**Assignment1**

***Bag Class Abstraction***

**Due Date: October 5th 11:59pm**

**Purpose:** This assignment will help you get practice with:

* Abstract Data Types
* Inheritance
* Polymorphism
* C++ pointers
* C++ Classes
* C++ virtual functions
* C++ Class Templates
* C++ Vector
* C++ statics

You are tasked with creating a hierarchy of C++ bag classes that simulate different types of bags for storing items. Each bag class should have specific characteristics and behaviors. Use templates, inheritance, and polymorphism to achieve this.

TODO: You are asked to provide implementation for each class described below:

**Bag**

* Represents the base class for all types of bags.
* Contains the following members:
  + Items: a vector to store items of any data type (use templates)
  + Size: an integer representing the current size of the bag
* Implement a constructor to initialize the bag with an initial size
* Implement methods to add an item, remove an item, check if the bag is empty, and display the bag’s content.

**ShoppingBag**

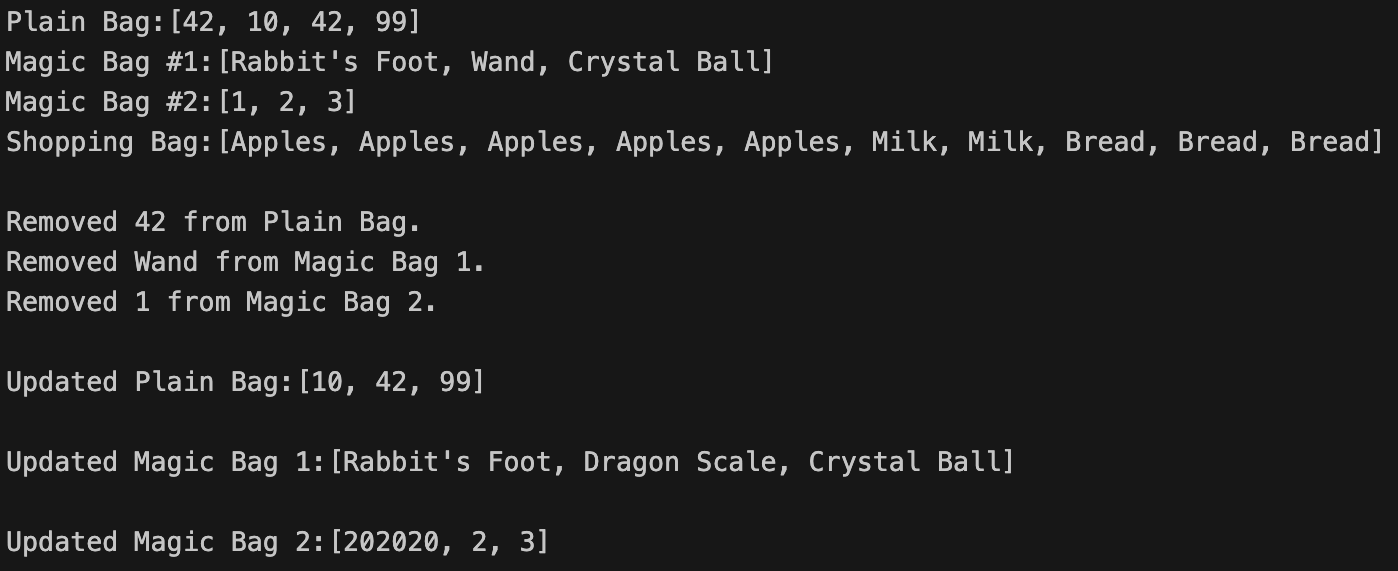
* Derived class from Bag
* Represents a shopping bag with quantities
* Implement methods to add an item with a specified quantity and display the quantity of each item

**MagicBag**

* Derived class from Bag.
* Like its parent class **Bag**, the **MagicBag** can hold items and allows for adding and removing them.
* What sets the **MagicBag** apart is its enchanting behavior. When you remove an item from the **MagicBag**, it doesn't leave an empty space. Instead, it magically replaces the item with another random magical item.
* It has a predefined list of magical items (defined as a static variable), and it selects a new magical item at random from this list to replace the removed item.
* Override removeItem() method to implement magical behavior

