Note: This document is a modification of the Quick Start Guide at the NetBeans website (http://netbeans.org/kb/docs/java/quickstart.html)

Assumptions
I have assumed that you do not have any experiences in using NetBeans.

Software
To finish this lab, my may need the following software

1. OpenJDK 8/JDK version 1.8.0 (jdk1.8.0_202, or later)
2. NetBeans IDE version 12.2 (or 12.5)

Aim
This lab aims to guide you through using NetBeans as an IDE for program development.

In this lab task, we will create a VehicleLib project with a utility class, then create a VehicleHireApp project with a main class that calls a method from the VehicleLib project. This document is designed to get you going as quickly as possible. For more information on working with NetBeans IDE, see the Support and Docs page (http://netbeans.org/kb/index.html) on the NetBeans website.

Overview of the Tasks
In this Lab, you will learn to use NetBeans to write Java programs:

LT1. Create a new Java Class Library
LT2. Create a new Java Application
LT3. Add the Java Class Library's project jar to the Java Application project
LT4. Program the required Java class in the Java Class Library
LT5. Program the Java Application
LT6. Run the Java Application

Lab Tasks

This Lab should be run on MS Windows Platform

LT1. Create a new Java class library

LT1.1. Choose "File" > "New Project…"
LT1.2. Select "Java" under "Categories"
LT1.3. Select "Java Class Library" under Projects
LT1.4. Click "Next"
LT1.5. In the "New Class Library" window, enter "VehicleLib" under "Project Name"
LT1.6. [Optional] Change the Project Location to any directory on your computer. e.g. `C:\Users\s1234567\COS30041\Labs`. From now on, we will refer to this directory as `${NetBeans_Projects}`. e.g. `${NetBeans_Projects}` represents `C:\Users\s1234567\COS30041\Labs\`
Note: The path specified above should appear as "`\${NetBeans_Projects}`\VehicleLib\" i..e. "`C:\Users\s1234567\COS30041\Labs` \VehicleLib\"
LT1.7. Click "Finish"
Note: The VehicleLib project opens in both the Projects window and the Files window.

LT2. Create a New Java Application

LT2.1. Choose "File" > "New Project…"
LT2.2. Select "Java" under "Categories"
LT2.3. Select "Java Application" under Projects
LT2.4. Click "Next"
LT2.5. In the "New Java Application" window
a. Enter "VehicleHireApp" under "Project Name"
b. Make sure the Project Location is set to "`${NetBeans_Projects}`"

Note: "vehiclehireapp.VehicleHireApp" has already been entered in the "Create Main Class" textfield

c. Ensure that the "Set as Main Project" and "Create Main Class" checkboxes are checked

d. Click "Finish"

Note: The VehicleHireApp project is displayed in the Project window and the file VehicleHireApp.java is opened in the Source Editor.

LT3. Add a project jar file to another project

LT3.1. Select "VehicleHireApp" project, expand on the node

LT3.2. Right-click the "Libraries" node

LT3.3. Choose "Add Project…"

LT3.4. Select the "VehicleLib" project (If needed, browse through `${NetBeans_Projects}` and select the VehicleLib project folder). The Project JAR Files textarea shows the JAR files that will be added to the project.

Note: A JAR file for VehicleLib is listed even though we have not actually built the JAR file yet. This JAR file will get built when we build and run the VehicleHireApp project.

LT3.5. Click "Add Project JAR Files".

LT3.6. Expand the Libraries node.

Note: The VehicleLib project's JAR file is added to the VehicleHireApp project's classpath.

Now we need to create a Java package and add the method that we'll use to construct a vehicle, after which we'll implement the vehicle method in the VehicleHireApp class.

LT4. Program the required Java class files in the class library

LT4.1. Program the "VehicleType.java" class file

a. Right-click the "VehicleLib" project node

b. Choose "New" > "Java Class…"

c. Type "VehicleType" as the Class name for the new class

d. Type "au.edu.swin.vehicle" in the Package field

e. Click "Finish"

Note: VehicleType.java opens in the Source Editor

f. Type or paste the following code segment in VehicleType.java (within the class)

```
private String code;
private String description;
private Integer seats;

public VehicleType(String code, String description, Integer seats) {
    this.code = code;
    this.description = description;
    this.seats = seats;}

public String getCode() {
    return code.toUpperCase();
}

public void setCode(String code) {
    this.code = code;
}

public String getDescription() {
    return description;
}

public void setDescription(String description) {
    this.description = description;
}

public Integer getSeats() {
```

```
    return seats;
}

public void setSeats(Integer seats) {
    this.seats = seats;
}
```

g. Format the code by pressing “Alt-Shift-F” (on Windows), “Ctrl-Shift-F” (on Mac)

h. Save the file

LT4.2. Program the “Vehicle.java” class

a. Right-click the “VehicleLib” project node

b. Choose “New” > “Java Class…”

c. Enter “Vehicle” as the Class name for the new class

d. Enter “au.edu.swin.vehicle” in the Package field

e. Click “Finish”

Note: Vehicle.java opens in the Source Editor.

