# 03. Vending Machine



*You are tasked with creating a program that simulates a vending machine for hot drinks. The vending machine should be able to store and dispense various types of hot drinks. You should create the necessary classes and methods to implement this functionality.*

**Preparation**

Download the skeleton provided in Judge. **Do not** change the **StartUp** class or its **namespace**.

**Pay attention to name the project VendingSystem, all the classes, their fields, and methods the same way they are presented in the following document. It is also important to keep the project structure as described.**

**Problem Description**

Your task is to create a repository that stores drinks by creating the classes described below.

### Drink

You are given a class **Drink** with the following properties:

* **Name – string**
* **Price - decimal**
* **Volume - int**

The class **constructor** should receive **name, price** and **volume**.

Override the **ToString()** method in the following format:  
**"Name: {Name}, Price: ${Price}, Volume: {Volume} ml"**

### VendingMachine

**Next**, you are given a class **VendingMachine** that has **Drinks** (a List that stores Drinks). The **VendingMachine** class should have the following **properties**:

* **ButtonCapacity - int**
* **Drinks – List<Drink>**
* **GetCount - int** - **returns** the number of **drinks**, **available** in the Vending machine.

The class **constructor** should receive **buttonCapacity**, also it should initialize the **Drinks** with a new instance of the collection.

Implement the following features:

* **Method AddDrink(Drink drink)** – **adds** an **entity** to the **collection** of Drinks, **if** the **Capacity** **allows it**.
* **Method RemoveDrink(string name)** – **removes** a **drink by** **given name,** if such **exists**, and **returns boolean** (**true** if it is removed, otherwise – **false**)
* **Method GetLongest()** – **returns** the **Drink** with the **biggest value** of **Volume** property**.**
* **Method GetCheapest() –** **returns** the **Drink** with the **lowest value** of **Price** property.
* **Method BuyDrink(string name) - returns a string** in the format **of the overriden ToString() method.**
* **Method Report()** – **returns** a **string** in the following **format**:
  + **"Drinks available:  
    {Drink1}  
    {Drink2}  
    (…)"**

**Constraints**

* The **names** of the drinks will be **always unique**.
* You will always have drinks added before receiving methods, manipulating the drinks in the VendingMachine.

**Examples**

This is an example of how the **VendingMachine** class is **intended to be used**.

|  |
| --- |
| **Sample code usage** |
| //Initialize the repository (VendingMachine)VendingMachine vendingMachine = **new** VendingMachine(6);  //Initialize entity (Drink)Drink tea = new Drink("Tea", 1.5m, 300);  Drink coffee = new Drink("Coffee", 2.0m, 120);  Drink hotChocolate = new Drink("Hot Chocolate", 2.5m, 200);  Drink latte = new Drink("Latte", 3.5m, 220);  Drink cappuccino = new Drink("Cappuccino", 2.8m, 180);  Drink mocha = new Drink("Mocha", 2.1m, 150);  Drink herbalTea = new Drink("Herbal Tea", 1.75m, 300);  //Get CountConsole.WriteLine(vendingMachine.GetCount);  //0//Add DrinksvendingMachine.AddDrink(tea);  vendingMachine.AddDrink(coffee);  vendingMachine.AddDrink(hotChocolate);  vendingMachine.AddDrink(latte);  vendingMachine.AddDrink(cappuccino);  vendingMachine.AddDrink(mocha);  //Try to add drinks when the capacity is full  vendingMachine.AddDrink(herbalTea);  //Get CountConsole.WriteLine(vendingMachine.GetCount);  //6  //Remove DrinkConsole.WriteLine(vendingMachine.RemoveDrink("Herbal Tea"));//False  Console.WriteLine(vendingMachine.RemoveDrink("Tea"));//True  //Get Longest DrinkConsole.WriteLine(vendingMachine.GetLongest());  //Name: Latte, Price: $3.5, Volume: 220 ml  //Get Cheapest Drink  Console.WriteLine(vendingMachine.GetCheapest());  //Name: Coffee, Price: $2.0, Volume: 120 ml  //Buy a specific Drink  Console.WriteLine(vendingMachine.BuyDrink("Cappuccino"));  //Name: Cappuccino, Price: $2.8, Volume: 180 ml  //Drinks Report  Console.WriteLine(vendingMachine.Report());  //Drinks available:  //Name: Coffee, Price: $2.0, Volume: 120 ml  //Name: Hot Chocolate, Price: $2.5, Volume: 200 ml  //Name: Latte, Price: $3.5, Volume: 220 ml  //Name: Cappuccino, Price: $2.8, Volume: 180 ml  //Name: Mocha, Price: $2.1, Volume: 150 ml |

**Submission**

Zip all the files in the project folder except **bin** and **obj** folders.