Assignment 2

Drink Machine Simulator

Building the Drink Machine

Write a program that simulates a soft drink machine. The program will need several classes: **DrinkItem**, **DrinkMachine** and **Receipt**. For each of the classes you must create the constructors and member functions required below. You can, optionally, add additional private member functions that can be used for doing implementation work within the class.

DrinkItem class

The `DrinkItem` class will contains the following private data members:

- `name`: Drink name (type of drink – read in from a file). Type is `std::string`.
- `price`: Drink cost (the retail cost of one drink). This is the price the customer will pay for one drink of this type. This is also read in from a file. Type is `double`.
- `quantity`: Number of drinks, of this type, in the machine. Initial value is read in from a file. This is also updated by the program. Type `unsigned int`.
- `purchased`: Drinks purchased. Initially 0. Updated whenever a drink is purchased. Type is `unsigned int`.
- `sales`: This is used to keep track of the amount of money paid for all of the drinks purchased. Every time there is a successful purchase the amount of the purchase should be added to **sales**. You need to initially set this to 0. Type is `double`.

You need to have public accessors for all of the above (five) data members. Make sure you accessors are `const` member functions. Only the **price**, and **name** data members should have a mutator functions. So you will have five get member functions (accessors) but only two set member functions (mutators).

Example:

The prototype for the accessor for **price** would be:

```
double getPrice() const;
```

and the mutator would be:

```
void setPrice(double newPrice);
```

You will also need the following additional public member functions / constructors:

`void addDrinks(unsigned int amount)`

When this member function is called your program needs to update the quantity by the specified amount.

`bool purchase()`

The **purchase** member function will check to see if there are any drinks left. If there is at least 1 drink remaining the **purchase** member function will subtract 1 drink from the number of drinks, add 1 to the drinks purchased, add the cost of the drink to sales, and return a `true` value.

If the number of drinks is 0 when this member function is called no member data should be updated and a value of `false` should be returned.

DrinkItem constructors (two total)

The **DrinkItem** class will have one constructor that takes the input values **name**, **price** and **quantity** (in that order). The **purchased** and **sales** member data variables will be set to 0.

The second constructor is a default constructor (no arguments) that will set the numeric values to 0 and the name to `""`;

Create a header file **drinkitem.h** for the class declaration and create a **drinkitem.cpp** file for the member functions. You can inline the accessors and mutations. Do not inline any of the other member functions and do not inline the constructors.

DrinkMachine class

The `DrinkMachine` class will contain the following data members:

- An `unsigned int` that contains a version number. For this assignment this will have a value of 1.
- An `unsigned int` that contains the actual number of `DrinkItem` objects being used in the drink item array.
- An array of `DrinkItem` objects. Each element of the array will be a `DrinkItem` object. This array will contain a maximum of 25 elements. You will keep track on the actual number in use via the `unsigned int` value you created above.
- An `unsigned int` that contains the maximum number of `DrinkItem` objects. This is used internally by the drink machine to make sure you don't put more than 25 `DrinkItem` objects in the array.

The public member functions in the `DrinkMachine` class are:

DrinkMachine() constructor

The constructor will take no parameters. The various drink machine values will be read in from the file **drink.txt**. The constructor will also write the drink machine values (read in from the file) to a file called **drink_backup.txt**. This way you will have a backup copy of your file in case there are bugs in your code.

The first value in the input file, `drink.txt`, is the number of drink items. There will then be drink information for each of the drink items specified by the first value in the file. A drink item is made up of a name, price and quantity. The first drink item you read in will go into the array of `DrinkItem` value at index 0. The next one will go into the entry with index 1, and so on. The sample file **drink.txt** provided with the exercise. See the section

Input file drink.txt on page 5 for details.

~DrinkMachine() destructor

The destructor will write the current list of items back to the output file, **drink.txt**. This will include the new quantity values for the **DrinkItem** objects.

unsigned int size() const

Returns the current number of `DrinkItem` entries being used by the drink machine. These are the ones read in from the input file.

unsigned int max_size() const

Returns the maximum number of `DrinkItem` entries allowed in the drink machine.

