UCS1313 Object Oriented Programming using Java Lab
AY: 2021-22

Name: Krithika Swaminathan
Roll No.: 205001057

Assignment 8 – Collection Framework

Q1: Write a program to perform string operations using ArrayList.

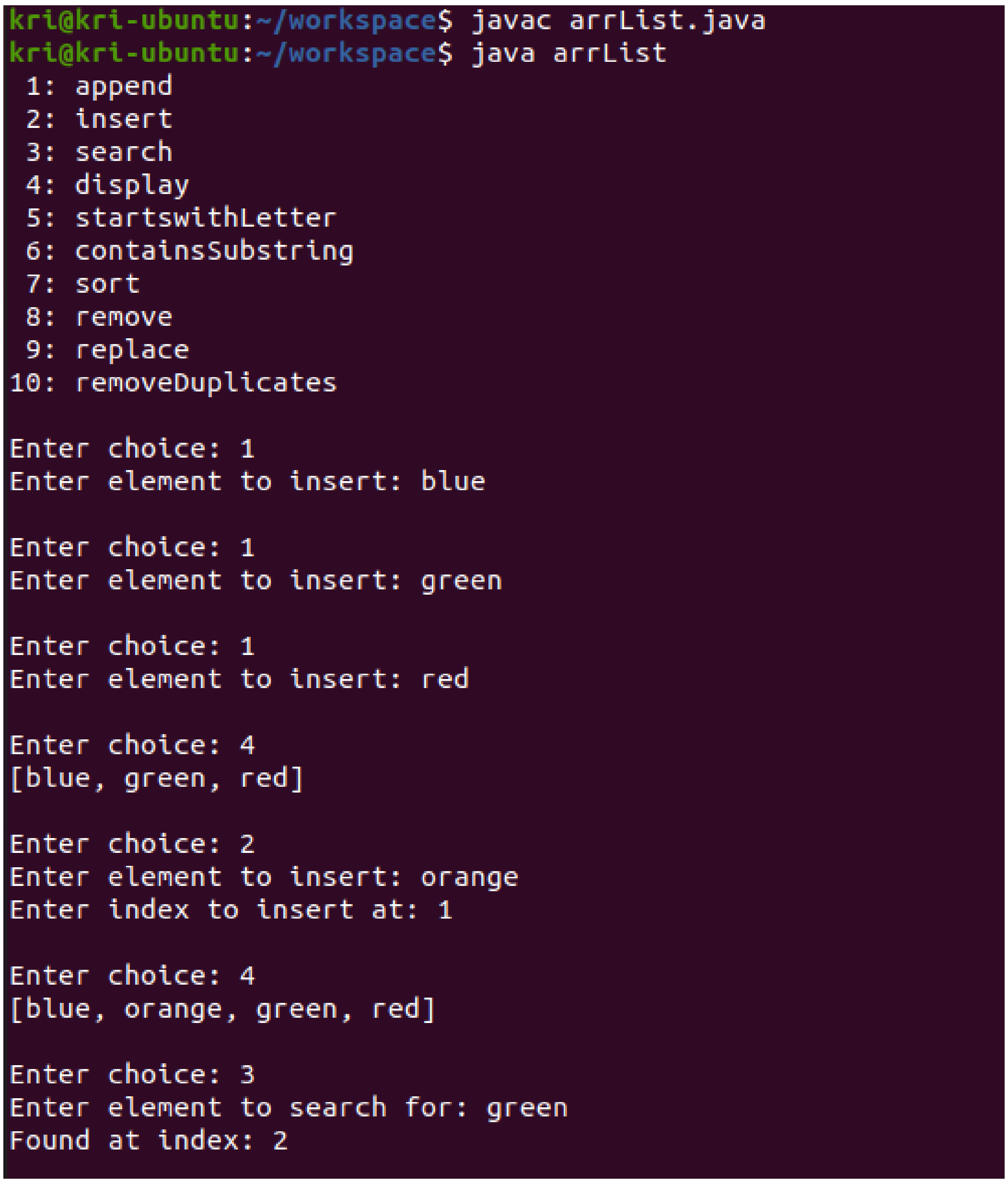
Code:

```
import java.util.*;

public class arrList {
	public static void main (String arg[]) {
		Scanner sc = new Scanner(System.in);

		ArrayList<String> alist = new ArrayList<String>();
		int ch;

		System.out.println(" 1: append\n 2: insert\n 3: search\n 4: display\n 5: startswithLetter\n 6: containsSubstring\n 7: sort\n 8: remove\n 9: replace\n10: removeDuplicates\n");
		System.out.print("Enter choice: ");
		ch = sc.nextInt();

		while (ch!=0) {
			switch (ch) {
				case 1: {
					System.out.print("Enter element to insert: ");
					String el = sc.next();
					alist.add(el);
					break;
					}
				case 2: {
					System.out.print("Enter element to insert: ");
					String el = sc.next();
					System.out.print("Enter index to insert at: ");
					int pos = sc.nextInt();
					alist.add(pos,el);
					break;
					}
				case 3: {
					System.out.print("Enter element to search for: ");
					String el = sc.next();
					int found = alist.indexOf(el);
					System.out.println("Found at index: "+found);
					break;
					}
				case 4: {
					System.out.println(alist);
					break;
					}
				case 5: {
					System.out.print("Enter letter to start with: ");
```

```
                        String letter = sc.next();
                        for (String element : alist) {
                                if (element.startsWith(letter)) {
                                        System.out.print(element+" ");
                                        }
                                }
                        System.out.println();
                        break;
                        }
                case 6: {
                        System.out.print("Enter substring to search for: ");
                        String substring = sc.next();
                        for (String element : alist) {
                                if (element.contains(substring)) {
                                        System.out.print(element+" ");
                                        }
                                }
                        System.out.println();
                        break;
                        }
                case 7: {
                        Collections.sort(alist);
                        break;
                        }
                case 8: {
                        System.out.print("Enter element to remove: ");
                        String el = sc.next();
                        alist.remove(el);
                        break;
                        }
                case 9: {
                        System.out.print("Enter element to replace: ");
                        String el1 = sc.next();
                        System.out.print("Enter element to replace with: ");
                        String el2 = sc.next();
                        alist.set(alist.indexOf(el1),el2);
                        break;
                        }
                case 10: {
                        alist = removeDuplicates(alist);
                        break;
                        }
                default: System.out.println("Invalid choice");
                }
        System.out.println();

        System.out.print("Enter choice: ");
        ch = sc.nextInt();
        }
```

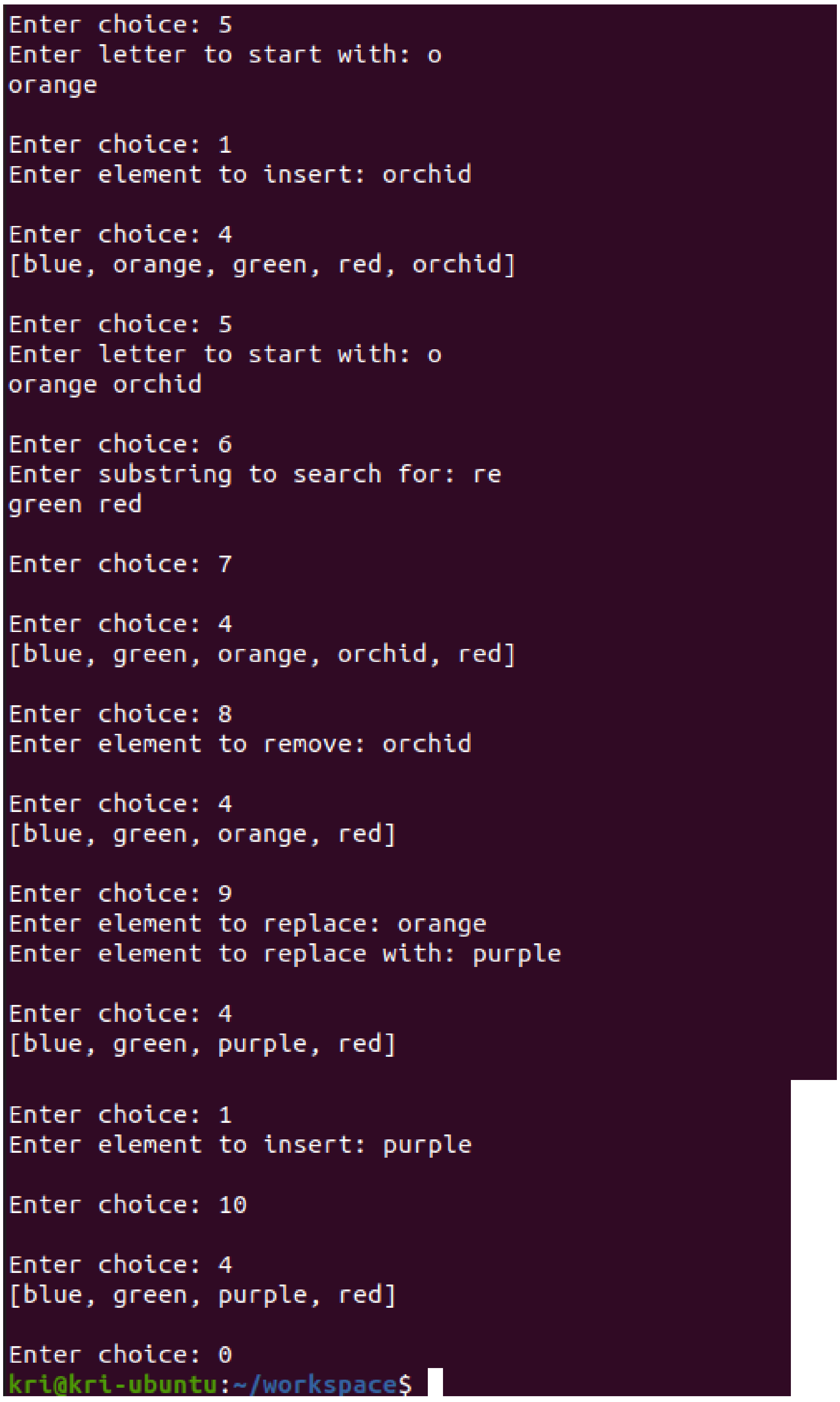
```
		}

	public static <T> ArrayList<T> removeDuplicates (ArrayList<T> list) {
		ArrayList<T> newList = new ArrayList<T>();
		for (T element : list) {
			if (!newList.contains(element)) {
				newList.add(element);
				}
			}
		return newList;
		}
	}
```

Output:



```
kri@kri-ubuntu:~/workspace$ javac arrList.java
kri@kri-ubuntu:~/workspace$ java arrList
 1: append
 2: insert
 3: search
 4: display
 5: startswithLetter
 6: containsSubstring
 7: sort
 8: remove
 9: replace
10: removeDuplicates

Enter choice: 1
Enter element to insert: blue

Enter choice: 1
Enter element to insert: green

Enter choice: 1
Enter element to insert: red

Enter choice: 4
[blue, green, red]

Enter choice: 2
Enter element to insert: orange
Enter index to insert at: 1

Enter choice: 4
[blue, orange, green, red]

Enter choice: 3
Enter element to search for: green
Found at index: 2
```

Name: Krithika Swaminathan
Roll No.: 205001057



```
Enter choice: 5
Enter letter to start with: o
orange

Enter choice: 1
Enter element to insert: orchid

Enter choice: 4
[blue, orange, green, red, orchid]

Enter choice: 5
Enter letter to start with: o
orange orchid

Enter choice: 6
Enter substring to search for: re
green red

Enter choice: 7

Enter choice: 4
[blue, green, orange, orchid, red]

Enter choice: 8
Enter element to remove: orchid

Enter choice: 4
[blue, green, orange, red]

Enter choice: 9
Enter element to replace: orange
Enter element to replace with: purple

Enter choice: 4
[blue, green, purple, red]

Enter choice: 1
Enter element to insert: purple

Enter choice: 10

Enter choice: 4
[blue, green, purple, red]

Enter choice: 0
kri@kri-ubuntu:~/workspace$
```

Name: Krithika Swaminathan
Roll No.: 205001057

Q2: Write a program to get two integer arraylist and perform the operations of merging, union, intersection and comparison.

