Question 1 — Inheritance

File: BaseEmployee.java

```
public class BaseEmployee extends Employee {
    private double baseSalary;

    public BaseEmployee(String firstName, String lastName, String ssn, double
baseSalary) {
        super(firstName, lastName, ssn);
        setBaseSalary(baseSalary);
    }

    public double getBaseSalary() {
        return baseSalary;
    }

    public void setBaseSalary(double baseSalary) {
        this.baseSalary = baseSalary;
    }
}
```

File: CommissionEmployee.java

```
public class CommissionEmployee extends Employee {
    private double commissionRate;
    private double grossSales;

    public CommissionEmployee(String firstName, String lastName, String ssn,
double commissionRate, double grossSales) {
        super(firstName, lastName, ssn);
        setCommissionRate(commissionRate);
        setGrossSales(grossSales);
    }

    public double getCommissionRate() {
        return commissionRate;
    }

    public void setCommissionRate(double commissionRate) {
        this.commissionRate = commissionRate;
    }

    public double getGrossSales() {
        return grossSales;
    }
```

```
    public void setGrossSales(double grossSales) {
        this.grossSales = grossSales;
    }
}
```

File: Driver.java

```
import java.util.ArrayList;
import java.util.List;

public class Driver {
    public static void main(String[] args) {
        List<Employee> employees = new ArrayList<>();

        // Salaried employees
        employees.add(new SalariedEmployee("Joe", "Jones", "111-11-1111",
2500.00));
        employees.add(new SalariedEmployee("Renwa", "Chanel", "555-55-5555",
1700.00));

        // Hourly employees
        employees.add(new HourlyEmployee("Stephanie", "Smith", "222-22-2222",
25.00, 32.0));
        employees.add(new HourlyEmployee("Mary", "Quinn", "333-33-3333",
19.00, 47.0));

        // Commission employees
        employees.add(new CommissionEmployee("Nicole", "Dior", "444-44-4444",
0.15, 50000.00));
        employees.add(new CommissionEmployee("Mahnaz", "Vaziri", "777-77-
7777", 0.22, 40000.00));

        // Base-salary-only employee
        employees.add(new BaseEmployee("Mike", "Davenport", "666-66-6666",
95000.00));

    }
}
```

File: Employee.java

```
public class Employee {
    private String firstName;
    private String lastName;
    private String socialSecurityNumber;
```

```
    public Employee(String firstName, String lastName, String
socialSecurityNumber) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.socialSecurityNumber = socialSecurityNumber;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getSocialSecurityNumber() {
        return socialSecurityNumber;
    }

    public void setSocialSecurityNumber(String socialSecurityNumber) {
        this.socialSecurityNumber = socialSecurityNumber;
    }

}
```

File: HourlyEmployee.java

```
public class HourlyEmployee extends Employee {
    private double hourlyWage;
    private double hoursWorked;

    public HourlyEmployee(String firstName, String lastName, String ssn,
double hourlyWage, double hoursWorked) {
        super(firstName, lastName, ssn);
        setHourlyWage(hourlyWage);
        setHoursWorked(hoursWorked);
    }
```

```
    public double getHourlyWage() {
        return hourlyWage;
    }

    public void setHourlyWage(double hourlyWage) {
        this.hourlyWage = hourlyWage;
    }

    public double getHoursWorked() {
        return hoursWorked;
    }

    public void setHoursWorked(double hoursWorked) {
        this.hoursWorked = hoursWorked;
    }
}
```

File: SalariedEmployee.java

```
public class SalariedEmployee extends Employee {
    private double weeklySalary;

    public SalariedEmployee(String firstName, String lastName, String ssn,
double weeklySalary) {
        super(firstName, lastName, ssn);
        setWeeklySalary(weeklySalary);
    }

    public double getWeeklySalary() {
        return weeklySalary;
    }

    public void setWeeklySalary(double weeklySalary) {
        this.weeklySalary = weeklySalary;
    }

}
```

Output Screenshots:

No output

Question 2 — Interface

File: Freelancer.java

```
public class Freelancer implements Payable {
    private String firstName;
    private String lastName;
    private double hourlyRate;
    private double hoursWorked;

    public Freelancer(String firstName, String lastName, double hourlyRate,
double hoursWorked) {
        this.firstName = firstName;
        this.lastName = lastName;
        setHourlyRate(hourlyRate);
        setHoursWorked(hoursWorked);
    }

    public String getFirstName() { return firstName; }
    public void setFirstName(String firstName) { this.firstName = firstName; }

    public String getLastName() { return lastName; }
    public void setLastName(String lastName) { this.lastName = lastName; }

    public double getHourlyRate() { return hourlyRate; }
    public void setHourlyRate(double hourlyRate) {
        if (hourlyRate < 0) throw new IllegalArgumentException("Hourly rate
must be non-negative");
        this.hourlyRate = hourlyRate;
    }

    public double getHoursWorked() { return hoursWorked; }
    public void setHoursWorked(double hoursWorked) {
        if (hoursWorked < 0) throw new IllegalArgumentException("Hours worked
must be non-negative");
        this.hoursWorked = hoursWorked;
    }

    @Override
    public double calculatePayment() {
        double regularHours = Math.min(40.0, hoursWorked);
        double overtimeHours = Math.max(0.0, hoursWorked - 40.0);
        return regularHours * hourlyRate + overtimeHours * hourlyRate * 1.5;
    }

    @Override
    public String getPayeeName() {
        return firstName + " " + lastName;
    }
```

```
    public void print() {
        System.out.printf("Freelancer: %s, Payment: $%.2f%n", getPayeeName(),
calculatePayment());
    }
}
```

File: Payable.java

```
public interface Payable {
    double calculatePayment();
    String getPayeeName();
}
```

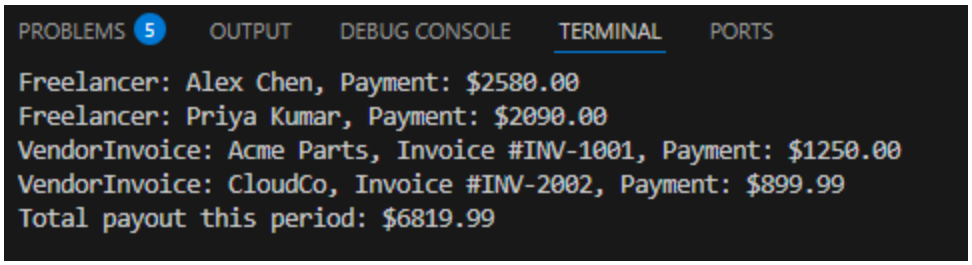
File: PayableDriver.java

```
import java.util.ArrayList;
import java.util.List;

public class PayableDriver {
    public static void main(String[] args) {
        List<Payable> payables = new ArrayList<>();

        payables.add(new Freelancer("Alex", "Chen", 60.0, 42.0));
        payables.add(new Freelancer("Priya", "Kumar", 55.0, 38.0));

        payables.add(new VendorInvoice("Acme Parts", "INV-1001", 1250.00));
        payables.add(new VendorInvoice("CloudCo", "INV-2002", 899.99));

        double totalPayout = 0.0;
        for (Payable p : payables) {
            if (p instanceof Freelancer) {
                ((Freelancer) p).print();
            } else if (p instanceof VendorInvoice) {
                ((VendorInvoice) p).print();
            }
            totalPayout += p.calculatePayment();
        }

        System.out.printf("Total payout this period: $%.2f%n", totalPayout);
    }
}
```

