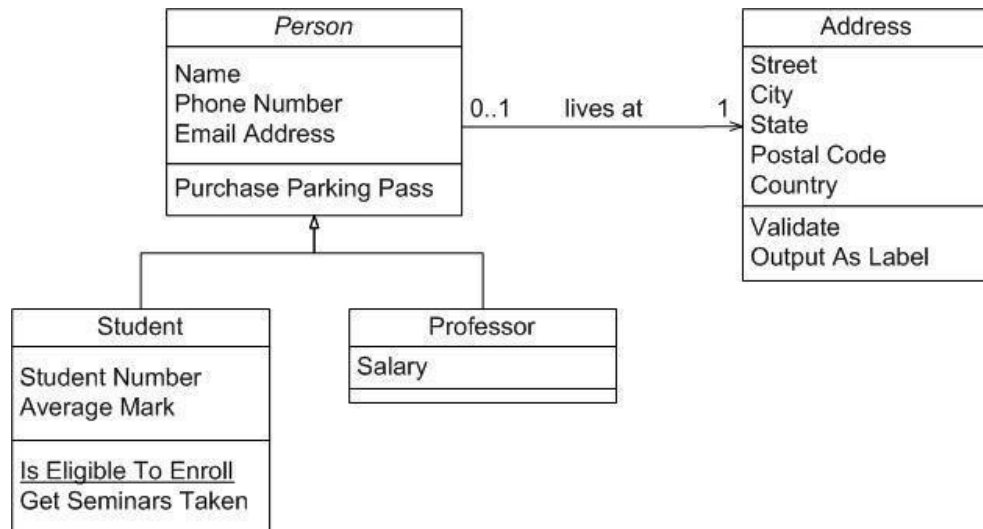


Name : Evan Diantha Fafian
 Class : SIB 2G
 Absent : 09
 NIM : 2341760163

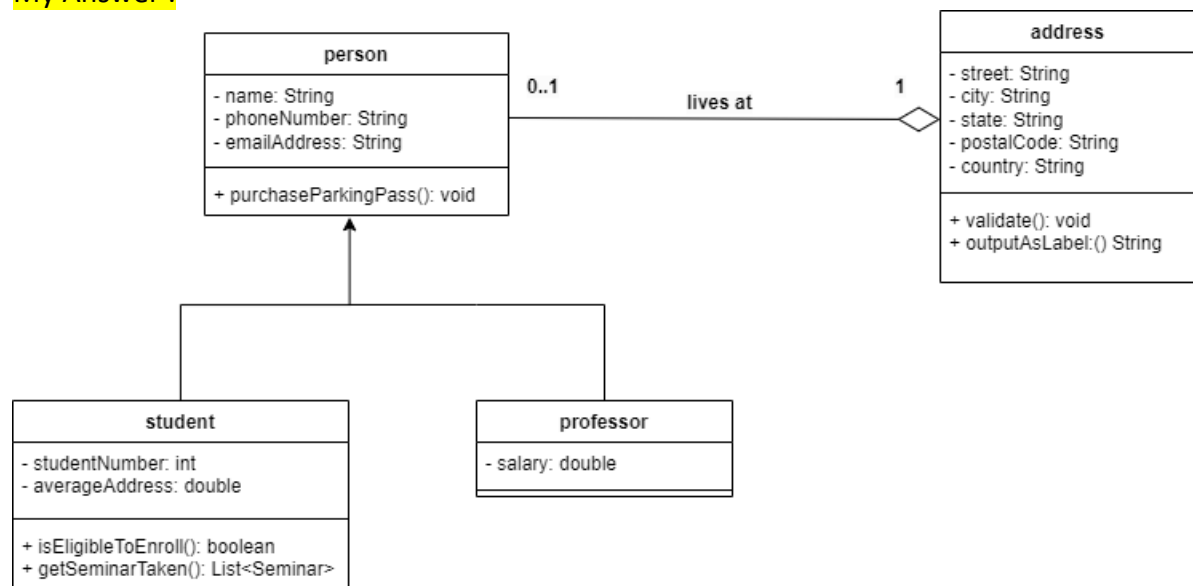
UTS QUESTIONS

OBJECT-BASED PROGRAMMING PRACTICUM

1. Identify the following diagram class, make complete improvements and in accordance with the rules for writing the diagram class.

