Amei Ayuel

CS 303
Assignment 1 ReadMe
Due 9/9/24

Project Overview

This project implements basic operations on a one-dimensional array in C++. The operations include finding an element, modifying an element, adding a new element, and either replacing or removing an element from the array. The code also includes basic exception handling.

How to Run the Code

Prerequisites

- A C++ compiler (e.g., g++, clang++)
- A text editor or IDE

Step-by-Step Instructions

1. **Copy the Code**: Copy the provided code for `main.cpp`, `array_operations.h`, and `array_operations.cpp` into your text editor or IDE.
2. **Compile the Code**:
   - If using the command line, navigate to the directory containing your files and run:
     bash

     Copy code

     ```
     g++ main.cpp array_operations.cpp -o array_operations
     ```
   - This command compiles the code and creates an executable named `array_operations`.
3. **Run the Executable**:
   - On the command line, run the executable:
     bash
     Copy code

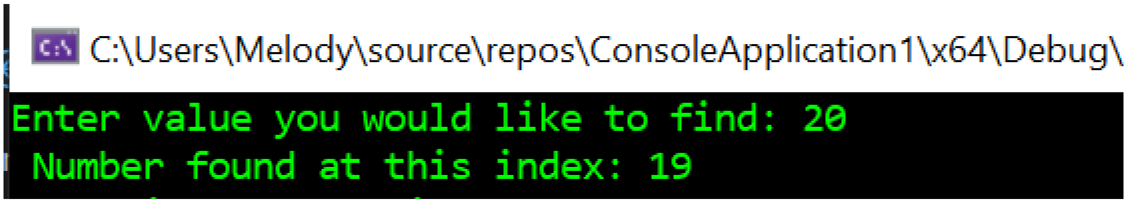
     ```
     ./array_operations
     ```
   - The program will perform several operations on the array and display the results in the terminal.

Output Examples

1. **Finding an Element in the Array**:

    - The program checks for the presence of an element and returns its index if found.

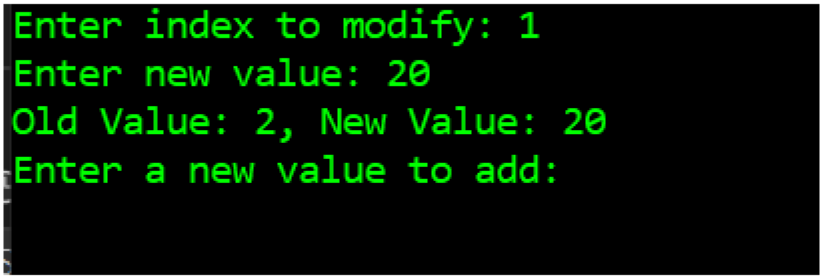
Example Output:



```
Enter value you would like to find: 20
 Number found at this index: 19
```

2. **Modifying an Element with Exception Handling**:
    - The program modifies an element at a specified index, handling errors if the index is out of bounds.

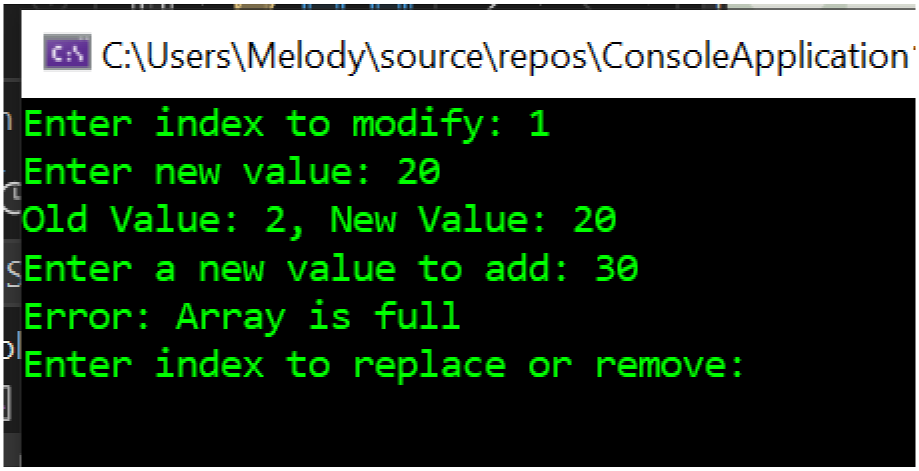
Example Output:



```
Enter index to modify: 1
Enter new value: 20
Old Value: 2, New Value: 20
Enter a new value to add:
```

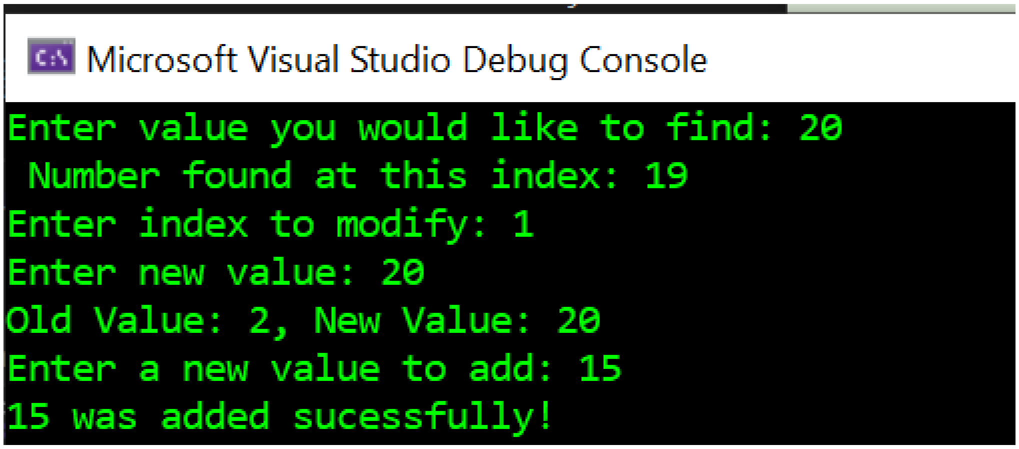
3. **Adding a New Element with Exception Handling**:
    - The program adds a new element to the end of the array and handles cases where the array is full.

Example Output when array is full



```
Enter index to modify: 1
Enter new value: 20
Old Value: 2, New Value: 20
Enter a new value to add: 30
Error: Array is full
Enter index to replace or remove:
```

Example of when a new element was added. Changed the max size from 100 to 200.

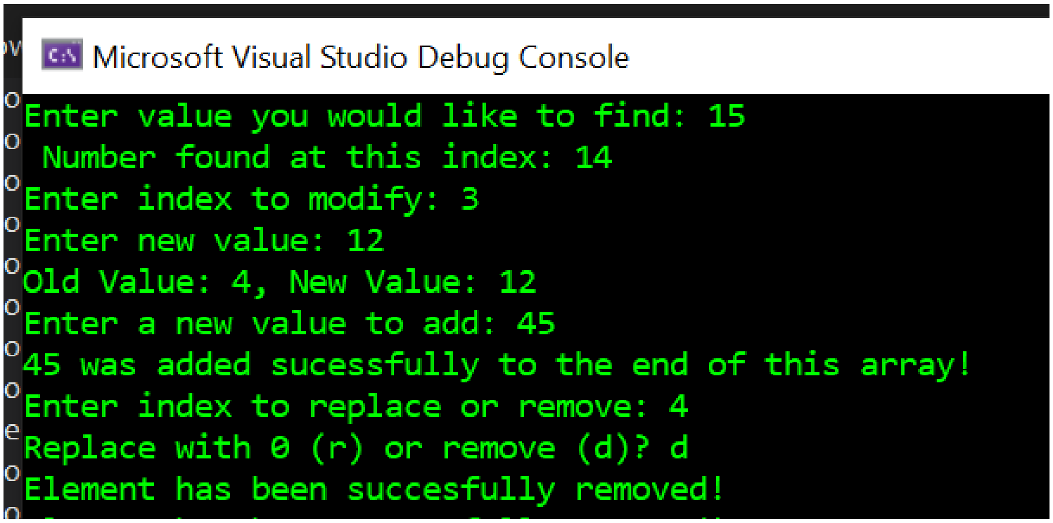


```
Enter value you would like to find: 20
 Number found at this index: 19
Enter index to modify: 1
Enter new value: 20
Old Value: 2, New Value: 20
Enter a new value to add: 15
15 was added sucessfully!
```

4. **Replacing/Removing an Element**:
    - The program replaces an element with 0 or removes it, adjusting the array accordingly.

