Lab 5 Report

Group 25: Randy Hucker and Sam Graler

Instructions:

Code for the first 3 tasks can be found in the project folder “Lab 5 Tasks 1-3”. Task 4 is in a separate project “Lab 5 Task 4”, but that project still contains all the other necessary files. Both are complete visual studio projects, and can be compiled and run like normal.

- “Lab 5 Tasks 1-3” has a file named “Task2.cpp” which is the code from task 2 before exception handling was added. It is commented out using a block comment, but it is included in the submission in case output from task 2 specifically is to be tested.

Write a lab report including the following information:

a. A description of the objectives/concepts explored in this assignment including why you think they are important to this course and a career in CS and/or Engineering.

The concepts explored in this lab were templates and exceptions, and both are important to a career in computer science. First of all, custom exceptions are very useful in a wide range of applications. Exceptions are useful for dealing with human error, because many times users accidentally enter data that the program cannot store in the correct way. With exceptions, we can handle errors without the entire program crashing. Exceptions can also be used to handle errors triggered by conditions that cannot be controlled otherwise (like stack overflow and underflow). Exceptions allow us to gracefully deal with errors that would otherwise cause the program to terminate with an error message that may or may not be helpful. This kind of preventative thinking is important in computer science when system crashes can have catastrophic consequences.

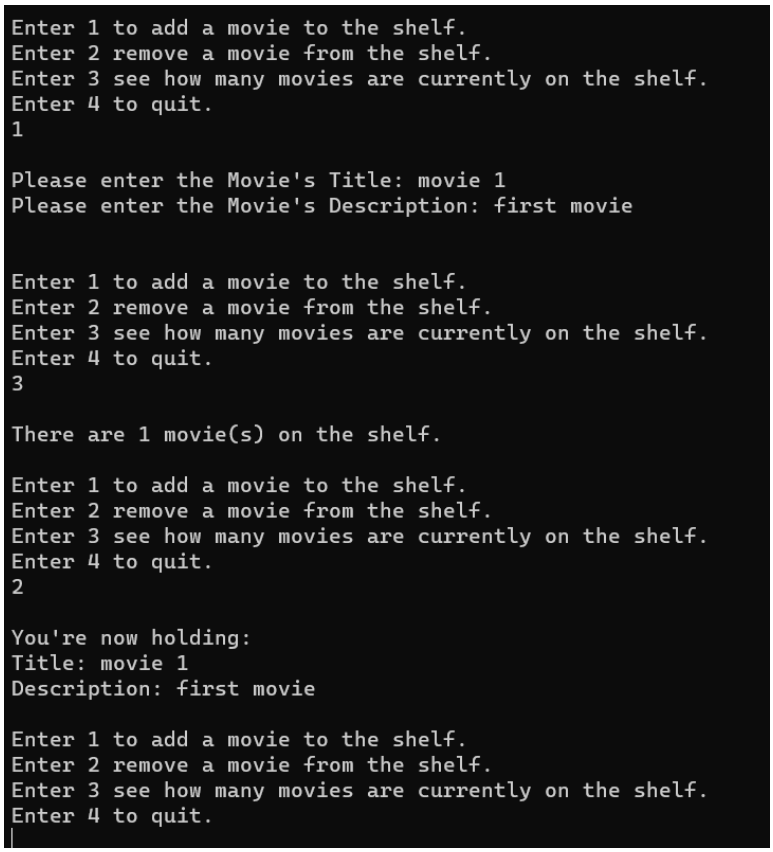
Templates allow us to reuse code for classes that can be implemented in different situations using different types. This saves a lot of time by eliminating code writing and debugging, and allows us to concentrate on the logic of the functions and not get bogged down by the specific types and all of that. Of course templates can’t be as specific as typed classes, but that is the point. This type of abstraction is an important way of thinking for an engineer, because dividing a problem into smaller parts and finding ways to reuse solutions is always better than tackling something all at once and then writing redundant solutions because you didn’t take the time to think through other possible applications.

b. The sections from each task indicated to be included in the lab report.

Task 1: 3. Include in the submission a description of how you designed your add and remove methods. This description should not be code or pseudo code. Complete this before moving on to task 2.

We designed the add and remove functions to act like those of a stack. There is a private variable in the MovieShelf class called MovieCount that represents the number of movies on the shelf, which also means it is the first available index in the array of Movies. For the add function, we add the movie pointer that is passed in, and then we increment the MovieCount so that we have the next available index for adding more movies (and keeping track of the count). For the remove function, we first decrement MovieCount so we have the index of the last movie added, and then we return the Movie to "remove" it. We don't actually delete the movie stored in that spot, but because the counter is incremented it is no longer accessible, and the next movie added to that index overwrites the one we removed.