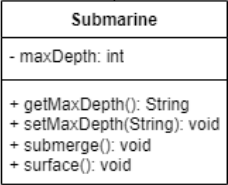
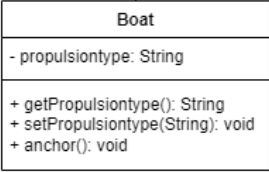
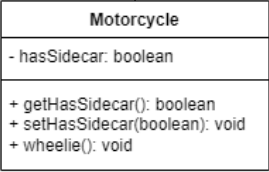
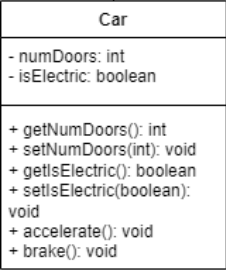
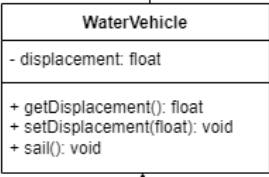
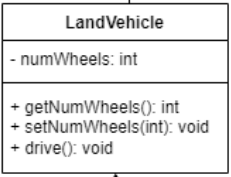
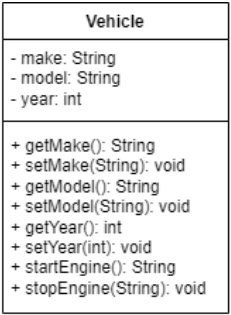


My Answer :



2. Create a diagram class that uses multilevel inheritance and create the program code!

My Answer :



- Vehicle

```
1 package week_8_UTS.question_2;
2
3 public class Vehicle {
4     private String make;
5     private String model;
6     private int year;
7
8     public Vehicle(String make, String model, int year) {
9         this.make = make;
10        this.model = model;
11        this.year = year;
12    }
13
14    public String getMake() {
15        return make;
16    }
17
18    public void setMake(String make) {
19        this.make = make;
20    }
21
22    public String getModel() {
23        return model;
24    }
25
26    public void setModel(String model) {
27        this.model = model;
28    }
29
30    public int getYear() {
31        return year;
32    }
33
34    public void setYear(int year) {
35        this.year = year;
36    }
37
38    public void startEngine() {
39        System.out.println("Starting the engine of " + make + " " + model);
40    }
41
42    public void stopEngine() {
43        System.out.println("Stopping the engine of " + make + " " + model);
44    }
45 }
```

- LandVehicle

```
1 package week_8_UTS.question_2;
2
3 public class LandVehicle extends Vehicle {
4     private int numWheels;
5
6     public LandVehicle(String make, String model, int year, int numWheels) {
7         super(make, model, year);
8         this.numWheels = numWheels;
9     }
10
11     public int getNumWheels() {
12         return numWheels;
13     }
14
15     public void setNumWheels(int numWheels) {
16         this.numWheels = numWheels;
17     }
18
19     public void drive() {
20         System.out.println("Driving the " + getMake() + " " + getModel() + " on " + numWheels + " wheels");
21     }
22 }
23
```

- WaterVehicle

```
1 package week_8_UTS.question_2;
2
3 public class WaterVehicle extends Vehicle {
4     private float displacement;
5
6     public WaterVehicle(String make, String model, int year, float displacement) {
7         super(make, model, year);
8         this.displacement = displacement;
9     }
10
11     public float getDisplacement() {
12         return displacement;
13     }
14
15     public void setDisplacement(float displacement) {
16         this.displacement = displacement;
17     }
18
19     public void sail() {
20         System.out.println("Sailing the " + getMake() + " " + getModel() + " with " + displacement + " displacement");
21     }
22 }
```

