



NEW YORK CITY COLLEGE OF TECHNOLOGY THE CITY UNIVERSITY OF NEW YORK
Department of Computer Engineering Technology 300 Jay Street, Brooklyn, NY 11201-1909

LAB REPORT

CET 3640 – OL30

**(SOFTWARE FOR COMPUTER
CONTROL)**

LAB#6

JAVA PROGRAM INTERFACES

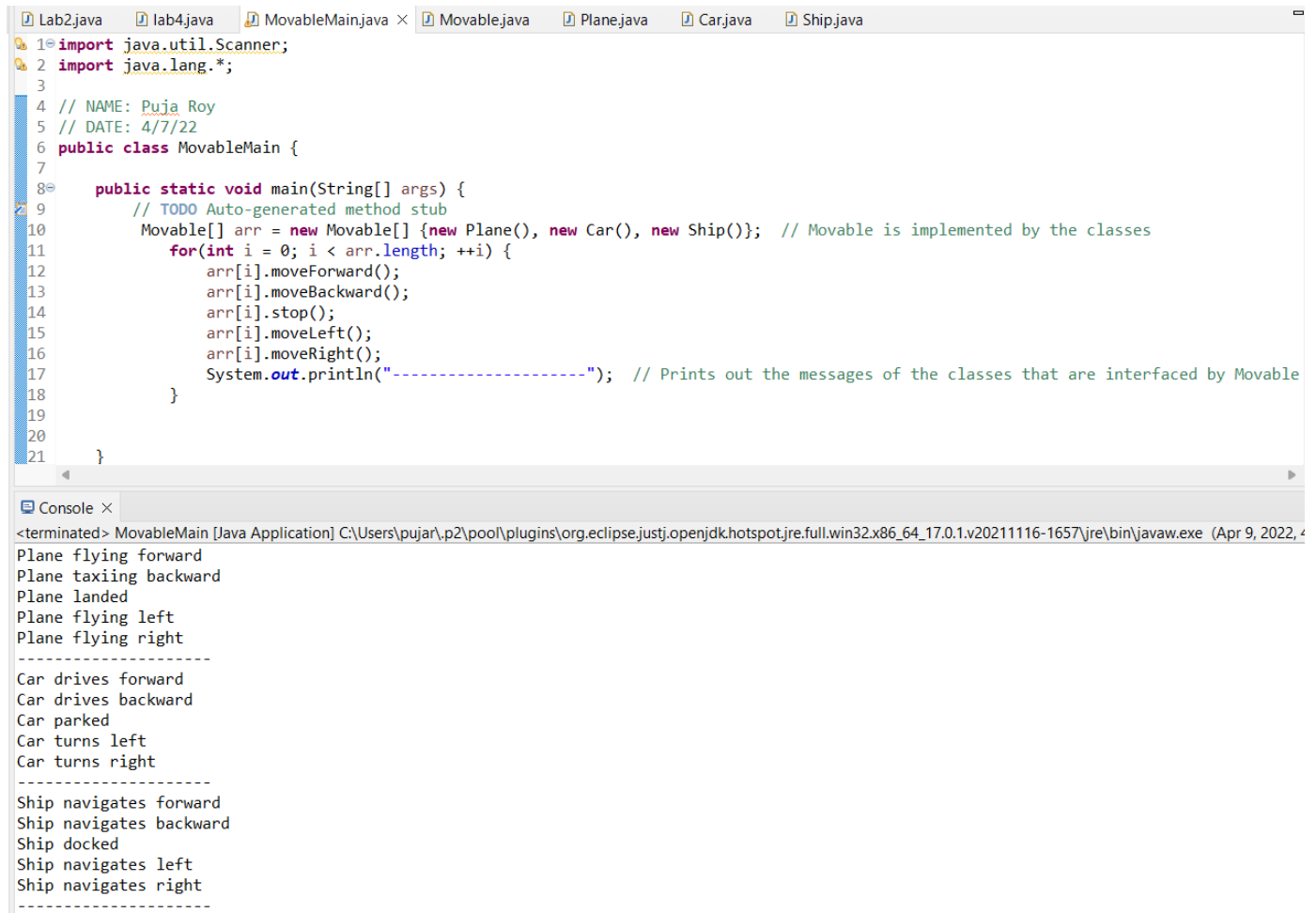
Name: Puja Roy

Date: 4/7/22

Due Date: 4/9/22

DESCRIPTION OF THE LAB:

In this lab, I wrote a java program in Eclipse that includes interfaces, methods, and classes. An interface in Java is an abstract type to specify the behavior and output that classes implement. First, I created a java project and named it MovableMain. Then, I right clicked on the source file of the Lab6 java project file and created an interface named Movable. Then, I created 3 classes Plane, Car and Ship that implements the interface Movable. Each of the classes implement the interface Movable. While writing the java code to implement the classes successfully, I had to save the file of each of the classes. Then, I had to right click on each class and select “add unimplemented methods” which automatically processed the methods. I created functions in the Java program that prints messages regarding the action and navigations of the classes Plane, Car, and Ship.



The screenshot displays the Eclipse IDE with several Java files open in the editor: Lab2.java, lab4.java, MovableMain.java, Movable.java, Plane.java, Car.java, and Ship.java. The MovableMain.java file is the active editor, showing the following code:

```
1 import java.util.Scanner;
2 import java.lang.*;
3
4 // NAME: Puja Roy
5 // DATE: 4/7/22
6 public class MovableMain {
7
8     public static void main(String[] args) {
9         // TODO Auto-generated method stub
10        Movable[] arr = new Movable[] {new Plane(), new Car(), new Ship()}; // Movable is implemented by the classes
11        for(int i = 0; i < arr.length; ++i) {
12            arr[i].moveForward();
13            arr[i].moveBackward();
14            arr[i].stop();
15            arr[i].moveLeft();
16            arr[i].moveRight();
17            System.out.println("-----"); // Prints out the messages of the classes that are interfaced by Movable
18        }
19
20    }
21 }
```

Below the editor, the Console window shows the output of the program, which is terminated. The output consists of messages for each class (Plane, Car, and Ship) and is separated by dashed lines:

```
<terminated> MovableMain [Java Application] C:\Users\pujar\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (Apr 9, 2022, 4
Plane flying forward
Plane taxiing backward
Plane landed
Plane flying left
Plane flying right
-----
Car drives forward
Car drives backward
Car parked
Car turns left
Car turns right
-----
Ship navigates forward
Ship navigates backward
Ship docked
Ship navigates left
Ship navigates right
-----
```

```
Lab2.java  lab4.java  MovableMain.java  Movable.java ×  Plane.java  Car.java  Ship.java
1 // NAME: Puja Roy
2 // DATE: 4/7/22
3
4 public interface Movable { // Interface that includes methods for the classes
5     public void moveForward();
6     public void moveBackward();
7     public void stop();
8     public void moveLeft();
9     public void moveRight();
10 }
11
```

```
Lab2.java  lab4.java  MovableMain.java  Movable.java  Plane.java ×  Car.java  Ship.java
1 // NAME: Puja Roy
2 // DATE: 4/7/22
3
4 public class Plane implements Movable { // Plane is the class that implements the interface Movable
5
6     @Override
7     public void moveForward() {
8         System.out.println("Plane flying forward"); // Method that prints out message
9     }
10
11     @Override
12     public void moveBackward() {
13         System.out.println("Plane taxiing backward"); // Method that prints out message
14     }
15
16     @Override
17     public void stop() {
18         System.out.println("Plane landed"); // Method that prints out message
19     }
20
21     @Override
22     public void moveLeft() {
23         System.out.println("Plane flying left"); // Method that prints out message
24     }
25
26     @Override
27     public void moveRight() {
28         System.out.println("Plane flying right"); // Method that prints out message
29     }
30
31 }
32
```

```
Lab2.java  lab4.java  MovableMain.java  Movable.java  Plane.java  Car.java ×  Ship.java
1  // NAME: Puj Roy
2  // DATE: 4/7/22
3
4  public class Car implements Movable { // Car is the class that implements the interface Movable
5
6      @Override
7      public void moveForward() {
8          System.out.println("Car drives forward"); // Method that prints out message
9      }
10
11      @Override
12      public void moveBackward() {
13          System.out.println("Car drives backward"); // Method that prints out message
14      }
15
16      @Override
17      public void stop() {
18          System.out.println("Car parked"); // Method that prints out message
19      }
20
21      @Override
22      public void moveLeft() {
23          System.out.println("Car turns left"); // Method that prints out message
24      }
25
26      @Override
27      public void moveRight() {
28          System.out.println("Car turns right"); // Method that prints out message
29      }
30
31  }
```

```

Lab2.java  lab4.java ×  MovableMain.java  Movable.java  Plane.java  Car.java  Ship.java ×
1  // NAME: Pujja Roy
2  // DATE: 4/7/22
3
4  public class Ship implements Movable { // Ship is the class that implements the interface Movable
5
6      @Override
7      public void moveForward() {
8          System.out.println("Ship navigates forward"); // Method that prints out message
9      }
10
11     @Override
12     public void moveBackward() {
13         System.out.println("Ship navigates backward"); // Method that prints out message
14     }
15
16     @Override
17     public void stop() {
18         System.out.println("Ship docked"); // Method that prints out message
19     }
20
21     @Override
22     public void moveLeft() {
23         System.out.println("Ship navigates left"); // Method that prints out message
24     }
25
26     @Override
27     public void moveRight() {
28         System.out.println("Ship navigates right"); // Method that prints out message
29     }
30
31 }

```

UML Diagram:

