```
----- Question 1-----
import java.util.Scanner;
import java.util.Stack;
public class BrowserHistory {
  private Stack<String> backStack = new Stack<>();
  private Stack<String> forwardStack = new Stack<>();
  private String currentPage = "Home";
  public void visit(String url) {
    backStack.push(currentPage);
    currentPage = url;
    forwardStack.clear();
    System.out.println("Visited: " + currentPage);
  }
  public void goBack() {
    if (!backStack.isEmpty()) {
       forwardStack.push(currentPage);
       currentPage = backStack.pop();
       System.out.println("Went back to: " + currentPage);
    } else {
       System.out.println("No pages in back history.");
    }
```

```
}
  public void goForward() {
    if (!forwardStack.isEmpty()) {
       backStack.push(currentPage);
       currentPage = forwardStack.pop();
       System.out.println("Went forward to: " + currentPage);
    } else {
       System.out.println("No pages in forward history.");
    }
  }
  public void showCurrentPage() {
    System.out.println("Current Page: " + currentPage);
  }
  public static void main(String[] args) {
    BrowserHistory browser = new BrowserHistory();
     Scanner scanner = new Scanner(System.in);
    while (true) {
       System.out.println("\n1. Visit New Page\n2. Go Back\n3. Go Forward\n4. Show Current
Page\n5. Exit");
       int choice = scanner.nextInt();
       scanner.nextLine();
       switch (choice) {
```

```
case 1:
            System.out.print("Enter URL: ");
            String url = scanner.nextLine();
            browser.visit(url);
            break;
          case 2:
            browser.goBack();
            break;
          case 3:
            browser.goForward();
            break;
          case 4:
            browser.showCurrentPage();
            break;
          case 5:
            System.exit(0);
       }
    }
  }
}
```

into new customers



```
⇔ Share

                                                                     Run
                                                                               Output
BrowserHistory.java
                System.out.println("\n1. Visit New Page\n2. Go Back\n3.
                    Go Forward\n4. Show Current Page\n5. Exit");
                                                                              1. Visit New
45
                int choice = scanner.nextInt();
                                                                              2. Go Back
46
                scanner.nextLine();
                                                                              3. Go Forward
47 -
                switch (choice) {
                                                                              4. Show Curre
                    case 1:
48
                                                                              5. Exit
                        System.out.print("Enter URL: ");
49
                        String url = scanner.nextLine();
50
51
                        browser.visit(url);
52
                        break;
53
                    case 2:
54
                        browser.goBack();
55
                        break;
56
                    case 3:
                        browser.goForward();
57
58
                        break;
59
                    case 4:
                        browser.showCurrentPage();
```

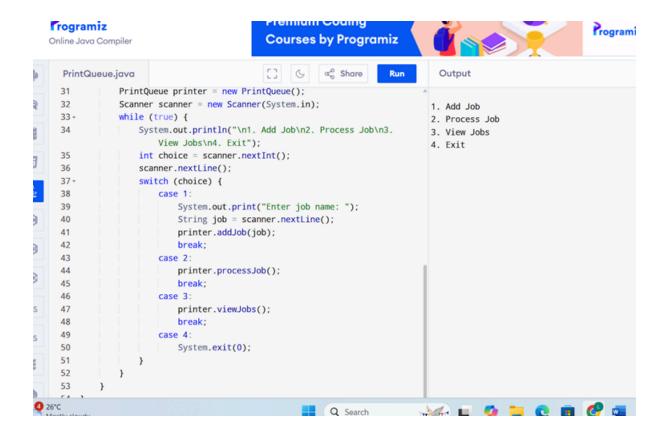
```
import java.util.LinkedList;
import java.util.Queue;
import java.util.Scanner;

public class PrintQueue {
    private Queue<String> queue = new LinkedList<>();
    public void addJob(String job) {
        queue.offer(job);
        System.out.println("Added job: " + job);
    }
}
```

-----Question 2-----

```
}
public void processJob() {
  if (!queue.isEmpty()) {
     String job = queue.poll();
     System.out.println("Processing job: " + job);
  } else {
     System.out.println("No jobs to process.");
  }
}
public void viewJobs() {
  if (queue.isEmpty()) {
     System.out.println("No pending jobs.");
  } else {
     System.out.println("Pending jobs: " + queue);
  }
}
public static void main(String[] args) {
  PrintQueue printer = new PrintQueue();
  Scanner scanner = new Scanner(System.in);
  while (true) {
     System.out.println("\n1. Add Job\n2. Process Job\n3. View Jobs\n4. Exit");
```

```
int choice = scanner.nextInt();
        scanner.nextLine();
        switch (choice) {
          case 1:
             System.out.print("Enter job name: ");
             String job = scanner.nextLine();
             printer.addJob(job);
             break;
          case 2:
             printer.processJob();
             break;
          case 3:
             printer.viewJobs();
             break;
          case 4:
             System.exit(0);
       }
     }
  }
}
```



```
import java.util.*;

public class UndoRedoManager {
   Stack<String> undoStack;
   Stack<String> redoStack;

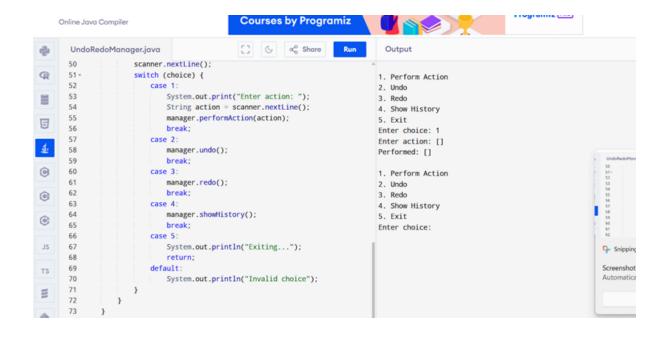
public UndoRedoManager() {
   undoStack = new Stack<>();
   redoStack = new Stack<>();
```

}

```
public void performAction(String action) {
  undoStack.push(action);
  redoStack.clear();
  System.out.println("Performed: " + action);
}
public void undo() {
  if (undoStack.isEmpty()) {
     System.out.println("Nothing to undo.");
     return;
  }
  String action = undoStack.pop();
  redoStack.push(action);
  System.out.println("Undone: " + action);
}
public void redo() {
  if (redoStack.isEmpty()) {
     System.out.println("Nothing to redo.");
     return;
  }
  String action = redoStack.pop();
  undoStack.push(action);
```

```
System.out.println("Redone: " + action);
}
public void showHistory() {
  System.out.println("Undo Stack: " + undoStack);
  System.out.println("Redo Stack: " + redoStack);
}
public static void main(String[] args) {
  UndoRedoManager manager = new UndoRedoManager();
  Scanner scanner = new Scanner(System.in);
  while (true) {
     System.out.println("\n1. Perform Action\n2. Undo\n3. Redo\n4. Show History\n5. Exit");
     System.out.print("Enter choice: ");
     int choice = scanner.nextInt();
     scanner.nextLine();
     switch (choice) {
       case 1:
          System.out.print("Enter action: ");
          String action = scanner.nextLine();
          manager.performAction(action);
          break;
       case 2:
          manager.undo();
```

```
break;
          case 3:
            manager.redo();
            break;
          case 4:
            manager.showHistory();
            break;
          case 5:
            System.out.println("Exiting...");
            return;
          default:
            System.out.println("Invalid choice");
       }
    }
  }
}
```



```
import java.util.*;

public class TicketBookingSystem {
    Queue<String> bookingQueue;

public TicketBookingSystem() {
    bookingQueue = new LinkedList<>();
    }

public void addPerson(String name) {
    bookingQueue.offer(name);
    System.out.println(name + " added to the booking queue.");
}
```

```
public void serveNext() {
  if (bookingQueue.isEmpty()) {
     System.out.println("No one in the queue.");
     return;
  }
  String name = bookingQueue.poll();
  System.out.println(name + " has been served.");
}
public void cancelTicket(String name) {
  if (bookingQueue.remove(name)) {
     System.out.println("Ticket cancelled for " + name);
  } else {
     System.out.println(name + " not found in queue.");
  }
}
public void displayQueue() {
  if (bookingQueue.isEmpty()) {
     System.out.println("Booking queue is empty.");
  } else {
     System.out.println("Current booking queue:");
     for (String person : bookingQueue) {
       System.out.println(person);
```

```
}
    }
  }
  public static void main(String[] args) {
     TicketBookingSystem system = new TicketBookingSystem();
     Scanner scanner = new Scanner(System.in);
    while (true) {
       System.out.println("\n1. Add Person\n2. Serve Next\n3. Cancel Ticket\n4. Display
Queue\n5. Exit");
       System.out.print("Enter choice: ");
       int choice = scanner.nextInt();
       scanner.nextLine();
       switch (choice) {
         case 1:
            System.out.print("Enter name: ");
            String name = scanner.nextLine();
            system.addPerson(name);
            break;
         case 2:
            system.serveNext();
            break;
         case 3:
            System.out.print("Enter name to cancel: ");
            String cancelName = scanner.nextLine();
```

```
system.cancelTicket(cancelName);
break;
case 4:
system.displayQueue();
break;
case 5:
System.out.println("Exiting...");
return;
default:
System.out.println("Invalid choice");
}
}
}
```

```
TicketBookingSystem.java
                                                                            Output
52
                       System.out.print("Enter name: ");
53
                                                                           1. Add Person
54
                       String name = scanner.nextLine();
                                                                           2. Serve Next
55
                        system.addPerson(name);
                                                                          3. Cancel Ticket
56
                       break;
                                                                           4. Display Queue
57
                    case 2:
                                                                           5. Exit
58
                        system.serveNext();
                                                                           Enter choice: 1
59
                        break;
                                                                           Enter name: xyz
60
                                                                           xyz added to the booking queue.
61
                       System.out.print("Enter name to cancel: ");
                       String cancelName = scanner.nextLine();
62
                                                                          1. Add Person
63
                        system.cancelTicket(cancelName);
                                                                          2. Serve Next
64
                       break;
                                                                          3. Cancel Ticket
65
                    case 4:
                                                                           4. Display Queue
66
                        system.displayQueue();
                                                                           5. Exit
67
                                                                           Enter choice: 2
                                                                           xyz has been served.
69
                       System.out.println("Exiting...");
70
                        return;
                                                                           1. Add Person
71
                    default:
72
                       System.out.println("Invalid choice");
                                                                           3. Cancel Ticket
73
                                                                           4. Display Queue
74
                                                                           5. Exit
75
                                                                           Enter choice:
                                                     Q Search
```