**Array Operations Program with Exception Handling**

**Overview**

This C++ program reads integers from a file, lets you perform multiple array operations, and uses strong exception handling to showcase essential array manipulation and error-handling techniques.

**Features**

- \*\*File Input\*\*: Reads integers from an input file into a one-dimensional array

- \*\*Search Function\*\*: Locates elements in the array and returns their index

- \*\*Modify Function\*\*: Updates array elements with exception handling for invalid indices

- \*\*Add Function\*\*: Appends new elements to the array with exception handling for full arrays

- \*\*Remove Function\*\*: Deletes elements from specified indices

- \*\*Display Function\*\*: Shows current array contents

- \*\*Exception Handling\*\*: Custom exceptions for error management

**Files Required**

1. Source Code Files

- ArrayOperations.cpp - Main program implementation

- ArrayOperations.h - Header file with class definitions

2. Input File

- A1input.txt - Contains integers to be loaded into the array

3. Sample Input File Content

1 2 3 4 5 6 7 8 9 10

11 12 13 14 15 16 17 18 19 20

21 22 23 24 25 26 27 28 29 30

31 32 33 34 35 36 37 38 39 40

41 42 43 44 45 46 47 48 49 50

51 52 53 54 55 56 57 58 59 60

61 62 63 64 65 66 67 68 69 70

71 72 73 74 75 76 77 78 79 80

81 82 83 84 85 86 87 88 89 90

91 92 93 94 95 96 97 98 99 100

**Functions Overview**

- readFromFile(): Loads data from input file

- searchElement(): Linear search implementation

- modifyElement(): Updates array element with exception handling

- addElement(): Appends element with exception handling

- removeElement(): Removes element and shifts array

- displayArray(): Prints array contents

- getSize(): Returns current array size

**Exception Classes**

- Custom exception classes inherit from “std::exception”

- Provide descriptive error messages

- Support both default and parameterized constructors

**Exception Handling Features**

Custom Exception Classes

1. \*\*InvalidIndexException\*\*: Thrown when accessing invalid array indices

2. \*\*ArrayFullException\*\*: Thrown when attempting to add elements to a full array

Exception Safety

- Modify and Add functions use try-catch blocks

- Descriptive error messages for debugging

- Program continues running after catching exceptions

**Technical Specifications**

Array Limitations

- \*\*Maximum Size\*\*: 1000 elements

- \*\*Data Type\*\*: Integer (int)

- \*\*Index Range\*\*: 0 to (current\_size - 1)

**Input File Requirements**

- Space-separated integers

- Can span multiple lines

- Maximum 1000 integers will be read

**Troubleshooting**

Common Issues

1. File Not Found Error\*\*

Error: Could not open file A1input.txt

Solution: Ensure “A1input.txt” exists in the same directory as the executable

2. Compilation Errors

- Ensure C++11 or later standard

- Include all necessary headers: <iostream>, <fstream>, <stdexcept>, <string>

3. Exception Not Caught

- Verify try-catch blocks are properly implemented

- Check exception class inheritance from “std::exception”

**Running the Program**

Step 1: Prepare Input File

Ensure `A1input.txt` is in the same directory as your executable file. The file should contain space-separated integers.

Step 2: Execute the Program

Step 3: Program Startup

The program will automatically:

1. Read data from `A1input.txt`

2. Display the number of elements loaded

3. Show the initial array contents

4. Present the main menu

Program Menu Options

========== Array Operations Menu ==========

1. Display Array

2. Search for an Element

3. Modify an Element (with Exception Handling)

4. Add an Element (with Exception Handling)

5. Remove an Element

6. Exit

===========================================

A screenshot of a computer

AI-generated content may be incorrect.

Usage Examples

1. Display Array

- Select option “1”

- Shows all current array elements

A screenshot of a computer

AI-generated content may be incorrect.

2. Search for Element

- Select option “2”

- Enter the value to search for

- Program returns the index if found, or "not found" message

A screenshot of a computer

AI-generated content may be incorrect.

Example:

Enter the value to search for: 25

Element 25 found at index 24

A screenshot of a computer

AI-generated content may be incorrect.

3. Modify Element (with Exception Handling)

- Select option “3”

- Enter the index of element to modify

- Enter the new value

- Program shows old and new values or catches exceptions for invalid indices

Example - Success:

Enter the index of element to modify: 5

Enter the new value: 999

Element at index 5 is modified successfully.

Old value: 6, New value: 999

A screenshot of a computer

AI-generated content may be incorrect.

Example - Exception:

Enter the index of element to modify: 150

Enter the new value: 999

Exception caught: Invalid index: 150. Index should be between 0 and 99

A screenshot of a computer

AI-generated content may be incorrect.

4. Add Element (with Exception Handling)

- Select option “4”

- Enter the value to add

- Program adds to end of array or catches exception if array is full

Example - Success:

Enter the value to add: 101

Element 101 added successfully at index 100

Example - Exception:

Enter the value to add: 500

Exception caught: Array is full of 1000 elements. Cannot add more elements.

A screenshot of a computer

AI-generated content may be incorrect.

5. Remove Element

- Select option “5”

- Enter the index of element to remove

- Program removes elements and shifts remaining elements

Example:

Enter the index of element to remove: 10

Element 11 removed successfully from index 10

A screenshot of a computer

AI-generated content may be incorrect.