Example Output:

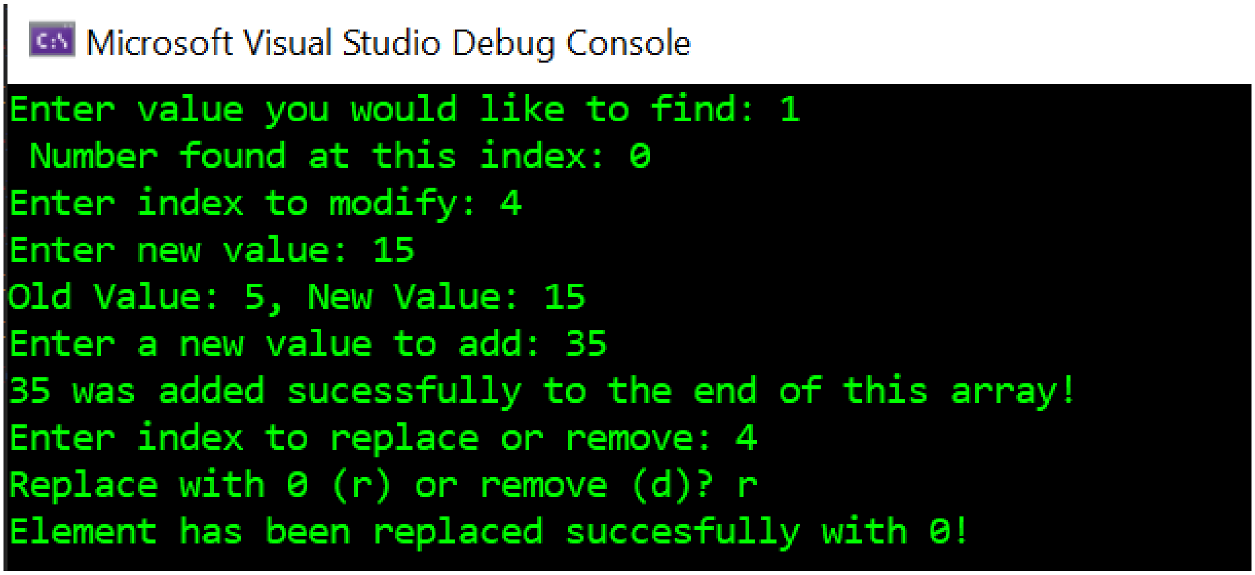
Remove



```
Enter value you would like to find: 15
 Number found at this index: 14
Enter index to modify: 3
Enter new value: 12
Old Value: 4, New Value: 12
Enter a new value to add: 45
45 was added sucessfully to the end of this array!
Enter index to replace or remove: 4
Replace with 0 (r) or remove (d)? d
Element has been succesfully removed!
```

Example output:

Replace



```
Enter value you would like to find: 1
 Number found at this index: 0
Enter index to modify: 4
Enter new value: 15
Old Value: 5, New Value: 15
Enter a new value to add: 35
35 was added sucessfully to the end of this array!
Enter index to replace or remove: 4
Replace with 0 (r) or remove (d)? r
Element has been replaced succesfully with 0!
```

Key Functions Explained

- `findElement`: Searches the array for a specific element and returns its index.
- `modifyElement`: Modifies the value of an element at a given index with exception handling for out-of-bounds indices.
- `addElement`: Adds a new element to the end of the array with exception handling for a full array.
- `replaceOrRemoveElement`: Replaces an element with 0 or removes it by shifting elements left.