Department of Computer Science & Engineering
UNIVERSITY OF LIBERAL ARTS
BANGLADESH

Lab Report 5

Submitted to:
Shakib Mahmud Dipto
Faculty

Submitted by:
Sumaiya Akter
ID: 201014071

Department of CSE
Summer'24

Course code: CSE 2104
Course Title: Object Oriented Programming Lab
Section: 01

University of Liberal Arts Bangladesh

August 08, 2024

Problem 01

Code Explanation :

The provided Java code defines a Car class with properties such as make, model, year, color, and price. It includes constructor overloading to initialize cars with different sets of information. The displayCarInfo() method outputs car details, while changeCar() methods allow updating the car's color and price. The Main class creates instances of Car, displays their information, modifies some attributes, and demonstrates the functionality of the class through method calls.

Code Screenshots:

Input:

Main.java × Car.java ×

Source History

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
 */

package main;

/**
 *
 * @author ANIK
 */
public class Main  {

    public static void main(String[] args) {
        Car car1 = new Car("Toyota", "Camry", 2022, "Red", 25000);
        Car car2 = new Car("Honda", "Accord", 2021);

        car1.displayCarInfo();
        car2.displayCarInfo();

        System.out.println("Hello World!");

        car1.changeCar("Yellow");
        car2.changeCar("Blue", 30000);

        car1.displayCarInfo();
        car2.displayCarInfo();
    }
}
```

Main.java × Car.java ×

Source History

```java
package main;

/**
 *
 * @author ANIK
 */
public class Car {
    private String make;
    private String model;
    private int year;
    private String color;
    private double price;

    // Constructor with all parameters
    public Car(String make, String model, int year, String color, double price) {
        this.make = make;
        this.model = model;
        this.year = year;
        this.color = color;
        this.price = price; // Initialize price
    }

    // Constructor with default color and price
    public Car(String make, String model, int year) {
        this(make, model, year, "Unknown", 0.0);
    }

    public void displayCarInfo() {
        System.out.println("Car Information:");
        System.out.println("Make: " + this.make);
        System.out.println("Model: " + this.model);
        System.out.println("Year: " + this.year);
        System.out.println("Color: " + this.color);
        System.out.println("Price: $" + this.price);
    }

    public void changeCar(String color) {
        this.color = color;
    }

    public void changeCar(String color, double price) {
        this.color = color;
        this.price = price; // Update price as well
    }
}
```

Output:

Output - Run (Main)

```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 5\code\Main; "JAVA_HOME=C:\\
Scanning for projects...

------------------------< com.mycompany:Main >------------------------
Building Main 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ Main ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments

--- compiler:3.11.0:compile (default-compile) @ Main ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ Main ---
Car Information:
Make: Toyota
Model: Camry
Year: 2022
Color: Red
Price: $25000.0
Car Information:
Make: Honda
Model: Accord
Year: 2021
Color: Unknown
Price: $0.0
Hello World!
Car Information:
Make: Toyota
Model: Camry
Year: 2022
Color: Yellow
Price: $25000.0
Car Information:
Make: Honda
Model: Accord
Year: 2021
Color: Blue
Price: $30000.0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.557 s
Finished at: 2024-07-28T15:13:53+06:00
------------------------------------------------------------------------
```

Practice Problem 01

Code Explanation:

The Person class represents an individual with properties such as name, age, gender, and address. It includes three constructors for different initialization scenarios: one for all properties, one for name and age, and another for just the name. The class features methods to display personal information and to change the address or both age and address. A main method demonstrates creating instances of Person and invoking these methods to showcase their functionality.

Code Screenshot:

Input:

Person.java

Source History

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.
 */

package person;

/**
 *
 * @author ANIK
 */
public class Person  {
    private String name;
    private int age;
    private String gender;
    private String address;

    // Constructor with all parameters
    public Person(String name, int age, String gender, String address) {
        this.name = name;
        this.age = age;
        this.gender = gender;
        this.address = address;
    }

    // Constructor with name and age
    public Person(String name, int age) {
        this(name, age, "Unknown", "Unknown");
    }

    // Constructor with name
    public Person(String name) {
        this(name, 0, "Unknown", "Unknown");
    }

    // Method to display person information
    public void displayPersonInfo() {
        System.out.println("Person Information:");
        System.out.println("Name: " + this.name);
        System.out.println("Age: " + this.age);
        System.out.println("Gender: " + this.gender);
        System.out.println("Address: " + this.address);
    }

    // Method to change address
    public void changeAddress(String address) {
        this.address = address;
    }
```

```java

    // Method to change age and address
    public void changeAgeAndAddress(int age, String address) {
        this.age = age;
        this.address = address;
    }

    // Main method for testing
    public static void main(String[] args) {
        Person person1 = new Person("Alice", 30, "Female", "123 Main St");
        Person person2 = new Person("Bob", 25);
        Person person3 = new Person("Charlie");

        person1.displayPersonInfo();
        person2.displayPersonInfo();
        person3.displayPersonInfo();

        // Changing address and age for person1
        person1.changeAddress("456 Elm St");
        person1.changeAgeAndAddress(31, "456 Elm St");

        System.out.println("\nUpdated Information for Person 1:");
        person1.displayPersonInfo();
    }
}
```

