Data Structures

Lab 5

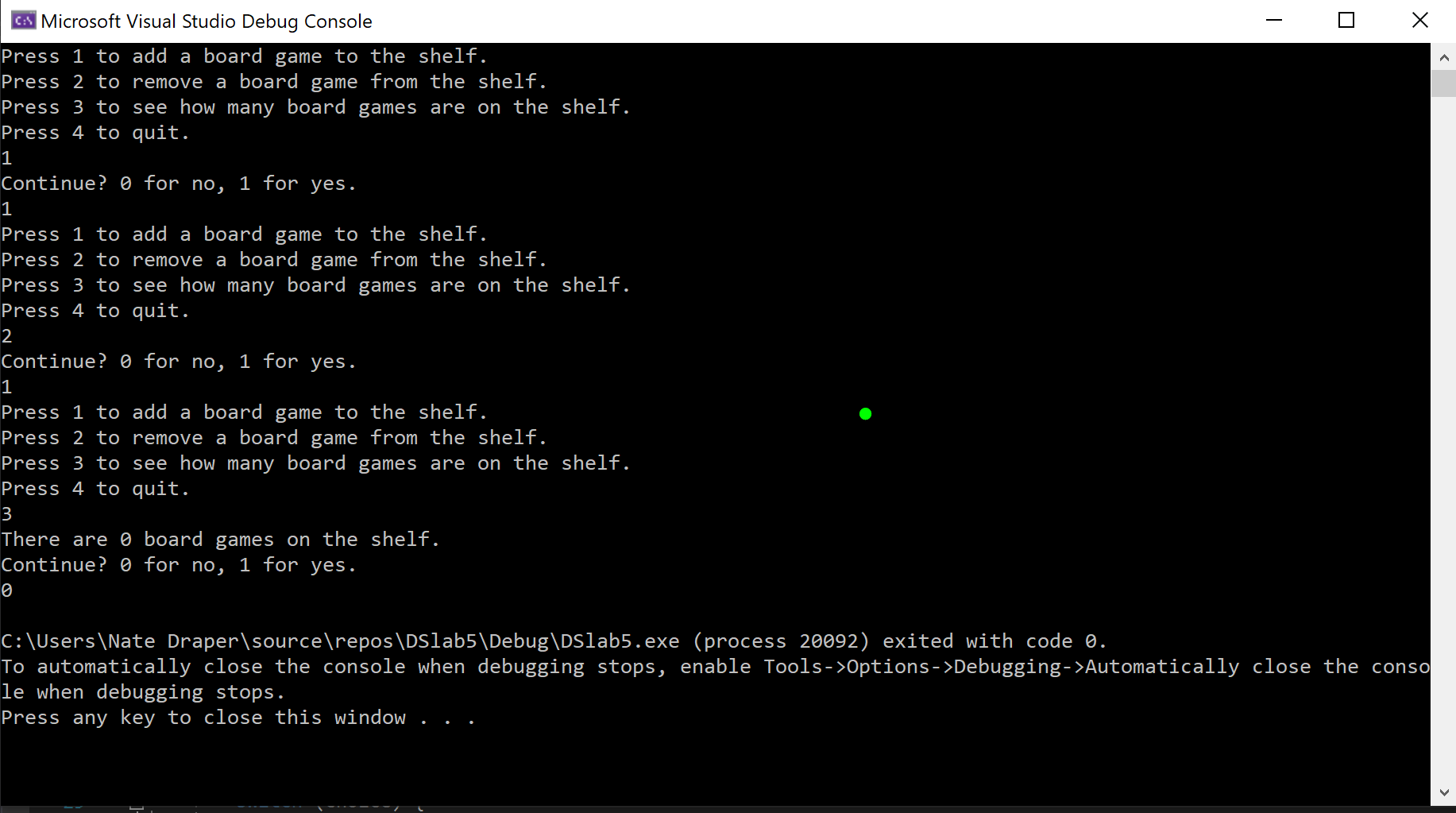
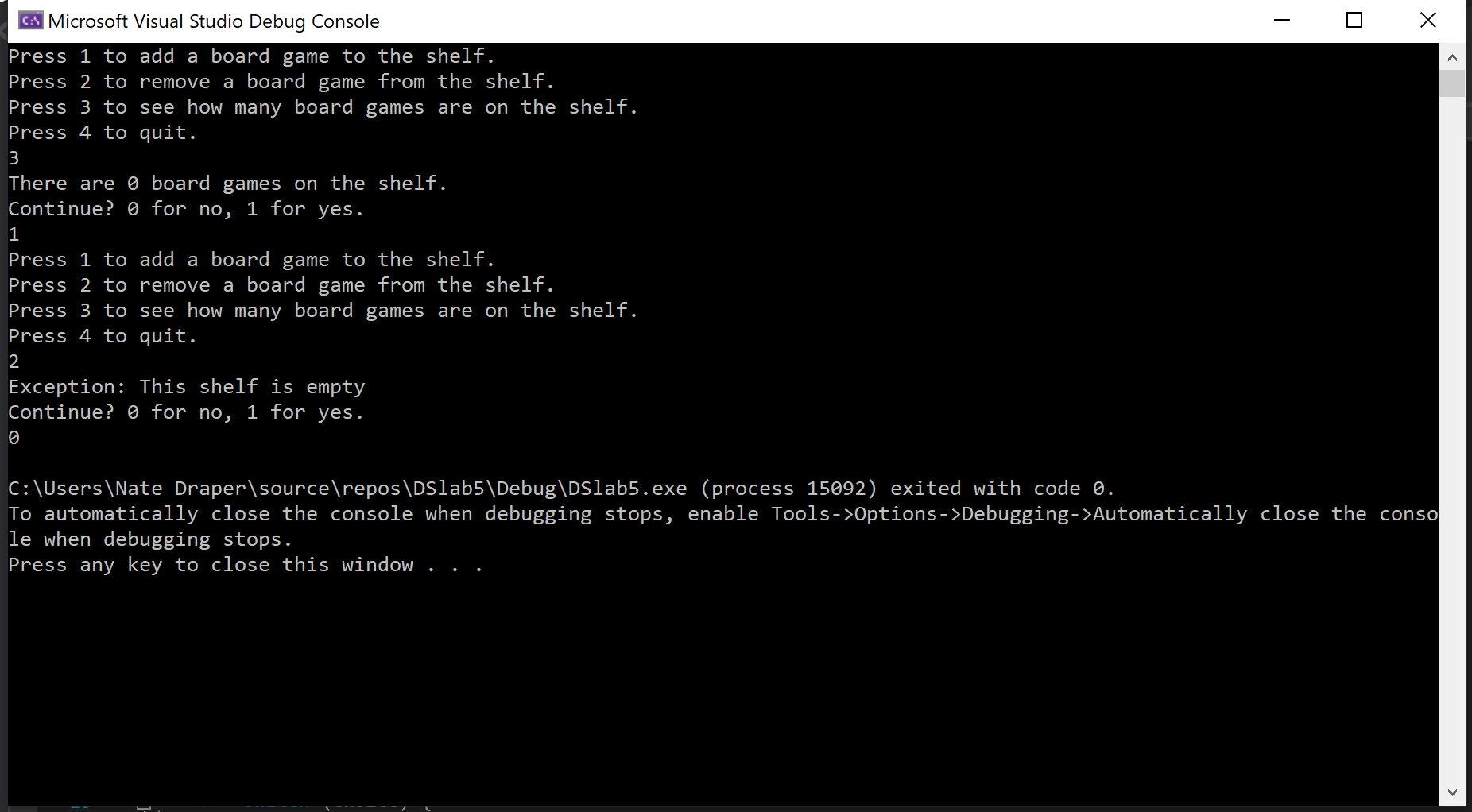
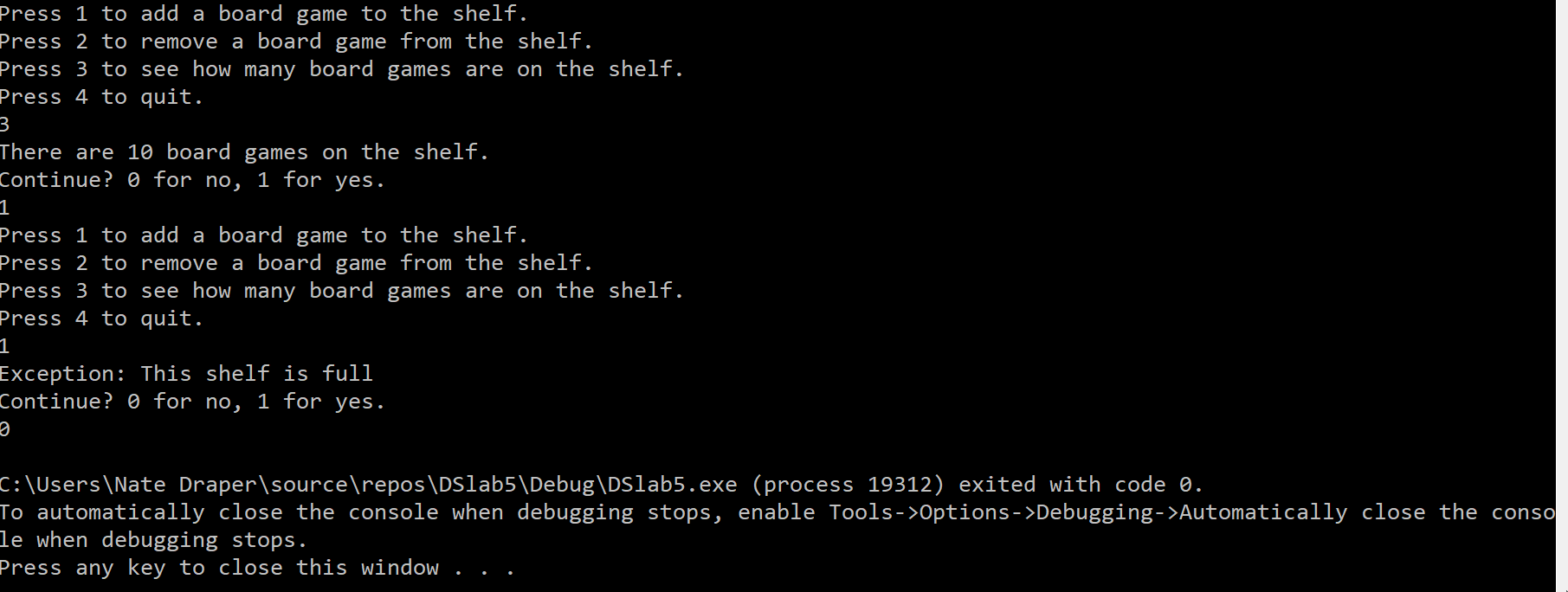
Group 11

