enough. When there is enough money, the machine should dispense the item and any change they are due. You will also need to manage item stock in the machine. All items should start with some stock which is depleted as the user purchases items. If the stock hits 0, then the user cannot buy that item.

Assume that there should be at least 3 classes involved in the vending machine: the Snacks it dispenses, a Vendor mechanism for dispensing the snacks and the MoneyHandler that receives coins, ensures sufficient money has been paid and gives change. The MoneyHandler would also have to tell the Vendor mechanism to release a Snack.

1. Using pseudocode or flowcharts, write down the series of steps needed for the vending machine to function and identify the functions and variables required for each class.
2. Implement your solution in C++.