f. Type or paste the following code segment in Vehicle.java

```
private String name;
private String colour;
private VehicleType type;
private Integer year;

/** Creates a new instance of Vehicle */
public Vehicle(String name, String colour, VehicleType type, Integer year){
    this.name = name;
    this.colour = colour;
    this.type = type;
    this.year = year;}

public String getColour() {
    return colour;
}

public void setColour(String colour) {
    this.colour = colour;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public VehicleType getType() {
    return type;
}

public void setType(VehicleType type) {
    this.type = type;
}

public Integer getYear() {
    return year;
}

public void setYear(Integer year) {
    this.year = year;
}

public String toString() {
```

```
    String objString = String.format("%s %s (%s) %d", this.getName(),
        this.getColour(), this.getType().getDescription(), this.getYear());
    return objString;
}
```

Remember to format the code and save the file.

LT5. Program the main application "VehicleHireApp.java"

LT5.1. Select the "VehicleHireApp.java" tab in the Source Editor.

Note: If it isn't already open, expand "VehicleHireApp" > "Source Packages" > "vehiclehireapp" in the Projects window and double-click "VehicleHireApp.java".

LT5.2. Type or paste the following code segment in VehicleHireApp.java

```
/**
  * @param args the command line arguments
  */
public static void main(String[] args) {
    // Create the vehicle types
    VehicleType sedan = new VehicleType("SEDAN", "A standard sedan", 4);
    VehicleType limo6 = new VehicleType("LIMO6", "A six seater limo", 6);
    VehicleType limo8 = new VehicleType("LIMO8", "An eight seater limo", 8);

    // Create the vehicles
    ArrayList<Vehicle> vehicles = new ArrayList();
    vehicles.add(new Vehicle("Ed's Holden Caprice", "Silver", sedan, 2002));
    vehicles.add(new Vehicle("John's Mercedes C200", "Black", sedan, 2005));
    vehicles.add(new Vehicle("Guy's Volvo 244 DL", "Blue", sedan, 1976));
    vehicles.add(new Vehicle("Sasco's Ford Limo", "White", limo6, 2014));
    vehicles.add(new Vehicle("Peter's Ford Limo", "White", limo6, 2004));
    vehicles.add(new Vehicle("Robert's Ford Limo", "White", limo8, 2003));

    System.out.println("\n\nList of vehicles in system:");
    for(Vehicle vehicle : vehicles) {
        System.out.println(vehicle);
    }

    String typeCode = args[0];
    System.out.println("\n\nList of vehicle of type " + typeCode);
    for(Vehicle vehicle : vehicles) {
        if(vehicle.getType().getCode().equals(typeCode)) {
            System.out.println(vehicle);
    }
}
```

LT5.3. Import the appropriate class from Java's SDK or the library you have just added in LT3 by pressing Ctrl-Shift-I (the uppercase "I" not one)

Note: Alternatively, type or paste the following import statements in "VehicleHireApp.java".

```
import au.edu.swin.vehicle.Vehicle;
import au.edu.swin.vehicle.VehicleType;
import java.util.ArrayList;
```

LT6. Run a Java application

LT6.1. Set the main class and execution arguments

a. Right-click the "VehicleHireApp" project node
b. Choose "Properties"
c. Select the Run node in the dialogue's left pane under Categories
   Note: The main class is already set to "`vehiclehireapp.VehicleHireApp`".
d. Enter "SEDAN" in the Arguments field (note: all uppercase letters)
e. Click OK

LT6.2. Run the Java application

a. Right-click the "VehicleHireApp" project node
b. Choose "Run"

Note: To see the output, Double-click the Output window to maximize it so you can see all the output.

Note: Ant builds VehicleLib.jar first and then uses it to compile VehicleHireApp. Finally, it prints the output from the program.

LT6.3. Run the Java application again with "sedan" in the Arguments field (see LT6.1). Observe what happen.

Practice Task:

Extend the code "VehicleHireApp.java" by adding a user menu for choosing a vehicle type. After a vehicle type is chosen, a list of vehicles for this type is displayed. A sample run is shown below:

```
run:


List of vehicles in system:
Ed's Holden Caprice Silver (A standard sedan) 2002
John's Mercedes C200 Black (A standard sedan) 2005
Guy's Volvo 244 DL Blue (A standard sedan) 1976
Sasco's Ford Limo White (A six seater limo) 2014
Peter's Ford Limo Black (A six seater limo) 2004
Robert's Ford Limo White (An eight seater limo) 2003


List of vehicle of type SEDAN
Ed's Holden Caprice Silver (A standard sedan) 2002
John's Mercedes C200 Black (A standard sedan) 2005
Guy's Volvo 244 DL Blue (A standard sedan) 1976

It will display a list of vehicles based on the vehicle type you choose:
1: SEDAN
2: LIMO6
3: LIMO8
4: Exit

Please select an option (1-4): 2
Sasco's Ford Limo White (A six seater limo) 2014
Peter's Ford Limo Black (A six seater limo) 2004

1: SEDAN
2: LIMO6
3: LIMO8
4: Exit

Please select an option (1-4): 4
BUILD SUCCESSFUL (total time: 12 seconds)
```