Nathan Draper

Spencer Jones

Michael Reagan

Report:

1. The objective of this lab was to firstly understand how to use exceptions in a way that would allow for seamless data flow that would not produce errors. Exceptions are useful when trying to trap data that would otherwise cause problems throughout your program from invalid user input. Exceptions are essentially a “catch-all” for data that would otherwise need copious amounts of if-statements to weed out invalid data. In the field of computer engineering, exceptions are important for debugging, as well as data organization.  
     
   The second objective of this assignment was to understand and implement class templates. Templates are important as they allow the programmer to write generic classes with abstract data types. This allows the programmer to pick and choose what attributes to add in addition to the generic ones of the template. By doing this we not only save memory, but we can also keep our data organized without the need to create a whole new class. Essentially templates allow us to group like, class attributes.
2. **Task 1: Include in the submission a description of how you designed your add and remove methods:**  
     
   The way the add and remove methods were designed similarly to push and pop methods for a stack. Because the game objects were assumed to be fungible, it didn’t matter in what order the objects were placed in. This means that when an object is added to the shelf, it is added to the end of the list so that no objects on the shelf need to be shifted. A size variable keeps track of where the end of the list is, so that the list does not need to be searched to find the end. The remove method works similarly, by removing objects from the end of the list. The size variable is updated appropriately after adding and removing from the shelf.  
     
   Task 2 output:  
     
     
     
   **Task 3: Include a discussion of the advantages of trapping an error in the class versus the calling function:**  
     
   An advantage to trapping an error inside the class, is that you know exactly where the error is coming from. If the error is caught inside the calling function, it is not clear what line of code was causing the error and can make the error more difficult to fix. When the error is trapped in the class, error handling code only needs to be written once. If the error is handled in the calling function, every time the function is called which could contain an exception, the error handling code needs to be rewritten.  
     
     
     
     
     
   **Task 4: Include a discussion of the advantages of using a template over specific types:**

The biggest advantage of using a template over a specific type is that the code is reusable. That means that the code does not need to be rewritten to make the code work with a different type. This reduces the amount of code that needs to be written and can help make the code more readable. The code becomes more readable because there do not need to be two functions that have the same purpose but with different names. Templates also make the code easier to maintain. If there is a problem with the function, it only needs to be changed in one place. If there were multiple functions created for different types, then it could introduce more chances for errors.

Task 4 output:

