PB: 230: Write a Java program to remove duplicates from an ArrayList as per given input and output. Take main class 'Exam'. Create ArrayList: [1, 2, 3, 4, 4, 5, 6, 7, 8, 9, 6, 9, 1, 1, 10] in main method then you need to implement another method named 'removeDuplicate' that takes an ArrayList as an argument , removes all duplicates and returns an ArrayList without duplicates. At end print that Arraylist from main method. (No need to use Scannner)

Output of your code should be:- [2, 3, 5, 7, 8, 10]

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;

public class Exam {

   public static void main(String[] args) {
      ArrayList<Integer> numbers = new ArrayList<>(Arrays.asList(1, 2, 3, 4, 4, 5, 6, 7, 8, 9, 6, 9, 1, 1, 10));
      System.out.println(removeDuplicate(numbers));
   }

   public static ArrayList removeDuplicate(ArrayList<Integer> list) {
      for (int i = 0; i < list.size(); i++) {
         Integer current = list.get(i);
         int count  = Collections.frequency(list, current);

         // If duplicate found, remove all its occurrences
         if (count > 1) {
            ArrayList a = new ArrayList<>();
            a.add(current);
            list.removeAll(a);
         }
      }
      return list;
   }
}
```

PB: 234: Problem Statement: Employee Management System Implement an employee management system in Java using ArrayList and a class. The system should have the following functionalities: Add an employee to the system Remove an employee from the system Update an employee's details Display all employees in the system To implement this system, you can create an Employee class with attributes such as name, age, salary, etc. Then, you can create an ArrayList to store all the employees. You can then implement functions to add, remove, update and display employees.

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

class Employee {
   String name;
   int age;
   double salary;
```

```
    public Employee(String name, int age, double salary) {
        this.name = name;
        this.age = age;
        this.salary = salary;
    }

    public void display() {
        System.out.println("Name: " + name + ", Age: " + age + ", Salary: $" + salary);
    }
}

public class EmployeeManagementSystem {
    static ArrayList<Employee> employees = new ArrayList<>();
    static Scanner sc = new Scanner(System.in);  // Global Scanner

    public static void addEmployee() {
        System.out.print("Enter name: ");
        String name = sc.nextLine();
        System.out.print("Enter age: ");
        int age = sc.nextInt();
        System.out.print("Enter salary: ");
        double salary = sc.nextDouble();
        sc.nextLine(); // consume leftover newline
        employees.add(new Employee(name, age, salary));
        System.out.println("Employee added successfully.");
    }

    public static void removeEmployee() {
        System.out.print("Enter name of employee to remove: ");
        String name = sc.nextLine();
        boolean found = false;
        for (int i = 0; i < employees.size(); i++) {
            if (employees.get(i).name.equalsIgnoreCase(name)) {
                employees.remove(i);
                System.out.println("Employee removed successfully.");
                found = true;
                break;
            }
        }
        if (!found) {
            System.out.println("Employee not found.");
        }
    }

    public static void updateEmployee() {
        System.out.print("Enter name of employee to update: ");
        String name = sc.nextLine();
        boolean found = false;

        for (Employee emp : employees) {
            if (emp.name.equalsIgnoreCase(name)) {
                found = true;
                System.out.println("What do you want to update?");
                System.out.println("1. Name");
                System.out.println("2. Age");
                System.out.println("3. Salary");
                System.out.print("Enter your choice: ");
                int updateChoice = sc.nextInt();
                sc.nextLine(); // consume newline

                switch (updateChoice) {
                    case 1:
                        System.out.print("Enter new name: ");
                        emp.name = sc.nextLine();
                        System.out.println("Name updated successfully.");
                        break;
                    case 2:
                        System.out.print("Enter new age: ");
                        emp.age = sc.nextInt();
                        sc.nextLine();
                        System.out.println("Age updated successfully.");
                        break;
                    case 3:
                        System.out.print("Enter new salary: ");
                        emp.salary = sc.nextDouble();
                        sc.nextLine();
                        System.out.println("Salary updated successfully.");
                        break;
                    default:
                        System.out.println("Invalid choice.");
                }
                break;
            }
        }

        if (!found) {
            System.out.println("Employee not found.");
        }
    }

    public static void displayEmployees() {
        if (employees.isEmpty()) {
            System.out.println("No employees in the system.");
        } else {
            for (Employee emp : employees) {
```

```
            emp.display();
        }
    }
}

public static void main(String[] args) {
    int choice;

    do {
        System.out.println("\n=== Employee Management System ===");
        System.out.println("1. Add Employee");
        System.out.println("2. Remove Employee");
        System.out.println("3. Update Employee");
        System.out.println("4. Display All Employees");
        System.out.println("5. Exit");
        System.out.print("Enter your choice: ");
        choice = sc.nextInt();
        sc.nextLine(); // consume newline

        switch (choice) {
            case 1:
                addEmployee();
                break;
            case 2:
                removeEmployee();
                break;
            case 3:
                updateEmployee();
                break;
            case 4:
                displayEmployees();
                break;
            case 5:
                System.out.println("Exiting system. Goodbye!");
                break;
            default:
                System.out.println("Invalid choice. Please try again.");
        }

    } while (choice != 5);
}
}
```

PB: 235: Write a Java program that creates an ArrayList to store a list of students and their grades. The program should provide the following functionality:

- **Add a student: The program should prompt the user for a student name and their grade, and add the student to the ArrayList.**
- **Remove a student: The program should prompt the user for a student name and remove the corresponding entry from the ArrayList.**

- **Search for a student: The program should prompt the user for a student name and output their corresponding grade. If the student name is not found in the ArrayList, the program should output an error message.**
- **Sort students by name: The program should sort the ArrayList by student name in ascending order.**
- **Sort students by grade: The program should sort the ArrayList by student grade in descending order.**

To accomplish these tasks, you can use the following Collections class methods:

1. **addAll(collection, elements): Adds all elements from one collection to another.**
2. **sort(list): Sorts the elements in a list in natural order.**
3. **sort(list, comparator): Sorts the elements in a list using a custom**
4. **comparator. reverse(list): Reverses the order of the elements in a list.**

You can also use the following ArrayList methods:

1. **add(element): Adds an element to the end of the ArrayList.**
2. **remove(element): Removes the first occurrence of an element from the**
3. **ArrayList. get(index): Returns the element at the specified index in the**
4. **ArrayList. size(): Returns the number of elements in the ArrayList.**

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.Scanner;

class Student {
    private String name;
    private double grade;

    public Student(String name, double grade) {
        this.name = name;
        this.grade = grade;
    }

    // Getter methods for Comparator.comparing
    public String getName() {
        return name;
    }

    public double getGrade() {
        return grade;
    }

    public void display() {
        System.out.println("Name: " + name + ", Grade: " + grade);
    }
}

public class StudentGradeManager {
```

```
static ArrayList<Student> students = new ArrayList<>();
static Scanner sc = new Scanner(System.in);

public static void addStudent() {
   System.out.print("Enter student name: ");
   String name = sc.nextLine();
   System.out.print("Enter grade: ");
   double grade = sc.nextDouble();
   sc.nextLine(); // consume newline
   students.add(new Student(name, grade));
   System.out.println("Student added successfully.");
}

public static void removeStudent() {
   System.out.print("Enter name of student to remove: ");
   String name = sc.nextLine();
   boolean removed = false;

   for (int i = 0; i < students.size(); i++) {
      if (students.get(i).getName().equalsIgnoreCase(name)) {
         students.remove(i);
         System.out.println("Student removed successfully.");
         removed = true;
         break;
      }
   }

   if (!removed) {
      System.out.println("Student not found.");
   }
}

public static void searchStudent() {
   System.out.print("Enter name of student to search: ");
   String name = sc.nextLine();
   boolean found = false;

   for (Student s : students) {
      if (s.getName().equalsIgnoreCase(name)) {
         s.display();
         found = true;
         break;
      }
   }

   if (!found) {
      System.out.println("Student not found.");
   }
```

```
    }

    public static void sortByName() {
        Collections.sort(students, Comparator.comparing(Student::getName,
String.CASE_INSENSITIVE_ORDER));
        System.out.println("Students sorted by name (ascending):");
        for (Student s : students) {
            s.display();
        }
    }

    public static void sortByGrade() {
        Collections.sort(students, Comparator.comparing(Student::getGrade).reversed());
        System.out.println("Students sorted by grade (descending):");
        for (Student s : students) {
            s.display();
        }
    }

    public static void main(String[] args) {
        int choice;

        do {
            System.out.println("\n=== Student Grade Management System ===");
            System.out.println("1. Add Student");
            System.out.println("2. Remove Student");
            System.out.println("3. Search Student");
            System.out.println("4. Sort by Name");
            System.out.println("5. Sort by Grade");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();
            sc.nextLine(); // consume newline

            switch (choice) {
                case 1:
                    addStudent();
                    break;
                case 2:
                    removeStudent();
                    break;
                case 3:
                    searchStudent();
                    break;
                case 4:
                    sortByName();
                    break;
                case 5:
```