DrinkItem& at(unsigned int index)

Return a reference to the drink item at the specified index.

const DrinkItem& at(unsigned int index) const

Returns a `const` reference to the specified drink item. This allows you to use a function that uses a `const DrinkItem` reference. You cannot update the `DrinkItem` object in this case.

bool available(unsigned int drinkId) const

Return `true` if the drink item at index **drinkId** has a quantity of 1 or more. Return `false` otherwise.

double getPrice(unsigned int drinkId) const

Get the price of the item at the specified index.

Receipt purchase(unsigned int drinkId, double amount)

Purchase an item at the specified index. If the purchase worked return a `Receipt` object with any change. If the drink could not be purchased (the quantity was 0) or if the funds were insufficient return `Receipt` with change of 0 and either insufficient or empty turned on. Check for the funds amount first. If the funds are enough for the purchase then check the quantity. Update the quantity and purchased totals as appropriate. Note your return statement will look similar to the following:

```
Receipt purchase(unsigned int drinkId, double amount)
{
      // code goes here
      return Receipt(…); // where … is the rest of the constructor
}
```

You are returning back a temporary object here. The return type for the purchase member function is also **Receipt**, which is a temporary object. The object will be deleted once it has been assigned in the calling function. You do not want to return by reference, it needs to be a copied temporary object. You also do not want to create it with the new operator or you will be creating a memory leak in your program.

void addDrinks(unsigned int drinkId, unsigned int amount)

Add the specified number of drinks for the specified drink item index.

void print(std::ostream& outStream) const

Print out the current contents of the drink machine including the drink machine version, the number of drink items in the machine and all of the information about the drink items (index, name, price, quantity, number of drinks purchased, and the sales for this drink).

Here is a sample of the output (you must have all of this data in your output). Make sure you align the output as shown here. Your spacing may be different.

```
Drink Machine Version 1
Number of entries: 8

    Id Name                          Price $     Qty    Sold   Sales $
     0 Cola                             1.25      24       1      1.25
     1 Root-beer                        1.25      19       1      1.25
     2 Lemon-lime                       1.25      25       0      0.00
     3 Water                            1.00      39       1      1.00
     4 Orange                           1.25       0       5      6.25
     5 Iced-tea                         1.25      35       0      0.00
     6 Grape                            1.30      15       0      0.00
     7 Iced-coffee                      2.00      34       1      2.00
```

double sales() const

Calculate the sales for all drinks. This is the amount of the sales, not the quantity.

double sales(unsigned int drinkId) const

Calculate the sales for a specific drink. This is the amount of all sales of this drink, not the quantity. You should have the **DrinkMachine::sales(unsigned int)** member function call the appropriate DrinkItem's **getSales()** member function.

Receipt class

The **Receipt** class is returned by the **DrinkMachine::purchase** member function. The calling function will interrogate the **Receipt** object to see if the purchased worked or not.

You will need to determine the member data that needs to be in the class based on the following member function requirements.

The **Receipt** class needs to have the following member functions. You will need to determine the format of the constructor and the member data that has to be part of the class. You can inline the constructor and member functions for this class. If you do this you will only have the **Receipt.h** header file and won't have the **Receipt.cpp** file.

bool success() const

Returns true if the purchase completed without any errors.

bool insufficient() const

Return true if the amount pass for the purchase was less than the purchase price. Note that the price should be checked before you check the quantity. If the amount is insufficient you should not check the quantity.

bool empty() const

Return true if the quantity was 0.

double getChange() const

Return the amount of change that is due to the purchaser of the drink. This amount is valid only if **success()** returns `true`. If either **insufficient()** or **empty()** returns true the **getChange()** member function must return **0.0**.

Input file drink.txt

Here is a sample file that contains the information for one such Drink Machine:

```
8
Cola                    1.25   25
Root-beer               1.25   20
Lemon-lime              1.25   25
Water                   1.00   40
Orange                  1.25    5
Iced-tea                1.25   35
Grape                   1.30   15
Iced-coffee             2.00   35
```

Note that the drink names do not include any spaces in the text. The drink name is first, the cost of the drink is second and the number of drinks in the machine at start up is third. Each value to be read in from the file is a token and you can use the stream operator >> to read in the values from the file.

Your input file must be in the format above. You should try adding additional drink items to the file to make sure you can handle up to 25 drinks. Here is an example with 9 drinks:

```
9
Cola                    1.25   25
Root-beer               1.25   20
Lemon-lime              1.25   25
Water                   1.00   40
Orange                  1.25    5
Java-water              1.45   10
Iced-tea                1.25   35
Grape                   1.30   15
Iced-coffee             2.00   35
```

Driver application

The driver application that makes use of the **DrinkMachine** is file **project2.cpp** and has been provided to you. This code will make use of the **DrinkMachine**, **DrinkItem** and **Receipt** classes. You must not change the driver code. If you have written the classes as specified above the driver code will work without any changes.

Comments, variable names and so on.

Make sure your variable names have meaningful names and that you have included appropriate comments in your application.

You must include the following as your first comments in your header files **(.h**) and in your **.cpp** files.

```
// Assignment 2 for CS 2336.002
// Programmer: <Your name goes here>
// Description:
// <Comments here to describe what the application does>
```

Make sure you replace **<Your name goes here>** with your name. Make sure you include comments at the top of your cpp file that describes what you are doing in the application code (the Drink Machine)

You need to ensure you have comments throughout your program and not just the comments at the top of the program.

Files you need to submit

You need to submit the header files (.h) and source files (.cpp) for your classes:

- `drinkmachine.h`
- `drinkmachine.cpp`
- `drinkitem.h`
- `drinkitem.cpp`
- `receipt.h`
- `receipt.cpp` (you won't have this one if you inline the class)

Note that you are not uploading the **project.cpp** file or the **drink.txt** file. You are not allowed to change the driver code and the format of the drink.txt file must be as shown.

Grading:

20% of the grade will for comments you have in your program, for having meaningful variable names, and for formatting your code as is shown in the C++ text book.

80% will be for the program itself. Make sure you do all of the parts and that your output matches what is shown in the sample runs.

Here is a sample run of the application

The output that follows shows the successful purchase of drinks, trying to purchase a drink and not having enough money, and trying to buy a drink that is out of stock. This is using the default input file and the driver code that has been provided to you.

```
Menu # Name                          Price $     Qty
     0 Cola                             1.25      25
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       5
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
```

```
     7 Iced-coffee                    2.00      35
     8 exit program
Enter a menu item: 0[Enter]
Enter the amount of the purchase: 1.25[Enter]
Your purchase of Cola succeeded.  There was no change

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       5
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      35
     8 exit program
Enter a menu item: 4[Enter]
Enter the amount of the purchase: 1.5[Enter]
Your purchase of Orange succeeded.  Your change is $0.25

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       4
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      35
     8 exit program
Enter a menu item: 4[Enter]
Enter the amount of the purchase: 1[Enter]
You need additional money for the purchase

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       4
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      35
     8 exit program
Enter a menu item: 4[Enter]
```

```
Enter the amount of the purchase: 2[Enter]
Your purchase of Orange succeeded.  Your change is $0.75

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       3
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      35
     8 exit program
Enter a menu item: 4[Enter]
Enter the amount of the purchase: 1.25[Enter]
Your purchase of Orange succeeded.  There was no change

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       2
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      35
     8 exit program
Enter a menu item: 7[Enter]
Enter the amount of the purchase: 2.01[Enter]
Your purchase of Iced-coffee succeeded.  Your change is $0.01

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       2
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      34
     8 exit program
Enter a menu item: 4[Enter]
Enter the amount of the purchase: 2[Enter]
Your purchase of Orange succeeded.  Your change is $0.75
```

```
Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       1
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      34
     8 exit program
Enter a menu item: 4[Enter]
Enter the amount of the purchase: 2[Enter]
Your purchase of Orange succeeded.  Your change is $0.75

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       0
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      34
     8 exit program
Enter a menu item: 4[Enter]
Enter the amount of the purchase: 2[Enter]
Your selection is empty

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
     2 Lemon-lime                       1.25      25
     3 Water                            1.00      40
     4 Orange                           1.25       0
     5 Iced-tea                         1.25      35
     6 Grape                            1.30      15
     7 Iced-coffee                      2.00      34
     8 exit program
Enter a menu item: 4[Enter]
Enter the amount of the purchase: 1[Enter]
You need additional money for the purchase

Menu # Name                          Price $     Qty
     0 Cola                             1.25      24
     1 Root-beer                        1.25      20
```

```
     2 Lemon-lime                    1.25      25
     3 Water                         1.00      40
     4 Orange                        1.25       0
     5 Iced-tea                      1.25      35
     6 Grape                         1.30      15
     7 Iced-coffee                   2.00      34
     8 exit program
Enter a menu item: 4[Enter]
Enter the amount of the purchase: 2[Enter]
Your selection is empty

Menu # Name                       Price $     Qty
     0 Cola                          1.25      24
     1 Root-beer                     1.25      20
     2 Lemon-lime                    1.25      25
     3 Water                         1.00      40
     4 Orange                        1.25       0
     5 Iced-tea                      1.25      35
     6 Grape                         1.30      15
     7 Iced-coffee                   2.00      34
     8 exit program
Enter a menu item: 9[Enter]
The menu item is not valid

Menu # Name                       Price $     Qty
     0 Cola                          1.25      24
     1 Root-beer                     1.25      20
     2 Lemon-lime                    1.25      25
     3 Water                         1.00      40
     4 Orange                        1.25       0
     5 Iced-tea                      1.25      35
     6 Grape                         1.30      15
     7 Iced-coffee                   2.00      34
     8 exit program
Enter a menu item: x[Enter]
Invalid input "x" has been discarded

Menu # Name                       Price $     Qty
     0 Cola                          1.25      24
     1 Root-beer                     1.25      20
     2 Lemon-lime                    1.25      25
     3 Water                         1.00      40
     4 Orange                        1.25       0
     5 Iced-tea                      1.25      35
     6 Grape                         1.30      15
```

```
     7 Iced-coffee                    2.00      34
     8 exit program
Enter a menu item: 1[Enter]
Enter the amount of the purchase: 2[Enter]
Your purchase of Root-beer succeeded.  Your change is $0.75

Menu # Name                         Price $     Qty
     0 Cola                            1.25      24
     1 Root-beer                       1.25      19
     2 Lemon-lime                      1.25      25
     3 Water                           1.00      40
     4 Orange                          1.25       0
     5 Iced-tea                        1.25      35
     6 Grape                           1.30      15
     7 Iced-coffee                     2.00      34
     8 exit program
Enter a menu item: 3[Enter]
Enter the amount of the purchase: 1[Enter]
Your purchase of Water succeeded.  There was no change

Menu # Name                         Price $     Qty
     0 Cola                            1.25      24
     1 Root-beer                       1.25      19
     2 Lemon-lime                      1.25      25
     3 Water                           1.00      39
     4 Orange                          1.25       0
     5 Iced-tea                        1.25      35
     6 Grape                           1.30      15
     7 Iced-coffee                     2.00      34
     8 exit program
Enter a menu item: 8[Enter]


Drink Machine Version 1
Number of entries: 8

    Id Name                         Price $     Qty    Sold   Sales $
     0 Cola                            1.25      24       1      1.25
     1 Root-beer                       1.25      19       1      1.25
     2 Lemon-lime                      1.25      25       0      0.00
     3 Water                           1.00      39       1      1.00
     4 Orange                          1.25       0       5      6.25
     5 Iced-tea                        1.25      35       0      0.00
     6 Grape                           1.30      15       0      0.00
     7 Iced-coffee                     2.00      34       1      2.00
```

```
The total sales is $11.75

Now restock the drink machine
Drink Orange is out of stock. Restocking with 5 drinks.


Drink Machine Version 1
Number of entries: 8
```

Id	Name	Price $	Qty	Sold	Sales $
0	Cola	1.25	24	1	1.25
1	Root-beer	1.25	19	1	1.25
2	Lemon-lime	1.25	25	0	0.00
3	Water	1.00	39	1	1.00
4	Orange	1.25	5	5	6.25
5	Iced-tea	1.25	35	0	0.00
6	Grape	1.30	15	0	0.00
7	Iced-coffee	2.00	34	1	2.00

Let me know if you have any questions.