- Car

```
1 package week_8_UTS.question_2;
2
3 public class Car extends LandVehicle {
4     private int numDoors;
5     private boolean isElectric;
6
7     public Car(String make, String model, int year, int numWheels, int numDoors, boolean isElectric) {
8         super(make, model, year, numWheels);
9         this.numDoors = numDoors;
10        this.isElectric = isElectric;
11    }
12
13    public int getNumDoors() {
14        return numDoors;
15    }
16
17    public void setNumDoors(int numDoors) {
18        this.numDoors = numDoors;
19    }
20
21    public boolean isElectric() {
22        return isElectric;
23    }
24
25    public void setElectric(boolean electric) {
26        isElectric = electric;
27    }
28
29    public void accelerate() {
30        System.out.println("Accelerating the " + getMake() + " " + getModel());
31    }
32
33    public void brake() {
34        System.out.println("Braking the " + getMake() + " " + getModel());
35    }
36 }
```

- Motorcycle

```
1 package week_8_UTS.question_2;
2
3 public class Motorcycle extends LandVehicle {
4     private boolean hasSidecar;
5
6     public Motorcycle(String make, String model, int year, int numWheels, boolean hasSidecar) {
7         super(make, model, year, numWheels);
8         this.hasSidecar = hasSidecar;
9     }
10
11    public boolean hasSidecar() {
12        return hasSidecar;
13    }
14
15    public void setHasSidecar(boolean hasSidecar) {
16        this.hasSidecar = hasSidecar;
17    }
18
19    public void wheelie() {
20        if (!hasSidecar) {
21            System.out.println("Performing a wheelie on the " + getMake() + " " + getModel());
22        } else {
23            System.out.println("Cannot perform a wheelie with a sidecar on the " + getMake() + " " + getModel());
24        }
25    }
26 }
```

- Boat

```
1 package week_8_UTS.question_2;
2
3 public class Boat extends WaterVehicle {
4     private String propulsionType;
5
6     public Boat(String make, String model, int year, float displacement, String propulsionType) {
7         super(make, model, year, displacement);
8         this.propulsionType = propulsionType;
9     }
10
11     public String getPropulsionType() {
12         return propulsionType;
13     }
14
15     public void setPropulsionType(String propulsionType) {
16         this.propulsionType = propulsionType;
17     }
18
19     public void anchor() {
20         System.out.println("Anchoring the " + getMake() + " " + getModel());
21     }
22 }
```

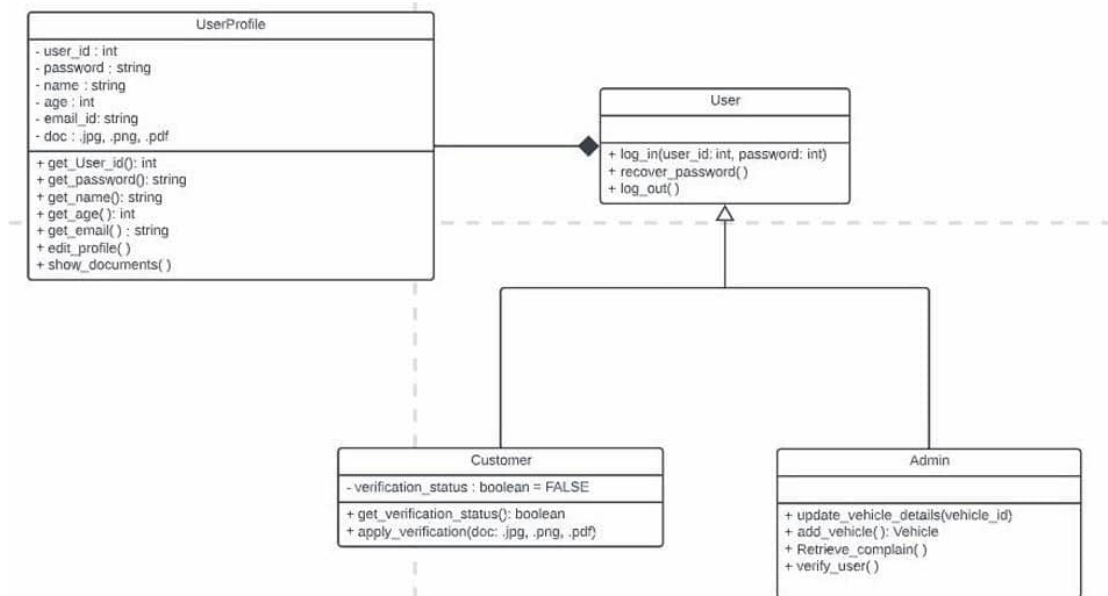
- Submarine

```
1 package week_8_UTS.question_2;
2
3 public class Submarine extends WaterVehicle {
4     private int maxDepth;
5
6     public Submarine(String make, String model, int year, float displacement, int maxDepth) {
7         super(make, model, year, displacement);
8         this.maxDepth = maxDepth;
9     }
10
11     public int getMaxDepth() {
12         return maxDepth;
13     }
14
15     public void setMaxDepth(int maxDepth) {
16         this.maxDepth = maxDepth;
17     }
18
19     public void submerge() {
20         System.out.println(
21             "Submerging the " + getMake() + " " + getModel() + " to a maximum depth of " + maxDepth + " meters");
22     }
23
24     public void surface() {
25         System.out.println("Surfacing the " + getMake() + " " + getModel());
26     }
27 }
```

- Main

```
1 package week_8_UTS.question_2;
2
3 public class MainDemoVehicle {
4     public static void main(String[] args) {
5         // Demonstrate Car
6         System.out.println("=== Car Demo ===");
7         Car car = new Car("Tesla", "Model S", 2023, 4, 4, true);
8         demonstrateCar(car);
9
10        // Demonstrate Motorcycle
11        System.out.println("\n=== Motorcycle Demo ===");
12        Motorcycle motorcycle = new Motorcycle("Harley Davidson", "Sportster", 2023, 2, false);
13        demonstrateMotorcycle(motorcycle);
14
15        // Demonstrate Boat
16        System.out.println("\n=== Boat Demo ===");
17        Boat boat = new Boat("Sea Ray", "Sundancer", 2023, 10000f, "Inboard");
18        demonstrateBoat(boat);
19
20        // Demonstrate Submarine
21        System.out.println("\n=== Submarine Demo ===");
22        Submarine submarine = new Submarine("General Dynamics", "Virginia class", 2023, 7800f, 800);
23        demonstrateSubmarine(submarine);
24    }
25
26    private static void demonstrateCar(Car car) {
27        System.out.println("Car: " + car.getMake() + " " + car.getModel() + " (" + car.getYear() + ")");
28        System.out.println("Number of doors: " + car.getNumDoors());
29        System.out.println("Is electric: " + car.isElectric());
30        car.startEngine();
31        car.drive();
32        car.accelerate();
33        car.brake();
34        car.stopEngine();
35    }
36
37    private static void demonstrateMotorcycle(Motorcycle motorcycle) {
38        System.out.println("Motorcycle: " + motorcycle.getMake() + " " + motorcycle.getModel() + " (" +
39            + motorcycle.getYear() + ")");
40        System.out.println("Has sidecar: " + motorcycle.hasSidecar());
41        motorcycle.startEngine();
42        motorcycle.drive();
43        motorcycle.wheelie();
44        motorcycle.stopEngine();
45    }
46
47    private static void demonstrateBoat(Boat boat) {
48        System.out.println("Boat: " + boat.getMake() + " " + boat.getModel() + " (" + boat.getYear() + ")");
49        System.out.println("Displacement: " + boat.getDisplacement() + " tons");
50        System.out.println("Propulsion type: " + boat.getPropulsionType());
51        boat.startEngine();
52        boat.sail();
53        boat.anchor();
54        boat.stopEngine();
55    }
56
57    private static void demonstrateSubmarine(Submarine submarine) {
58        System.out.println(
59            "Submarine: " + submarine.getMake() + " " + submarine.getModel() + " (" + submarine.getYear() + ")");
60        System.out.println("Displacement: " + submarine.getDisplacement() + " tons");
61        System.out.println("Maximum depth: " + submarine.getMaxDepth() + " meters");
62        submarine.startEngine();
63        submarine.sail();
64        submarine.submerge();
65        submarine.surface();
66        submarine.stopEngine();
67    }
68 }
```

3. Please identify the class diagram by providing an explanation of the concept of inheritance, the relationship between classes and the following system flow, create a program code from the following class diagram!



My Answer :

- UserProfile

```
1 package week_8_UTS.question_3;
2
3 public class UserProfile09 {
4     private int userId;
5     private String password;
6     private String name;
7     private int age;
8     private String emailId;
9     private String doc;
10
11     public UserProfile09(int userId, String password, String name, int age, String emailId, String doc) {
12         this.userId = userId;
13         this.password = password;
14         this.name = name;
15         this.age = age;
16         this.emailId = emailId;
17         this.doc = doc;
18     }
19
20     public int getUserId() {
21         return userId;
22     }
23
24     public String getPassword() {
25         return password;
26     }
27
28     public String getName() {
29         return name;
30     }
31
32     public int getAge() {
33         return age;
34     }
35
36     public String getEmail() {
37         return emailId;
38     }
39
40     public void editProfile(String newName, int newAge, String newEmail) {
41         this.name = newName;
42         this.age = newAge;
43         this.emailId = newEmail;
44         System.out.println("Profile updated successfully.");
45     }
46
47     public void showDocuments() {
48         System.out.println("Documents: " + doc);
49     }
50
51     public String getDoc() {
52         return doc;
53     }
54 }
```

- User

```
1 package week_8_UTS.question_3;
2
3 public class User09 {
4     public boolean login(int userId, String password) {
5         System.out.println("User logged in with ID: " + userId);
6         return true;
7     }
8
9     public void recoverPassword() {
10        System.out.println("Password recovery initiated.");
11    }
12
13    public void logout() {
14        System.out.println("User logged out.");
15    }
16 }
```

- Customer

```
1 package week_8_UTS.question_3;
2
3 public class Customer09 extends User09 {
4     private boolean verificationStatus = false;
5
6     public boolean getVerificationStatus() {
7         return verificationStatus;
8     }
9
10    public void applyVerification(String doc) {
11        if (doc != null && !doc.isEmpty()) {
12            verificationStatus = true;
13            System.out.println("Verification applied with document: " + doc);
14        } else {
15            System.out.println("Document not valid for verification.");
16        }
17    }
18 }
```

- Admin

```
1 package week_8_UTS.question_3;
2
3 public class Admin09 extends User09 {
4     public void updateVehicleDetails(int vehicleId) {
5         System.out.println("Vehicle details updated for Vehicle ID: " + vehicleId);
6     }
7
8     public void addVehicle() {
9         System.out.println("New vehicle added.");
10    }
11
12    public void retrieveComplain() {
13        System.out.println("Retrieving user complaints.");
14    }
15
16    public void verifyUser(Customer09 customer) {
17        if (customer.getVerificationStatus()) {
18            System.out.println("Customer verified.");
19        } else {
20            System.out.println("Customer not verified.");
21        }
22    }
23 }
```

- Main

```
1 package week_8_UTS.question_3;
2
3 public class Main09 {
4     public static void main(String[] args) {
5         UserProfile09 userProfile = new UserProfile09(101, "password101", "Emil", 30, "emil@gmail.com", "profile.jpg");
6         System.out.println("User Profile:");
7         System.out.println("-".repeat(30));
8         System.out.println("Name: " + userProfile.getName());
9         System.out.println("Age: " + userProfile.getAge());
10        System.out.println("Email: " + userProfile.getEmail());
11        System.out.println("Documents: " + userProfile.getDoc());
12        userProfile.showDocuments();
13        userProfile.editProfile("Edo", 31, "edo@gmail.com");
14
15        Customer09 customer = new Customer09();
16        System.out.println("\nLogin sebagai Customer:");
17        System.out.println("-".repeat(30));
18        customer.login(101, "password101");
19        customer.applyVerification("member_card.png");
20        System.out.println("Verification Status: " + customer.getVerificationStatus());
21        customer.logout();
22
23        Admin09 admin = new Admin09();
24        System.out.println("\nLogin sebagai Admin:");
25        System.out.println("-".repeat(30));
26        admin.login(000, "admin");
27        admin.addVehicle();
28        admin.updateVehicleDetails(010);
29        admin.retrieveComplain();
30        admin.verifyUser(customer);
31        admin.logout();
32    }
33 }
```

---- Good Luck ----