```
                sortByGrade();
                break;
            case 6:
                System.out.println("Exiting program.");
                break;
            default:
                System.out.println("Invalid choice. Try again.");
        }

    } while (choice != 6);

    sc.close();
  }
}
```

PB: 236: Problem Statement: Library Management System You are required to design a library management system in Java using ArrayList. The system should have the following classes:

- **Book - This class should contain the following attributes: bookId, bookName, author, and quantity. It should also have a method to display book details.**
- **User - This class should contain the following attributes: userId, userName, and booksIssued. It should also have methods to display user details and to issue/return a book.**
- **Library - This class should contain an ArrayList to store all the books and an ArrayList to store all the users. It should have methods to add/remove books and users, to display all books/users, to issue/return a book, and to display all books issued by a particular user.**

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

// Book Class
class Book {
  int bookId;
  String bookName;
  String author;
  int quantity;

  public Book(int bookId, String bookName, String author, int quantity) {
    this.bookId = bookId;
    this.bookName = bookName;
    this.author = author;
    this.quantity = quantity;
  }

  public void display() {
    System.out.println("ID: " + bookId + ", Name: " + bookName + ", Author: " + author + ", Quantity: " +
quantity);
  }
}
```

```
// User Class
class User {
    int userId;
    String userName;
    ArrayList<Book> booksIssued = new ArrayList<>();

    public User(int userId, String userName) {
        this.userId = userId;
        this.userName = userName;
    }

    public void display() {
        System.out.println("User ID: " + userId + ", Name: " + userName);
    }

    public void issueBook(Book book) {
        booksIssued.add(book);
    }

    public void returnBook(int bookId) {
        for (int i = 0; i < booksIssued.size(); i++) {
            if (booksIssued.get(i).bookId == bookId) {
                booksIssued.remove(i);
                return;
            }
        }
    }

    public void displayIssuedBooks() {
        System.out.println("Books issued to " + userName + ":");
        if (booksIssued.isEmpty()) {
            System.out.println("No books issued.");
        } else {
            for (Book book : booksIssued) {
                book.display();
            }
        }
    }
}

// Library Class
class Library {
    ArrayList<Book> books = new ArrayList<>();
    ArrayList<User> users = new ArrayList<>();
    Scanner sc = new Scanner(System.in);

    public void addBook() {
        System.out.print("Enter book ID, name, author, quantity: ");
        int id = sc.nextInt();
        sc.nextLine();
        String name = sc.nextLine();
        String author = sc.nextLine();
        int qty = sc.nextInt();
        sc.nextLine();
        books.add(new Book(id, name, author, qty));
        System.out.println("Book added.");
    }

    public void removeBook() {
        System.out.print("Enter book ID to remove: ");
        int id = sc.nextInt();
        sc.nextLine();
        for (int i = 0; i < books.size(); i++) {
            if (books.get(i).bookId == id) {
                books.remove(i);
                System.out.println("Book removed.");
                return;
            }
        }
        System.out.println("Book not found.");
    }

    public void addUser() {
        System.out.print("Enter user ID and name: ");
        int id = sc.nextInt();
        sc.nextLine();
        String name = sc.nextLine();
        users.add(new User(id, name));
        System.out.println("User added.");
    }

    public void removeUser() {
        System.out.print("Enter user ID to remove: ");
        int id = sc.nextInt();
        sc.nextLine();
        for (int i = 0; i < users.size(); i++) {
            if (users.get(i).userId == id) {
                users.remove(i);
                System.out.println("User removed.");
                return;
            }
        }
        System.out.println("User not found.");
    }

    public void displayAllBooks() {
```

```
    System.out.println("Books in library:");
    for (Book b : books) {
        b.display();
    }
}

public void displayAllUsers() {
    System.out.println("Users in library:");
    for (User u : users) {
        u.display();
    }
}

public void issueBook() {
    System.out.print("Enter user ID and book ID to issue: ");
    int uid = sc.nextInt();
    int bid = sc.nextInt();
    sc.nextLine();

    User user = null;
    Book book = null;

    for (User u : users) {
        if (u.userId == uid) {
            user = u;
            break;
        }
    }

    for (Book b : books) {
        if (b.bookId == bid && b.quantity > 0) {
            book = b;
            break;
        }
    }

    if (user != null && book != null) {
        user.issueBook(book);
        book.quantity--;
        System.out.println("Book issued.");
    } else {
        System.out.println("User or book not found, or book out of stock.");
    }
}

public void returnBook() {
    System.out.print("Enter user ID and book ID to return: ");
    int uid = sc.nextInt();
```

```
        int bid = sc.nextInt();
        sc.nextLine();

        User user = null;

        for (User u : users) {
            if (u.userId == uid) {
                user = u;
                break;
            }
        }

        if (user != null) {
            for (Book b : books) {
                if (b.bookId == bid) {
                    user.returnBook(bid);
                    b.quantity++;
                    System.out.println("Book returned.");
                    return;
                }
            }
        }

        System.out.println("User or book not found.");
    }

    public void showIssuedBooksByUser() {
        System.out.print("Enter user ID: ");
        int uid = sc.nextInt();
        sc.nextLine();

        for (User u : users) {
            if (u.userId == uid) {
                u.displayIssuedBooks();
                return;
            }
        }

        System.out.println("User not found.");
    }
}

// Main Class
public class LibraryManagementSystem {
    public static void main(String[] args) {
        Library library = new Library();
        Scanner sc = new Scanner(System.in);
        int choice;
```

```
do {
  System.out.println("\n=== Library Management Menu ===");
  System.out.println("1. Add Book");
  System.out.println("2. Remove Book");
  System.out.println("3. Add User");
  System.out.println("4. Remove User");
  System.out.println("5. Display All Books");
  System.out.println("6. Display All Users");
  System.out.println("7. Issue Book");
  System.out.println("8. Return Book");
  System.out.println("9. Show Issued Books by User");
  System.out.println("10. Exit");
  System.out.print("Enter your choice: ");
  choice = sc.nextInt();

  switch (choice) {
    case 1:
      library.addBook();
      break;
    case 2:
      library.removeBook();
      break;
    case 3:
      library.addUser();
      break;
    case 4:
      library.removeUser();
      break;
    case 5:
      library.displayAllBooks();
      break;
    case 6:
      library.displayAllUsers();
      break;
    case 7:
      library.issueBook();
      break;
    case 8:
      library.returnBook();
      break;
    case 9:
      library.showIssuedBooksByUser();
      break;
    case 10:
      System.out.println("Exiting Library System.");
      break;
    default:
```

```
                System.out.println("Invalid choice!");
            }
        } while (choice != 10);

        sc.close();
    }
}
```