**Sample main.cpp:**

* You are given the **main.cpp** file. Please don’t make any modification in the main method in the main.cpp file. Use it as its.
* The sample run for this application is as follows:

****

**Hints**

***Vectors***

* In C++, a vector is a dynamic array that can grow or shrink in size as needed. To use vectors, include the **<vector>** header at the beginning of your C++ program.
* **Declaring a Vector:** Declare a vector by specifying the data type it will hold.

std::vector<int> myVector;

* **Initializing Vectors:** You can initialize a vector when declaring it, just like you would with an array:

std::vector<int> myVector = {1, 2, 3, 4, 5};

* **Adding Elements:** To add elements to a vector, use the **push\_back** method:

myVector.push\_back(6);

* **Accessing Elements:** Access elements in a vector using square brackets **[]** just like you would with an array:

int element = myVector[0];

* **Getting Size:** To get the size (number of elements) of a vector, use the **size** method:

int size = myVector.size();

* **Iterating through Vectors:** You can use loops to iterate through vector elements:

for (int i = 0; i < myVector.size(); i++) {

int element = myVector[i];

}

***Declaring static variable***

* In C++, you can declare a static template variable within a class or class template. This variable will be shared among all instances of the class or class template with the same template parameter.

template <class T>

class MyClass {

public:

static T myStaticVariable; *// Declaration*

};

*// Definition (outside the class definition)*

template <class T>

T MyClass<T>::myStaticVariable;

* You can initialize the static template variable outside of the class or class template definition. This initialization should be done for each template specialization:

*// For a specific specialization, e.g., int*

template <>

int MyClass<int>::myStaticVariable = 42;

***Random number generation***

* Include the following header files

#include <cstdlib> // For std::srand, std::rand

#include <ctime> // For std::time

* To ensure that the random number generator produces different sequences each time your program runs, you should seed it with a unique value.

std::srand(static\_cast<unsigned>(std::time(nullptr)));

* Once you've seeded the random number generator, you can generate random numbers using std::rand(). Here's how to generate a random integer within a specified range

int min = 1; // Minimum value

int max = 100; // Maximum value

int randomValue = min + (std::rand() % (max - min + 1));

**Important Notes**

1. You are **not allowed to modify main.cpp** implementation. Please use it as it is.
2. Before you submit your work, please make sure the entire folder works. Here is a sample command that you can perform a sanity check:

*Windows Users:*

yourfolder> g++ -o myProgram.exe -std=c++14 \*.cpp  
  
yourfolder>myProgram.exe

*Mac Users:*

yourfolder$ g++ -o myProgram -std=c++14 \*.cpp  
  
yourfolder$ ./myProgram

1. You are not required to handle the edge cases in this assignment.

**Submission**

You are asked to submit your work as a single zip file via CANVAS. Zip file will include all source codes including .h and .cpp files.

Please use the following file format while naming the zip file:

LastNameFirstnameX\_Y.zip where LastNameFirstname is your last name with the first letter in capital, followed by your first name with the first letter in capital; the X is the course code; the Y is the assignment #. (ex: SerceFatmaCS300\_1.zip)

**Rubric**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Rating** | **Description** |
| **Program Execution (40%)** | 40 | Program runs correctly |
| 20 | Program produces correct output half of the time |
| 10 | Program runs, but mostly incorrectly |
| 0 | Program doesn’t compile or run at all |
| **Requirement Satisfaction (30%)** | 30 | Program meets requirements completely and correctly |
| 15 | Many parts of the requirements no implemented |
| 5 | Program does not satisfy requirements |
| **Coding Style (20%)** | 20 | Well-formatted, understandable code |
| 10 | Code hard to follow in one reading |
| 5 | Incomprehensive code |
| **Comments (10%)** | 10 | Concise, meaningful, well-formatted comments |
| 5 | Partial, poorly written or poorly formatted comments |
| 3 | Unnecessary, incorrect, badly formatted comments |
| 0 | No comments at all |