Inheritance & Comparable

```java
public class SavingsAccount extends BankAccount
                            implements Comparable<SavingsAccount> {
    private double rate;

    public SavingsAccount(String name, double balance, double rate) {
        super(name, balance);
        this.rate = rate;
    }

    public double getInterestRate() {
        return rate;
    }

    public void nextYearBalance() {
        double interest = this.getBalance() * this.rate;
        super.setBalance(this.getBalance() + interest);
    }

    @Override
    public String toString() {
        if (this.getBalance() <= 0) {
            return "no money D:";
        }
        return super.toString();
    }

    public int compareTo(SavingsAccount other) {
        if (this.getBalance() > other.getBalance()) {
            return 1;
        } else if (this.getBalance() < other.getBalance()) {
            return -1;
        } else if (this.getInterestRate() > other.getInterestRate()) {
            return 1;
        } else if (this.getInterestRate() < other.getInterestRate()) {
            return -1;
        } else {
            return 0;
        }
    }
}
```

LinkedList

```java
public void removeDuplicates() {
    if (head != null) {
        // Optional: Specifically accounting for front case
        while (head.next != null && head.data == head.next.data) {
            head.next = head.next.next;
            size--;
        }
        ListNode curr = head.next;
        while (curr != null && curr.next != null) {
            if (curr.data == curr.next.data) {
                curr.next = curr.next.next;
                size--;
            } else {
                curr = curr.next;
            }
        }
    }
}
```

Binary Trees

```
public void mirror() {
    overallRoot = mirror(overallRoot);
}

private IntTreeNode mirror(IntTreeNode root) {
    if (root != null) {
        IntTreeNode temp = root.left;
        root.left = mirror(root.right);
        root.right = mirror(temp);
    }
    return root;
}
```