# Homework – Creating a Vending Machine Class

In this homework, you will create a class representing a vending machine. Your class will encapsulate data and behaviors of a simple coke vending machine.

The vending machine behaves in the following manner:

* The vending machine dispenses only 1 product: Coke which costs $1 for each bottle.
* The vending machine only accepts dollar bills (**one at a time**) and it only accepts exact change, but you can repeatedly add one dollar before purchasing. For example, you can add 1 dollar 3 times, *and then* after doing that, when your balance is $3, you can buy a coke 3 times.
* The vending machine has a money “return” option and it will return all your unused money.
* The vending machine has a “purchase” option which dispenses the Coke bottle (pretend!) **but only if** there are still bottles in the machine **and** the user has some money in the machine’s balance.
* The vending machine starts with only 5 bottles of coke. If the machine is empty, when the “purchase” method is called, it should return an indication that it failed (see below).
* If the machine has no money in it, and the user tries to buy a Coke, it should return the message that the user needs to add money.

* If the machine is both empty and there is no money in it, and the user tries to buy a Coke, the class should return a message to the console program that **both** the machine is empty **and** there is no money on deposit. The console program would in turn let the user know.

**Step 1**: Create a new Visual Studio solution with a Console application project **and** a class library project.

**Important OO Concept**: Never mix data and functions across classes. For example, in this homework, the VendingMachine class should have no code that talks to the Console. Its code maintains the state of the VendingMachine and execute changes (methods) to that state. All user interaction (UI) should be done in the Console Application. Likewise, the Console Application must not maintain any state of the VendingMachine nor make any decisions that belong to the coke machine. For example, do not test the coke machine’s balance in the Console App to decide if the user can buy a coke or not. The console app just blindly tries to buy a coke. It is up to the coke machine class’s logic to make decisions and to return a success or a failure, and then the Console App alerts the user.

**Step 2**: Design and implement your VendingMachine class.

* It should have 2 **private** class fields, one to keep track of how much money the user has put in (Note, since all money is one dollar bills, you can use a simple int to keep track of it.), and one field to keep track of how many bottles are left.
* It should have a constructor method that starts the bottle count field at 5 bottles and sets the money balance field to zero.
* It should have 2 public properties, which are read only (Get*, no Set*), one should return the current number of bottles and the other should return the amount of cash the customer has put in. (These will be used by the Console App to ***display*** information, **not** for the Console App *to make decisions*.)
* It should have 3 methods
  + int BuyCoke() // this method should “do the right thing” inside the class, meaning it should run through logic and dispense or not, reduce the balance and inventory if successful, and then return
    - 0 if it was a success
    - 1 if there are no more bottles left
    - 2 if the user’s balance is zero
    - 3 if there are no more bottles and the user’s balance is zero
  + void AcceptCash() // each time this is called, 1 dollar is added to the user’s balance. Note there is no parameter passed, so you can not, for example, add $4 in one call. One dollar at a time for each call.
  + int GiveRefund() // this method should return the amount of money the user has not used as an int to the calling code (the Console App.), and also set the balance to zero, (pretending that the machine actually refunded the money).
* There should be ***no*** Console.WriteLine() or Console.Readline() statements in this class. All the UI should happen from the console app. This is ALWAYS the case in this course. UI is done from the console app, NOT from any of the other classes. (it is ok to add Console.Writeline(some message) statements as a debugging aid, for example, to see if your code actually gets to certain parts, however these should be removed after the code is debugged.)

**Step 3**: Implement your Console app

* Develop code in the console program. First add the class library as a reference to the Console project. (see the WORD doc “2-Creating a Class Library Solution” if you forget how to do this.) Then instantiate an instance of your class into an object you can use.
* Your Console app should allow a user to enter one of several commands and then do the appropriate actions by making calls to your new VendingMachine object. (A bit like our ATM program did. A switch statement might be useful.)
* At appropriate times, the code should indicate the current amount of money they have in the vending machine (their “balance”). It should start at zero, and go up in 1 dollar increments each time user selects “put a dollar in”, and down when they buy.
  + The user inputs are: P to ***p***ay a dollar, B to ***b***uy a Coke, R ***r****efund* to get all their money back, or Q to ***q***uit the program.
  + Please see the sample console output below that demonstrates the behavior your console program and your class should support.

Yellow highlighted text is user input, what you should type to test your code.

The blue text is what your console program should write out in response to the inputs.

The blue highlighted text is a comment I added just to explain what is going on, your output **will not** show this.

Sample Console Output

*(start your program)*

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

B

Sorry, you need to insert a dollar. Attempt to get coke with no money

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

P

Thank you, you now have $1.00 Enter money, see balance

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

P

Thank you, you now have $2.00 Enter more money, see balance

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

B

Thank you for your purchase, you have $1.00 left. Successful purchase

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

B

Thank you for your purchase, you have $0.00 left.

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

B

Sorry, you have to insert more money.

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

P

Thank you, you now have $1.00

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

P

Thank you, you now have $2.00

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

P

Thank you, you now have $3.00

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

P

Thank you, you now have $4.00

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

B

Thank you for your purchase, you have $3.00 left.

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

B

Thank you for your purchase, you have $2.00 left.

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

B

Thank you for your purchase, you have $1.00 left.

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

B

Sorry, the machine is empty, enter an R to get your money back. Machine is out of bottles

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

R

Here is your $1.00 Returns balance

Please type P to insert a dollar,

or B to buy a Coke,

or R to get all your money back,

or Q to quit this program.

Q

(I don’t show the example, but if the machine gives out 5 bottles and the user balance is also zero, and the user types B, the error message should be

**Sorry, the machine is empty, and you have no money left in the machine.**

When you have tested your code and proved that it works, please zip the entire solution, including both projects and all the files. Watch this video if you are not sure you know how to correctly zip and submit a solution:

<https://youtu.be/bmKcaY0-2mg>