Output:

Output - Run (Person)

```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 5\code\Person; "JAVA_HOME=C:\\Progr
Scanning for projects...

------------------------< com.mycompany:Person >------------------------
Building Person 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ Person ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 5\

--- compiler:3.11.0:compile (default-compile) @ Person ---
Changes detected - recompiling the module! :source
Compiling 1 source file with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ Person ---
Person Information:
Name: Alice
Age: 30
Gender: Female
Address: 123 Main St
Person Information:
Name: Bob
Age: 25
Gender: Unknown
Address: Unknown
Person Information:
Name: Charlie
Age: 0
Gender: Unknown
Address: Unknown

Updated Information for Person 1:
Person Information:
Name: Alice
Age: 31
Gender: Female
Address: 456 Elm St
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.834 s
Finished at: 2024-08-06T23:11:23+06:00
------------------------------------------------------------------------
```

Practice Problem 02

Code Explanation:

The Employee class models an employee with properties like name, id, salary, and designation. It features constructor overloading, allowing initialization with varying parameters: all properties, just name and id, or only name. The class includes methods to display employee information and update salary or designation individually or together. A main method demonstrates creating employee instances and showcasing the functionality of the class through method calls, illustrating its design and usability.

Code Screenshot:

Input:

Employee.java

Source History

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 */

package employee;

/**
 *
 * @author ANIK
 */
public class Employee {
    private String name;
    private int id;
    private double salary;
    private String designation;

    // Constructor with all parameters
    public Employee(String name, int id, double salary, String designation) {
        this.name = name;
        this.id = id;
        this.salary = salary;
        this.designation = designation;
    }

    // Constructor with name and id
    public Employee(String name, int id) {
        this(name, id, 0.0, "Unknown");
    }

    // Constructor with name
    public Employee(String name) {
        this(name, 0, 0.0, "Unknown");
    }

    // Method to display employee information
    public void displayEmployeeInfo() {
        System.out.println("Employee Information:");
        System.out.println("Name: " + this.name);
        System.out.println("ID: " + this.id);
        System.out.println("Salary: $" + this.salary);
        System.out.println("Designation: " + this.designation);
    }

    // Method to update salary
    public void updateSalary(double salary) {
        this.salary = salary;
    }
```

```java

    // Method to update designation
    public void updateDesignation(String designation) {
        this.designation = designation;
    }

    // Method to update both salary and designation
    public void updateSalaryAndDesignation(double salary, String designation) {
        this.salary = salary;
        this.designation = designation;
    }

    // Main method for testing
    public static void main(String[] args) {
        Employee emp1 = new Employee("Alice", 101, 75000, "Software Engineer");
        Employee emp2 = new Employee("Bob", 102);
        Employee emp3 = new Employee("Charlie");

        emp1.displayEmployeeInfo();
        emp2.displayEmployeeInfo();
        emp3.displayEmployeeInfo();

        // Updating salary and designation for emp1
        emp1.updateSalary(80000);
        emp1.updateDesignation("Senior Software Engineer");

        System.out.println("\nUpdated Information for Employee 1:");
        emp1.displayEmployeeInfo();
    }
}
```

Output:

Output - Run (Employee)

```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab 5\code\Employee; "JAVA_HOME=C:\\
Scanning for projects...

-----------------------< com.mycompany:Employee >-----------------------
Building Employee 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ Employee ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\lab

--- compiler:3.11.0:compile (default-compile) @ Employee ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ Employee ---
Employee Information:
Name: Alice
ID: 101
Salary: $75000.0
Designation: Software Engineer
Employee Information:
Name: Bob
ID: 102
Salary: $0.0
Designation: Unknown
Employee Information:
Name: Charlie
ID: 0
Salary: $0.0
Designation: Unknown

Updated Information for Employee 1:
Employee Information:
Name: Alice
ID: 101
Salary: $80000.0
Designation: Senior Software Engineer
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.543 s
Finished at: 2024-08-06T23:24:35+06:00
------------------------------------------------------------------------
```