

## Lab 3: Array List and Linked List - CST3108

**Name:** Chris Mugabo

**Date:** 28/01/2025

### Task 1: Implementing with ArrayList

#### Code Implementation and Explanation

Below is the complete Java code implemented for this lab:

```
import java.util.ArrayList;
```

```
import java.util.Objects;
```

```
class Student {
```

```
    String name;
```

```
    String lab;
```

```
    int ssid;
```

```
    public Student(String name, String lab, int ssid) {
```

```
        this.name = name;
```

```
        this.lab = lab;
```

```
        this.ssid = ssid;
```

```
    }
```

```
    @Override
```

```
    public String toString() {
```

```
        return "Name: " + this.name + ", Lab: " + this.lab + ", SSID: " + this.ssid;
```

```
    }
```

```
    @Override
```

```
    public boolean equals(Object obj) {
```

```
        if (this == obj) return true;
```

```
        if (obj == null || getClass() != obj.getClass()) return false;
```

```
        Student student = (Student) obj;
```

```
        return ssid == student.ssid && name.equals(student.name) && lab.equals(student.lab);
```

```
    }
```

```
    @Override
```

```
    public int hashCode() {
```

```
        return Objects.hash(name, lab, ssid);
```

```
    }
```

```
}
```

```

public class ArrayListDemo {
    public static void main(String[] args) {
        ArrayList<Student> AL1 = new ArrayList<Student>();
        AL1.add(new Student("St1", "CST3108", 1));
        AL1.add(new Student("St2", "CST3108", 2));
        AL1.add(new Student("St3", "CST3108", 3));
        AL1.add(new Student("St4", "CST3108", 4));
        AL1.add(new Student("St5", "CST3108", 5));
        AL1.add(new Student("St6", "CST3108", 6));

        Student duplicate = new Student("St6", "CST3108", 6);
        if (AL1.contains(duplicate)) {
            System.out.println("Error: This student is already in the list.");
        } else {
            AL1.add(duplicate);
            System.out.println("Added duplicate student as ArrayList allows duplicates without
explicit checks.");
        }

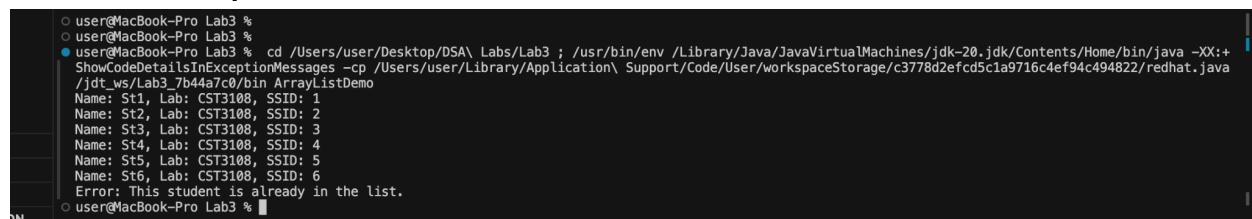
        for (Student s : AL1) {
            System.out.println(s);
        }
    }
}

```

### Detailed Explanation:

- **Student Class:** Defines the structure and behavior of **Student** objects with methods for object creation and data presentation.
- **ArrayList Implementation:** **AL1** is used to store and manage **Student** objects, showcasing the dynamic resizing capability of **ArrayList**.
- **Duplicate Handling:** The program checks for duplicates before adding a new student to the list. This demonstrates how to manage data uniqueness in a collection

### Results and Output:



```

user@MacBook-Pro Lab3 %
user@MacBook-Pro Lab3 %
user@MacBook-Pro Lab3 % cd /Users/user/Desktop/DSA\ Labs/Lab3 ; /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/bin/java -XX:+
ShowCodeDetailsInExceptionMessages -cp /Users/user/Library/Application\ Support/Code/User/workspaceStorage/c3778d2efcd5c1a9716c4ef94c494822/redhat.java
/jdt_ws/Lab3_7b44a7c0/bin ArrayListDemo
Name: St1, Lab: CST3108, SSID: 1
Name: St2, Lab: CST3108, SSID: 2
Name: St3, Lab: CST3108, SSID: 3
Name: St4, Lab: CST3108, SSID: 4
Name: St5, Lab: CST3108, SSID: 5
Name: St6, Lab: CST3108, SSID: 6
Error: This student is already in the list.
user@MacBook-Pro Lab3 %

```

**Task 2:** Submission by due date.

### 5. List Reversal Using Collections

- **Purpose:** Understand and implement list manipulation techniques.

**Action:**

```
Collections.swap(AL1, i, AL1.size() - i - 1);
```

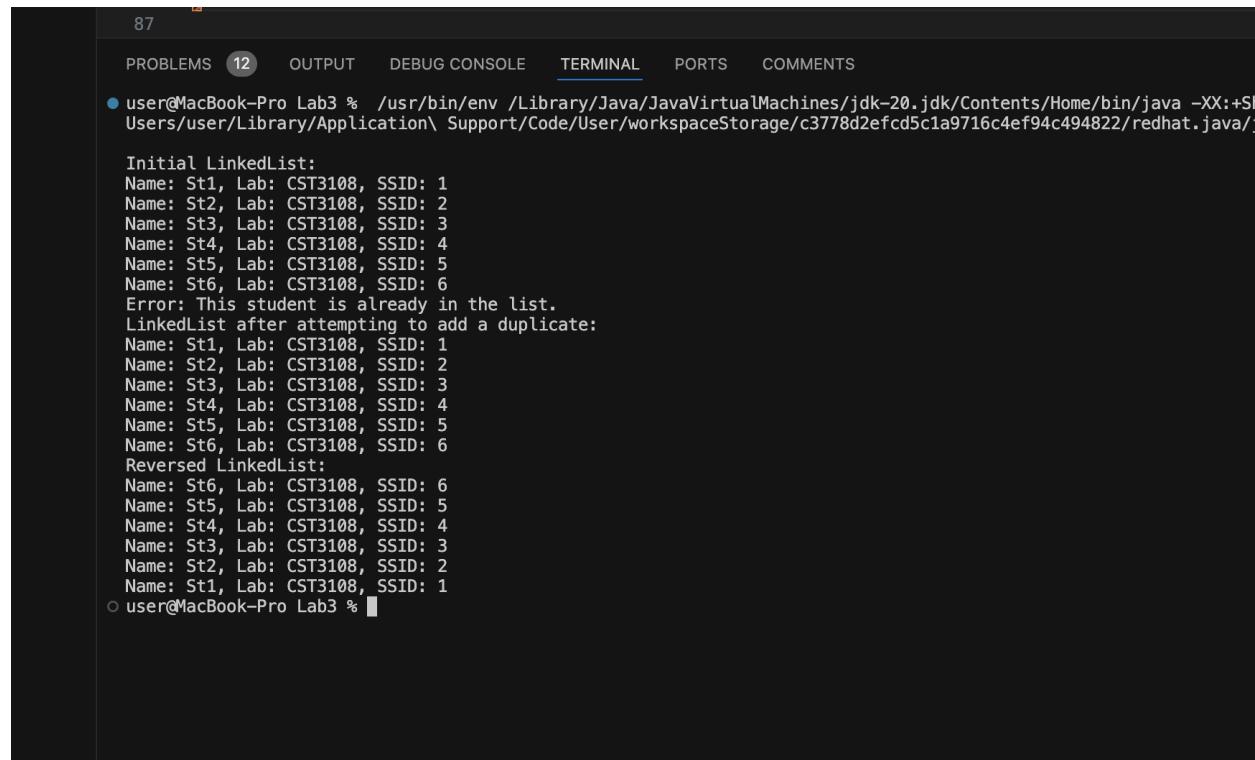
### 6. LinkedList Implementation

**Implementation:**

```
LinkedList<Student> LL1 = new LinkedList<>(AL1);  
// Perform similar actions as with ArrayList
```

### 7. Results Display

- **Console Output**



```
87  
PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS  
● user@MacBook-Pro Lab3 % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/bin/java -XX:+S  
Users/user/Library/Application\ Support/Code/User/workspaceStorage/c3778d2efcd5c1a9716c4ef94c494822/redhat.java/  
  
Initial LinkedList:  
Name: St1, Lab: CST3108, SSID: 1  
Name: St2, Lab: CST3108, SSID: 2  
Name: St3, Lab: CST3108, SSID: 3  
Name: St4, Lab: CST3108, SSID: 4  
Name: St5, Lab: CST3108, SSID: 5  
Name: St6, Lab: CST3108, SSID: 6  
Error: This student is already in the list.  
LinkedList after attempting to add a duplicate:  
Name: St1, Lab: CST3108, SSID: 1  
Name: St2, Lab: CST3108, SSID: 2  
Name: St3, Lab: CST3108, SSID: 3  
Name: St4, Lab: CST3108, SSID: 4  
Name: St5, Lab: CST3108, SSID: 5  
Name: St6, Lab: CST3108, SSID: 6  
Reversed LinkedList:  
Name: St6, Lab: CST3108, SSID: 6  
Name: St5, Lab: CST3108, SSID: 5  
Name: St4, Lab: CST3108, SSID: 4  
Name: St3, Lab: CST3108, SSID: 3  
Name: St2, Lab: CST3108, SSID: 2  
Name: St1, Lab: CST3108, SSID: 1  
○ user@MacBook-Pro Lab3 %
```