PB: 237: Write a Java program to insert the specified element at the specified position in the linked list.

```
import java.util.LinkedList;
import java.util.Scanner;

public class InsertInLinkedList {
    public static void main(String[] args) {
        LinkedList<String> list = new LinkedList<>();

        // Initial elements
        list.add("Apple");
        list.add("Banana");
        list.add("Cherry");

        System.out.println("Original LinkedList: " + list);

        Scanner sc = new Scanner(System.in);

        // Input element and position
        System.out.print("Enter element to insert: ");
        String element = sc.nextLine();

        System.out.print("Enter position (0 to " + list.size() + "): ");
        int position = sc.nextInt();

        // Validation
        if (position < 0 || position > list.size()) {
            System.out.println("Invalid position!");
        } else {
            list.add(position, element);
            System.out.println("Updated LinkedList: " + list);
        }

        sc.close();
    }
}
```

PB: 238: Write a Java program to get the first and last occurrence of the specified elements in a linked list.

```
import java.util.LinkedList;
import java.util.Scanner;

public class FirstLastOccurrence {
    public static void main(String[] args) {
        LinkedList<String> list = new LinkedList<>();

        // Adding elements with duplicates
        list.add("Apple");
        list.add("Banana");
        list.add("Cherry");
        list.add("Banana");
        list.add("Date");
        list.add("Banana");

        System.out.println("LinkedList: " + list);

        Scanner sc = new Scanner(System.in);
        System.out.print("Enter element to search: ");
        String element = sc.nextLine();

        int firstIndex = list.indexOf(element);
        int lastIndex = list.lastIndexOf(element);

        if (firstIndex == -1) {
            System.out.println("Element '" + element + "' not found in the list.");
        } else {
            System.out.println("First occurrence of '" + element + "': Index " + firstIndex);
            System.out.println("Last occurrence of '" + element + "': Index " + lastIndex);
        }

        sc.close();
    }
}
```

PB: 239: Write a Java program to clone an linked list to another linked list.

```
import java.util.LinkedList;

public class CloneLinkedList {
    public static void main(String[] args) {
        // Original LinkedList
        LinkedList<String> originalList = new LinkedList<>();
        originalList.add("Apple");
        originalList.add("Banana");
        originalList.add("Cherry");
        System.out.println("Original LinkedList: " + originalList);

        // Cloning the LinkedList
        @SuppressWarnings("unchecked")
        LinkedList<String> clonedList = (LinkedList<String>) originalList.clone();

        System.out.println("Cloned LinkedList: " + clonedList);
    }
}
```

Output:

```
Original LinkedList: [Apple, Banana, Cherry]
Cloned LinkedList: [Apple, Banana, Cherry]
```

PB: 240: Write a Java program to append the specified element to the end of a linked list & Write a Java program to iterate through all elements in a linked list.

```
import java.util.LinkedList;
import java.util.Scanner;

public class LinkedListOperations {
    public static void main(String[] args) {
        LinkedList<String> list = new LinkedList<>();

        // Adding some initial elements
        list.add("Apple");
        list.add("Banana");
        list.add("Cherry");

        System.out.println("Original LinkedList: " + list);

        // Appending element to the end
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter an element to append: ");
        String newElement = sc.nextLine();
        list.addLast(newElement); // or list.add(newElement);

        System.out.println("Updated LinkedList: " + list);

        // Iterating through all elements
        System.out.println("Iterating through LinkedList elements:");
        for (String item : list) {
            System.out.println(item);
        }

        sc.close();
    }
}
```

PB: 241: Problem Statement: Implement a Phone Book Write a Java program to implement a phone book using an ArrayList and a LinkedList. The phone book should allow users to add, delete, and search for contacts by name or phone number. To solve this problem, you can create a Contact class that stores the name and phone number of each contact. You can then use an ArrayList to store the contacts and a LinkedList to store the indices of the contacts sorted by name.

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.LinkedList;
import java.util.Scanner;

class Contact {
   String name;
   String phone;

   Contact(String name, String phone) {
      this.name = name;
      this.phone = phone;
   }

   void display() {
      System.out.println("Name: " + name + ", Phone: " + phone);
   }

   public String getName() {
      return name;
   }

   public String getPhone() {
      return phone;
   }
}

public class PhoneBook {
   static ArrayList<Contact> contacts = new ArrayList<>();
   static LinkedList<Contact> sortedIndices = new LinkedList<>();

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      int choice;

      do {
         System.out.println("\nPhone Book Menu:");
         System.out.println("1. Add Contact");
         System.out.println("2. Delete Contact");
         System.out.println("3. Search by Name");
         System.out.println("4. Search by Phone");
```

```java
            System.out.println("5. View All Contacts (Sorted by Name)");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            choice = Integer.parseInt(sc.nextLine());

            switch (choice) {
                case 1:
                    addContact(sc);
                    break;
                case 2:
                    deleteContact(sc);
                    break;
                case 3:
                    searchByName(sc);
                    break;
                case 4:
                    searchByPhone(sc);
                    break;
                case 5:
                    viewAllSorted();
                    break;
                case 6:
                    System.out.println("Exiting Phone Book...");
                    break;
                default:
                    System.out.println("Invalid choice!");
            }
        } while (choice != 6);

        sc.close();
    }

    static void addContact(Scanner sc) {
        System.out.print("Enter Name: ");
        String name = sc.nextLine();
        System.out.print("Enter Phone Number: ");
        String phone = sc.nextLine();

        Contact newContact = new Contact(name, phone);
        contacts.add(newContact);
        sortedIndices.add(newContact);
        System.out.println("Contact added.");
    }

    static void deleteContact(Scanner sc) {
        System.out.print("Enter Name to delete: ");
        String name = sc.nextLine();
        boolean found = false;
        for (int i = 0; i < contacts.size(); i++) {
            if (contacts.get(i).name.equalsIgnoreCase(name)) {
                contacts.remove(i);

                System.out.println("Contact deleted.");
                found = true;
                break;
            }
        }

        if (!found) {
            System.out.println("Contact not found.");
        }
    }

    static void searchByName(Scanner sc) {
        System.out.print("Enter Name to search: ");
        String name = sc.nextLine();
        boolean found = false;

        for (Contact c : contacts) {
            if (c.name.equalsIgnoreCase(name)) {
                c.display();
                found = true;
            }
        }

        if (!found) {
            System.out.println("Contact not found.");
        }
    }

    static void searchByPhone(Scanner sc) {
        System.out.print("Enter Phone Number to search: ");
        String phone = sc.nextLine();
        boolean found = false;

        for (Contact c : contacts) {
            if (c.phone.equals(phone)) {
                c.display();
                found = true;
            }
        }

        if (!found) {
            System.out.println("Contact not found.");
        }
    }

    static void viewAllSorted() {
        if (contacts.isEmpty()) {
            System.out.println("No contacts to display.");
            return;
        }

        System.out.println("All Contacts (Sorted by Name):");
        Collections.sort(sortedIndices, Comparator.comparing(Contact::getName));
        for (Contact index : sortedIndices) {
            index.display();
        }
    }
}
```