LAB ASSIGNMENT 02

Name :Abhi Mehta
Branch : IT
Rollno:221080001

Experiment: Write a test program to create your own long integer array class to perform operations on array such as insert, find, delete, display etc.

Description of different Java constructs you used for the implementation.

Every class has a constructor that is used to initialize its objects. A custom constructor 'public MyLongArray(int size)' is Used for creating an instance of the MyLongArray class.

Class Diagram :

Program Code :

```
import java.util.*;
/**
* This class provides utility functions to perform various operations on an array
* such as creating, searching, inserting, deleting, and displaying its elements.
*
* @author Abhi Mehta
* @version 1.0.0
* @since
17-09-2023
*/
/**
```

```
* Class for array of type long with added functionalities
*/
class MyLongArray {
private long[] a;
private int currentIndex;
/**
* Constructor to initialize the array
*
* @param size - The size of the array
*/
public MyLongArray(int size) {
a = new long[size];
currentIndex = 0;
}
/**
* Insert element at end
*
* @param value- The value of the element to be inserted
*/
public void insert(long value) {
try{
a[currentIndex++]=value;
}
catch(Exception e){
System.out.println("Array is: " +a);
}
}
/**
* Finding element using value
*
* @param searchKey- The value of the element to be found
* @return - returns index if found, else returns -1
*/
public int find(long searchKey) {
for (int i = 0; i < currentIndex; i++) {
if (a[i] == searchKey) {
return i;
}
}
return -1;
}
/*** Function to get value of element from index
*
* @param index- The index of the element of which value needs to found
```

```
* @return - The value of element is returned if found else -1
*/
public long getElem(int index) {
if (index >= 0 && index < currentIndex) {
return a[index];
} else {
System.out.println("Index out of bounds of array !!");
return -1;
}
}
/**
* Function to delete element by value
*
* @param value - The value of element to be deleted
* @return - returns true if element was deleted and false if not
*/
public boolean delete(long value) {
int index = find(value);
if (index != -1) {
for (int i = index; i < currentIndex - 1; i++) {
a[i] = a[i + 1];
}
currentIndex--;
return true;
}
return false;
}
/**
* Function to display the entire array
*
*/
public void display() {
for (int i = 0; i < currentIndex; i++) {
System.out.print(a[i] + " ");
}
System.out.println();
}
/**
* Function to delete dup elements using value
*
* @param value- The value of elements, duplicates of which are to be deleted
* @return - The number of elements deleted
*/
public int dupDelete(long value) {
```

```
int count = 0;
for (int i = 0; i < currentIndex; i++) {
if (a[i] == value) {
count++;
if (count > 1) {
deleteAt(i);
i--;
}
}
}return count - 1;
}
/**
* Function to insert element in middle of array
*
* @param index - The index at which the element is to be inserted
*/
public void insert(int index, long value) {
if (index < 0 || index > currentIndex) {
System.out.println("Index out of bounds of array !!");
return;
}
if (currentIndex == a.length) {
System.out.println("Array is full !!");
return;
}
for (int i = currentIndex; i > index; i--) {
a[i] = a[i - 1];
}
a[index] = value;
currentIndex++;
}
/**
* Function to delete element by index
*
* @param index - The index of the element which is to be deleted
* @return - returns the value of element deleted or -1 if index out of bounds
*/
public long deleteAt(int index) {
if (index >= 0 && index < currentIndex) {
long temp = a[index];
for (int i = index; i < currentIndex - 1; i++) {
a[i] = a[i + 1];
}
currentIndex--;
```

```
return temp;
} else {
System.out.println("Index out of bounds of array !!");
return -1;
}
}
}
public class Array_Operations {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
boolean loop = true;
boolean end = false;
System.out.print("\nEnter size of array: ");
int size = sc.nextInt();
MyLongArray arr = new MyLongArray(size);//Creation of array
System.out.println("Array is created !!");
while (loop) {
//Menu for array
int op;
System.out.println("\n***** MENU *****");
System.out.println("1. Insert an element in array at end");System.out.println("2. Find an array
element");
System.out.println("3. Get Array element by it's index");
System.out.println("4. Delete an Array element");
System.out.println("5. Display the array");
System.out.println("6. Delete duplicate elements of Array");
System.out.println("7. Insert element at a particular index");
System.out.println("8. Delete element at a particular index");
System.out.println("9. Exit");
System.out.println("\nEnter your Choice : ");
op = sc.nextInt();
end = false;
while(!end)
switch (op) {
case 1:
System.out.print("\nEnter an element to insert in Array: ");
long value = sc.nextLong();
arr.insert(value);
end = true;
break;
case 2:
System.out.print("\nPlease enter value of element to be found: ");
long value1 = sc.nextLong();
int foundIndex = arr.find(value1);
```

```
if (foundIndex != -1) {
System.out.println("Array Element found at index: " + foundIndex);
} else {
System.out.println("Element not found.");
}
end = true;
break;
case 3:
System.out.print("\nEnter array element Index: ");
int index = sc.nextInt();
long element = arr.getElem(index);
if (element != -1) {
System.out.println("The array element present at "+index+" is : "
+element);
}
end = true;
break;
case 4:
System.out.print("\nEnter value of array element to delete: ");
value = sc.nextLong();
if (arr.delete(value)) {
System.out.println("Element deleted from the Array !!");
} else {
System.out.println("Element not found !!");
}
end = true;
break;
case 5:
System.out.println("Array is : ");arr.display();
end = true;
break;
case 6:
System.out.print("\nPlease input value of element to delete duplicates of:
");
value = sc.nextLong();
int deletedCount = arr.dupDelete(value);
System.out.println("Duplicates element deleted: " + deletedCount);
end = true;
break;
case 7:
System.out.print("\nPlease Enter index of element to be inserted: ");
index = sc.nextInt();
System.out.print("\nPlease Enter Value of element to be inserted: ");
value = sc.nextLong();
```

```
arr.insert(index, value);
System.out.println("Element inserted");
end = true;
break;
case 8:
System.out.print("Enter index of element to be deleted: ");
index = sc.nextInt();
long deletedElement = arr.deleteAt(index);
if (deletedElement != -1) {
System.out.println("Element deleted: " + deletedElement);
}
end = true;
break;
case 9:
loop = false;
end = true;
break;
default:
System.out.println("Invalid input !@!@");
System.out.println("|| Enter Appropriate Number ||");
end = true;
break;
}
}
}
}
```

Output :

```
Enter size of array: 4
Array is created !!

***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
1

Enter an element to insert in Array: 5
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
2

Please enter value of element to be found: 5
Array Element found at index: 0
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
1

Enter an element to insert in Array: 3
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
1

Enter an element to insert in Array: 4
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
4

Enter value of array element to delete: 5
Element deleted from the Array !!
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
5
Array is :
3 4
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
3

Enter array element Index: 1
The array element present at 1 is : 4
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
7

Please Enter index of element to be inserted: 2

Please Enter Value of element to be inserted: 3
Element inserted
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
5
Array is :
3 4 3
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
6

Please input value of element to delete duplicates of:3
Duplicates element deleted: 1
```

```
***** MENU *****
1. Insert an element in array at end
2. Find an array element
3. Get Array element by it's index
4. Delete an Array element
5. Display the array
6. Delete duplicate elements of Array
7. Insert element at a particular index
8. Delete element at a particular index
9. Exit

Enter your Choice :
5
Array is :
3 4
```

Type of errors you encountered and the way you fixed it.

1. Array Index Out of Bound:

Error Scenario: This occurs if we try to insert more elements than the array size or try to access an invalid index.

Fix: You have already added some checks, but a more comprehensive strategy would involve thorough checks before array manipulations and providing feedback to the user about the issue.

2. InputMismatchException:

Error Scenario: When the user is expected to enter a number but enters a non-numeric value.

Fix: Add try-catch around the Scanner's nextInt() and nextLong() calls to handle incorrect input. Prompt the user again for the correct input.

Learning lessons from the experiment :

1. Importance of Documentation:

The program begins with clear comments and JavaDoc-style documentation. This emphasizes the significance of well-documented code, making it more maintainable and understandable for other developers.

2. Encapsulation:

The use of a separate MyLongArray class to encapsulate array operations is a good practice. It showcases how Object-Oriented Programming (OOP) concepts like encapsulation can help in structuring the code logically.

3. Error Handling:

The program has incorporated some basic error-handling mechanisms, teaching us the necessity of anticipating and managing potential errors to provide a smooth user experience.

Conclusion:

In this experiment we implemented a menu-driven application for performing operations on an array of long integers. In conclusion, the program serves as a valuable learning tool. It offers insights into good programming practices, areas for improvement, and the iterative nature of software development.