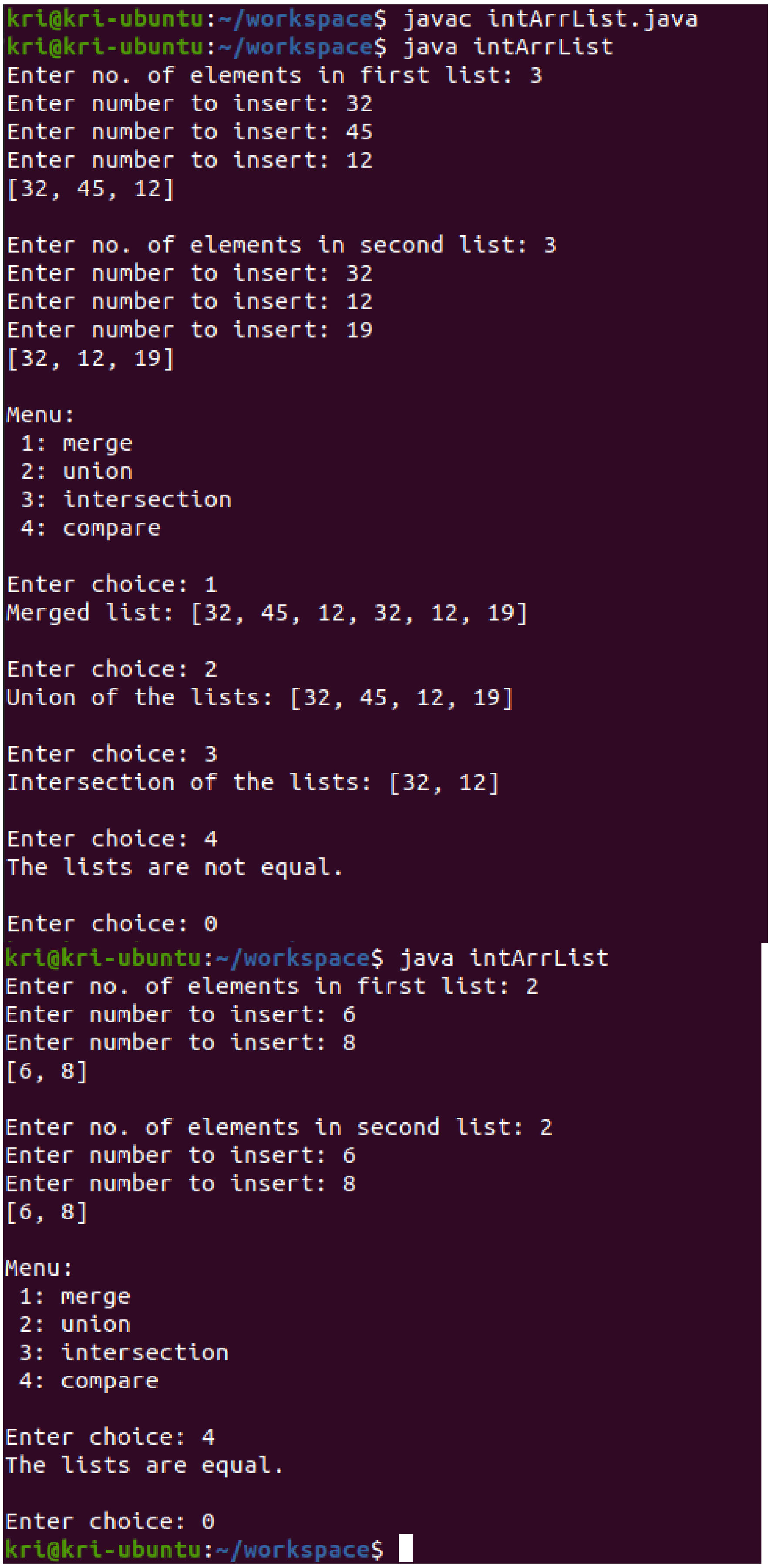
Code:

```
import java.util.*;

public class intArrList {
	public static void main (String arg[]) {
		Scanner sc = new Scanner(System.in);

		ArrayList<Integer> alist1 = new ArrayList<Integer>();
		ArrayList<Integer> alist2 = new ArrayList<Integer>();
		int ch, n, num;

		System.out.print("Enter no. of elements in first list: ");
		n = sc.nextInt();
		for (int i=0; i<n; i++) {
			System.out.print("Enter number to insert: ");
			num = sc.nextInt();
			alist1.add(num);
			}
		System.out.println(alist1);
		System.out.println();

		System.out.print("Enter no. of elements in second list: ");
		n = sc.nextInt();
		for (int i=0; i<n; i++) {
			System.out.print("Enter number to insert: ");
			num = sc.nextInt();
			alist2.add(num);
			}
		System.out.println(alist2);
		System.out.println();

		System.out.println("Menu:\n 1: merge\n 2: union\n 3: intersection\n 4: compare\n");
		System.out.print("Enter choice: ");
		ch = sc.nextInt();

		while (ch!=0) {
			switch (ch) {
				case 1: {
					ArrayList<Integer> alist = new ArrayList<Integer>();
					alist.addAll(alist1);
					alist.addAll(alist2);
					System.out.println("Merged list: "+alist);
					break;
					}
```

```
                    case 2: {
                        ArrayList<Integer> alist = new ArrayList<Integer>();
                        alist.addAll(alist1);
                        alist.addAll(alist2);
                        alist = removeDuplicates(alist);
                        System.out.println("Union of the lists: "+alist);
                        break;
                        }
                    case 3: {
                        ArrayList<Integer> alist = new ArrayList<Integer>();
                        for (Integer element : alist1) {
                            if (alist2.contains(element)) {
                                alist.add(element);
                                }
                            }
                        System.out.println("Intersection of the lists: "+alist);
                        break;
                        }
                    case 4: {
                        if (alist1.equals(alist2))
                            System.out.println("The lists are equal.");
                        else
                            System.out.println("The lists are not equal.");
                        break;
                        }
                    default: System.out.println("Invalid choice");
                    }
                System.out.println();

                System.out.print("Enter choice: ");
                ch = sc.nextInt();
                }
        }

    public static <T> ArrayList<T> removeDuplicates (ArrayList<T> list) {
        ArrayList<T> newList = new ArrayList<T>();
        for (T element : list) {
            if (!newList.contains(element)) {
                newList.add(element);
                }
            }
        return newList;
        }
    }
```

Name: Krithika Swaminathan
Roll No.: 205001057

Output:



```
kri@kri-ubuntu:~/workspace$ javac intArrList.java
kri@kri-ubuntu:~/workspace$ java intArrList
Enter no. of elements in first list: 3
Enter number to insert: 32
Enter number to insert: 45
Enter number to insert: 12
[32, 45, 12]

Enter no. of elements in second list: 3
Enter number to insert: 32
Enter number to insert: 12
Enter number to insert: 19
[32, 12, 19]

Menu:
 1: merge
 2: union
 3: intersection
 4: compare

Enter choice: 1
Merged list: [32, 45, 12, 32, 12, 19]

Enter choice: 2
Union of the lists: [32, 45, 12, 19]

Enter choice: 3
Intersection of the lists: [32, 12]

Enter choice: 4
The lists are not equal.

Enter choice: 0
kri@kri-ubuntu:~/workspace$ java intArrList
Enter no. of elements in first list: 2
Enter number to insert: 6
Enter number to insert: 8
[6, 8]

Enter no. of elements in second list: 2
Enter number to insert: 6
Enter number to insert: 8
[6, 8]

Menu:
 1: merge
 2: union
 3: intersection
 4: compare

Enter choice: 4
The lists are equal.

Enter choice: 0
kri@kri-ubuntu:~/workspace$ 
```

Q3: Using Collection framework, create a doubly linked list of integers and perform the given operations.

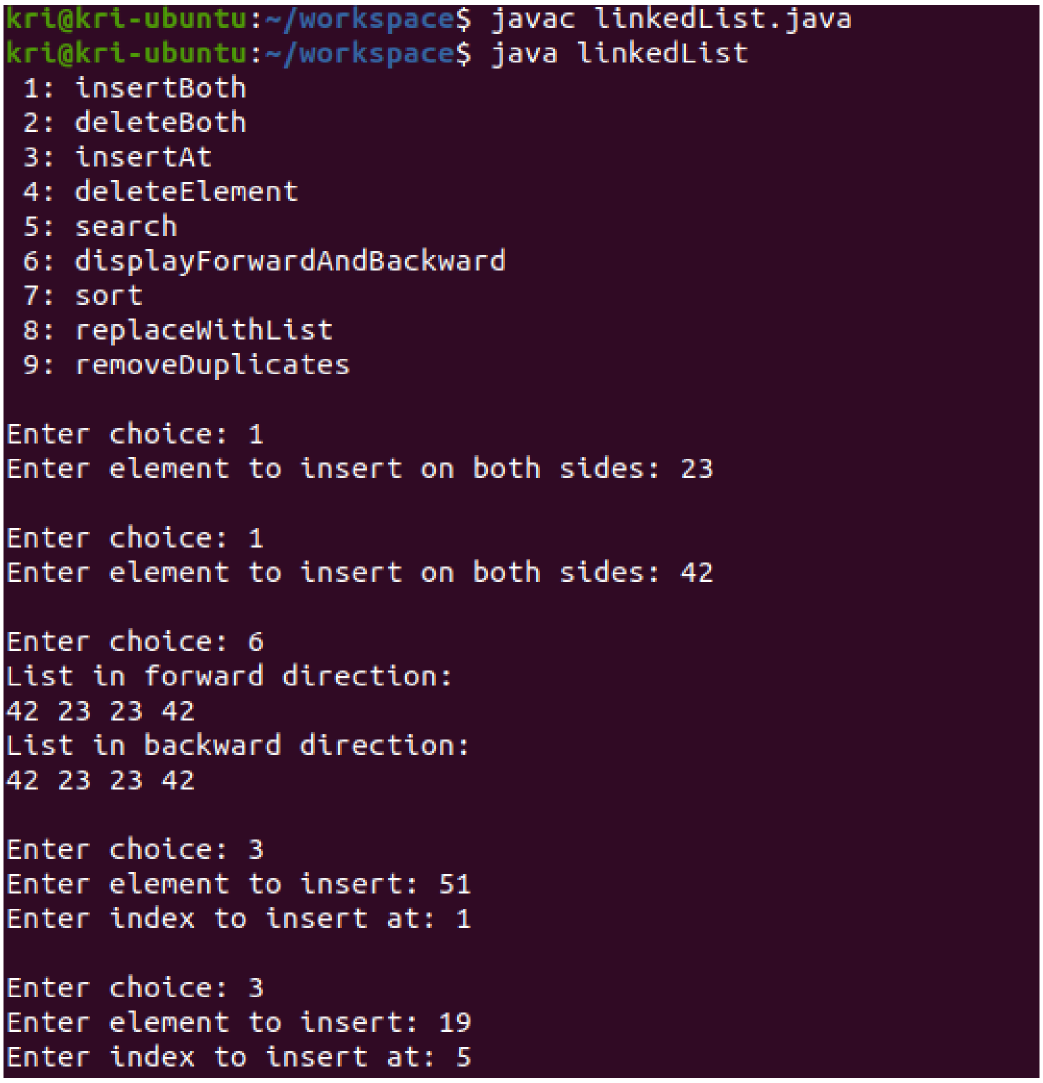
Code:

```
import java.util.*;

public class linkedList {
	public static void main (String arg[]) {
		Scanner sc = new Scanner(System.in);

		LinkedList<Integer> llist = new LinkedList<Integer>();
		int ch;

		System.out.println(" 1: insertBoth\n 2: deleteBoth\n 3: insertAt\n 4: deleteElement\n 5: 
search\n 6: displayForwardAndBackward\n 7: sort\n 8: replaceWithList\n 9: removeDuplicates\n");
		System.out.print("Enter choice: ");
		ch = sc.nextInt();

		while (ch!=0) {
			switch (ch) {
				case 1: {
					System.out.print("Enter element to insert on both sides: ");
					int el = sc.nextInt();
					llist.addFirst(el);
					llist.addLast(el);
					break;
					}
				case 2: {
					System.out.print("Deleting first element on both sides: ");
					llist.removeFirst();
					llist.removeLast();
					break;
					}
				case 3: {
					System.out.print("Enter element to insert: ");
					int el = sc.nextInt();
					System.out.print("Enter index to insert at: ");
					int pos = sc.nextInt();
					llist.add(pos,el);
					break;
					}
				case 4: {
					System.out.print("Enter element to remove: ");
					int el = sc.nextInt();
					llist.remove(llist.indexOf(el));
					break;
					}
```

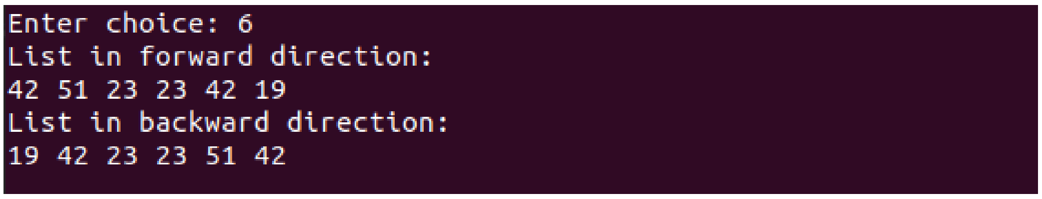
```
                case 5: {
                        System.out.print("Enter element to search for: ");
                        int el = sc.nextInt();
                        int found = llist.indexOf(el);
                        if (found == -1)
                                System.out.println("Element not found");
                        else
                                System.out.println("Found at index: "+found);
                        break;
                        }
                case 6: {
                        ListIterator<Integer> litr = llist.listIterator();
                        System.out.println("List in forward direction:");
                        while (litr.hasNext()) {
                                int el = litr.next();
                                System.out.print(el+" ");
                                }
                        System.out.println();
                        System.out.println("List in backward direction:");
                        while (litr.hasPrevious()) {
                                int el = litr.previous();
                                System.out.print(el+" ");
                                }
                        System.out.println();
                        break;
                        }
                case 7: {
                        Collections.sort(llist);
                        break;
                        }
                case 8: { //alternate version of case 8 included at the end
                        System.out.print("Enter element to replace: ");
                        int el1 = sc.nextInt();
                        System.out.print("Enter element to replace with: ");
                        int el2 = sc.nextInt();
                        llist.set(llist.indexOf(el1),el2);
                        break;
                        }
                case 9: {
                        llist = removeDuplicates(llist);
                        break;
                        }
                default: System.out.println("Invalid choice");
                }
        System.out.println();

        System.out.print("Enter choice: ");
        ch = sc.nextInt();
        }
```

```
		}

	public static <T> LinkedList<T> removeDuplicates (LinkedList<T> list) {
		LinkedList<T> newList = new LinkedList<T>();
		for (T element : list) {
			if (!newList.contains(element)) {
				newList.addLast(element);
				}
			}
		return newList;
		}
	}
```

Output:

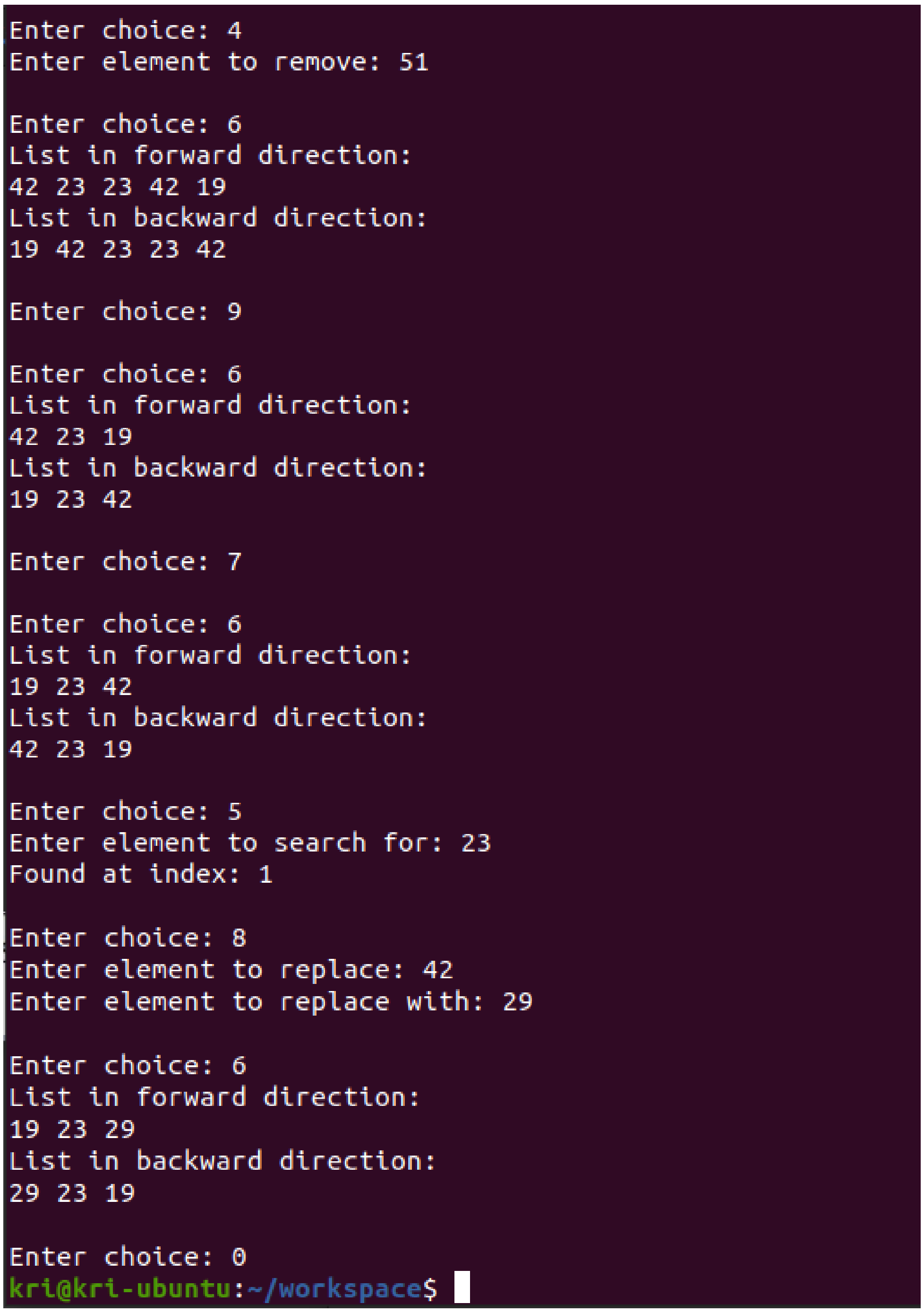


```
kri@kri-ubuntu:~/workspace$ javac linkedList.java
kri@kri-ubuntu:~/workspace$ java linkedList
 1: insertBoth
 2: deleteBoth
 3: insertAt
 4: deleteElement
 5: search
 6: displayForwardAndBackward
 7: sort
 8: replaceWithList
 9: removeDuplicates

Enter choice: 1
Enter element to insert on both sides: 23

Enter choice: 1
Enter element to insert on both sides: 42

Enter choice: 6
List in forward direction:
42 23 23 42
List in backward direction:
42 23 23 42

Enter choice: 3
Enter element to insert: 51
Enter index to insert at: 1

Enter choice: 3
Enter element to insert: 19
Enter index to insert at: 5
```



```
Enter choice: 6
List in forward direction:
42 51 23 23 42 19
List in backward direction:
19 42 23 23 51 42
```

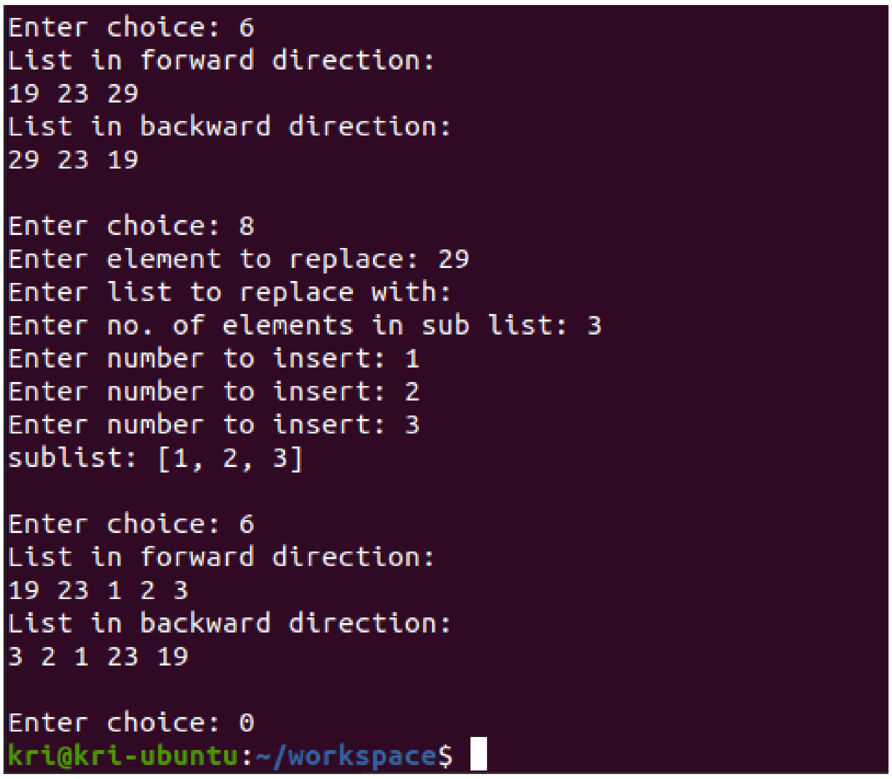
Name: Krithika Swaminathan
Roll No.: 205001057



```
Enter choice: 4
Enter element to remove: 51

Enter choice: 6
List in forward direction:
42 23 23 42 19
List in backward direction:
19 42 23 23 42

Enter choice: 9

Enter choice: 6
List in forward direction:
42 23 19
List in backward direction:
19 23 42

Enter choice: 7

Enter choice: 6
List in forward direction:
19 23 42
List in backward direction:
42 23 19

Enter choice: 5
Enter element to search for: 23
Found at index: 1

Enter choice: 8
Enter element to replace: 42
Enter element to replace with: 29

Enter choice: 6
List in forward direction:
19 23 29
List in backward direction:
29 23 19

Enter choice: 0
kri@kri-ubuntu:~/workspace$
```

Alternate version of case 8:

```
case 8: {
        LinkedList<Integer> llist2 = new LinkedList<Integer>();
        System.out.print("Enter element to replace: ");
        int el1 = sc.nextInt();
        System.out.println("Enter list to replace with: ");
        int n, num;
        System.out.print("Enter no. of elements in sub list: ");
        n = sc.nextInt();
        for (int i=0; i<n; i++) {
                System.out.print("Enter number to insert: ");
                num = sc.nextInt();
                llist2.add(num);
                }
        System.out.println("sublist: "+llist2);
        int index = llist.indexOf(el1);
        llist.remove(index);
        llist.addAll(index,llist2);
        break;
        }
```


```
Enter choice: 6
List in forward direction:
19 23 29
List in backward direction:
29 23 19

Enter choice: 8
Enter element to replace: 29
Enter list to replace with:
Enter no. of elements in sub list: 3
Enter number to insert: 1
Enter number to insert: 2
Enter number to insert: 3
sublist: [1, 2, 3]

Enter choice: 6
List in forward direction:
19 23 1 2 3
List in backward direction:
3 2 1 23 19

Enter choice: 0
kri@kri-ubuntu:~/workspace$
```