Task 2: 3. Include in the lab report a screenshot(s) board of the output of your tests.



Task 3: 3. Include in the lab report a screenshot(s) of the output of a test. Include a discussion of the advantages of trapping an error in the class versus the calling function.

An advantage of trapping an error in the class versus the calling function is that when an exception is thrown, all objects in the scope of the calling function will be lost. Catching the exception in the class allows us to handle it without losing data because we catch it before it causes anything to be removed from scope.

Screenshots:

EmptyShelfException Catch:

```
Enter 1 to add a movie to the shelf.
Enter 2 remove a movie from the shelf.
Enter 3 see how many movies are currently on the shelf.
Enter 4 to quit.
3

There are 0 movie(s) on the shelf.

Enter 1 to add a movie to the shelf.
Enter 2 remove a movie from the shelf.
Enter 3 see how many movies are currently on the shelf.
Enter 4 to quit.
2

Caught EmptyShelfException
The Shelf Was Empty! You cannot remove a movie from an empty shelf.

Enter 1 to add a movie to the shelf.
Enter 2 remove a movie from the shelf.
Enter 3 see how many movies are currently on the shelf.
Enter 4 to quit.
```

FullShelfException Catch:

```
Enter 1 to add a movie to the shelf.
Enter 2 remove a movie from the shelf.
Enter 3 see how many movies are currently on the shelf.
Enter 4 to quit.
3

There are 10 movie(s) on the shelf.

Enter 1 to add a movie to the shelf.
Enter 2 remove a movie from the shelf.
Enter 3 see how many movies are currently on the shelf.
Enter 4 to quit.
1

Please enter the Movie's Title: 11th add attempt
Please enter the Movie's Description: 11th add attempt

Caught FullShelfException
The Shelf Was Full! You cannot add a movie to a full shelf.

Enter 1 to add a movie to the shelf.
Enter 2 remove a movie from the shelf.
Enter 3 see how many movies are currently on the shelf.
Enter 4 to quit.
```

Task 4: 5. Include in the lab report a screenshot(s) of the output of a test. Include a discussion of the advantages of using a template over specific types.

Templates allow us to reuse code for classes that can be implemented using many different types. This saves time, and makes debugging easier because there are fewer different versions of a class that has a very similar use. In our example, we originally made a movie shelf, but changed it to a template so we can now have a shelf of any object (theoretically). Now that this template exists, we don't have to reimplement the same logic for a videogame shelf, a book shelf, a tv show collection etc.

Screenshots:

In our test program, we made two instances of the entertainment collection class. One of which holds movies and the other holds videogames (the video game class is a copy of the movie class just with a different name). The following screenshots prove the template class works as intended regardless of the type of object it is storing.

Adding / Removing Objects Testing:

```
Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
1

Please enter the Movie's Title: movie 1
Please enter the Movie's Description: text


Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
2

Please enter the Game's Title: game 1
Please enter the Game's Description: text


Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
5

There are 1 movie(s) on the movie shelf.
There are 1 game(s) on the video game shelf.
```

```
Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
3

You're now holding:
Title: movie 1
Description: text

Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
4

You're now holding:
Title: game 1
Description: text

Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
5

There are 0 movie(s) on the movie shelf.
There are 0 game(s) on the video game shelf.
```

Exception Handling Tests:

```
5

There are 0 movie(s) on the movie shelf.
There are 0 game(s) on the video game shelf.

Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
3

Caught EmptyShelfException
The Shelf Was Empty! You cannot remove a movie from an empty shelf.

Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
4

Caught EmptyShelfException
The Shelf Was Empty! You cannot remove a movie from an empty shelf.

Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
```

```
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
5

There are 10 movie(s) on the movie shelf.
There are 10 game(s) on the video game shelf.

Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
1

Please enter the Movie's Title: attempt 11
Please enter the Movie's Description: 11

Caught FullShelfException
The Shelf Was Full! You cannot add a movie to a full shelf.

Movie shelf and Video Game shelf are both instances of EntertainmentCollection
Enter 1 to add a movie to the Movie shelf.
Enter 2 to add a video game to the video game shelf.
Enter 3 remove a movie from the movie shelf.
Enter 4 remove a video game from the video game shelf.
Enter 5 see how many items are currently on the shelves.
Enter 6 to quit.
2

Please enter the Game's Title: attempt 11
Please enter the Game's Description: 11

Caught FullShelfException
The Shelf Was Full! You cannot add a movie to a full shelf.
```