File: VendorInvoice.java

```
public class VendorInvoice implements Payable {
    private String vendorName;
    private String invoiceNumber;
    private double amountDue;

    public VendorInvoice(String vendorName, String invoiceNumber, double
amountDue) {
        this.vendorName = vendorName;
        this.invoiceNumber = invoiceNumber;
        setAmountDue(amountDue);
    }

    public String getVendorName() { return vendorName; }
    public void setVendorName(String vendorName) { this.vendorName =
vendorName; }

    public String getInvoiceNumber() { return invoiceNumber; }
    public void setInvoiceNumber(String invoiceNumber) { this.invoiceNumber =
invoiceNumber; }

    public double getAmountDue() { return amountDue; }
    public void setAmountDue(double amountDue) {
        if (amountDue < 0) throw new IllegalArgumentException("Amount due must
be non-negative");
        this.amountDue = amountDue;
    }

    @Override
    public double calculatePayment() {
        return amountDue;
    }

    @Override
    public String getPayeeName() {
        return vendorName;
    }

    public void print() {
        System.out.printf("VendorInvoice: %s, Invoice #%s, Payment: $%.2f%n",
                vendorName, invoiceNumber, calculatePayment());
    }
}
```

Output Screenshots:



Question 3 — Polymorphism

File: CargoShip.java

```
public class CargoShip extends Ship {
    private int cargoCapacityTonnage;

    public CargoShip(String name, String yearBuilt, int cargoCapacityTonnage)
{
        super(name, yearBuilt);
        setCargoCapacityTonnage(cargoCapacityTonnage);
    }

    public int getCargoCapacityTonnage() { return cargoCapacityTonnage; }
    public void setCargoCapacityTonnage(int cargoCapacityTonnage) {
this.cargoCapacityTonnage = cargoCapacityTonnage; }

    @Override
    public void print() {
        System.out.printf("CargoShip: %s, Capacity: %d tons%n", getName(),
cargoCapacityTonnage);
    }
}
```

File: CruiseShip.java

```
public class CruiseShip extends Ship {
    private int maxPassengers;

    public CruiseShip(String name, String yearBuilt, int maxPassengers) {
        super(name, yearBuilt);
        setMaxPassengers(maxPassengers);
    }

    public int getMaxPassengers() { return maxPassengers; }
    public void setMaxPassengers(int maxPassengers) { this.maxPassengers =
maxPassengers; }
```

```
    @Override
    public void print() {
        System.out.printf("CruiseShip: %s, Max Passengers: %d%n", getName(),
maxPassengers);
    }
}
```

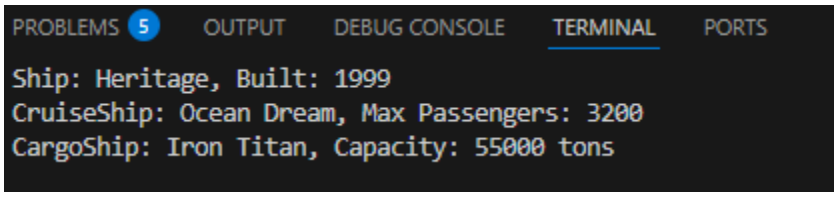
File: Ship.java

```
public class Ship {
    private String name;
    private String yearBuilt;

    public Ship(String name, String yearBuilt) {
        this.name = name;
        this.yearBuilt = yearBuilt;
    }

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

    public String getYearBuilt() { return yearBuilt; }
    public void setYearBuilt(String yearBuilt) { this.yearBuilt = yearBuilt; }

    public void print() {
        System.out.printf("Ship: %s, Built: %s%n", name, yearBuilt);
    }
}
```

File: ShipDemo.java

```
public class ShipDemo {
    public static void main(String[] args) {
        Ship[] fleet = new Ship[3];
        fleet[0] = new Ship("Heritage", "1999");
        fleet[1] = new CruiseShip("Ocean Dream", "2012", 3200);
        fleet[2] = new CargoShip("Iron Titan", "2008", 55000);

        for (Ship s : fleet) {
            s.print();
        }
    }
}
```

Output Screenshots:



Question 4 — Aggregation

File: Course.java

```
import java.util.ArrayList;
import java.util.List;

public class Course {
    private String courseName;
    private final List<Instructor> instructors = new ArrayList<>();
    private final List<Textbook> textbooks = new ArrayList<>();

    public Course(String courseName) {
        this.courseName = courseName;
    }

    public String getCourseName() { return courseName; }
    public void setCourseName(String courseName) { this.courseName =
courseName; }

    public void addInstructor(Instructor instructor) {
instructors.add(instructor); }
    public void addTextbook(Textbook textbook) { textbooks.add(textbook); }

    public void print() {
        System.out.println("Course: " + courseName);
        if (!instructors.isEmpty()) {
            System.out.print("Instructors: ");
            for (int i = 0; i < instructors.size(); i++) {
                Instructor inst = instructors.get(i);
                System.out.print(inst.getFirstName() + " " +
inst.getLastName());
                if (i < instructors.size() - 1) System.out.print(", ");
            }
            System.out.println();
        }
        if (!textbooks.isEmpty()) {
            System.out.print("Textbooks: ");
            for (int i = 0; i < textbooks.size(); i++) {
```

```
                Textbook t = textbooks.get(i);
                System.out.print(t.getTitle() + " by " + t.getAuthor());
                if (i < textbooks.size() - 1) System.out.print(", ");
            }
            System.out.println();
        }
    }
}
```

File: CourseDemo.java

```
public class CourseDemo {
    public static void main(String[] args) {
        Course course = new Course("CS 5800 - Advanced OOP");
        Instructor nima = new Instructor("Nima", "Davarpanah", "3-2636");
        Textbook cleanCode = new Textbook("Clean Code", "Robert C. Martin",
"Prentice Hall");
        course.addInstructor(nima);
        course.addTextbook(cleanCode);
        course.print();

        System.out.println();

        Instructor alex = new Instructor("Alex", "Nguyen", "4-1201");
        Textbook refactoring = new Textbook("Refactoring", "Martin Fowler",
"Addison-Wesley");
        course.addInstructor(alex);
        course.addTextbook(refactoring);
        course.print();
    }
}
```

File: Instructor.java

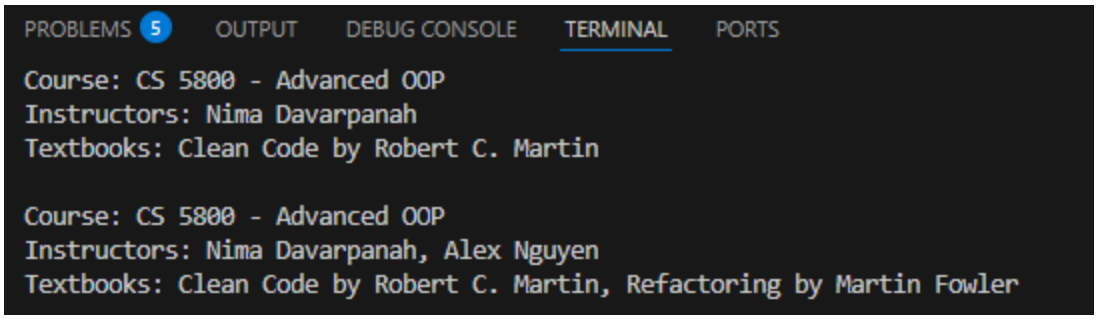
```
public class Instructor {
    private String firstName;
    private String lastName;
    private String officeNumber;

    public Instructor(String firstName, String lastName, String officeNumber)
{
        this.firstName = firstName;
        this.lastName = lastName;
        this.officeNumber = officeNumber;
    }
```

```
    public String getFirstName() { return firstName; }
    public void setFirstName(String firstName) { this.firstName = firstName; }

    public String getLastName() { return lastName; }
    public void setLastName(String lastName) { this.lastName = lastName; }

    public String getOfficeNumber() { return officeNumber; }
    public void setOfficeNumber(String officeNumber) { this.officeNumber =
officeNumber; }

}
```

File: Textbook.java

```
public class Textbook {
    private String title;
    private String author;
    private String publisher;

    public Textbook(String title, String author, String publisher) {
        this.title = title;
        this.author = author;
        this.publisher = publisher;
    }

    public String getTitle() { return title; }
    public void setTitle(String title) { this.title = title; }

    public String getAuthor() { return author; }
    public void setAuthor(String author) { this.author = author; }

    public String getPublisher() { return publisher; }
    public void setPublisher(String publisher) { this.publisher = publisher; }

}
```

Output Screenshots:



Question 5 — Composition

File: Folder.java

```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

public class Folder {
    private String name;
    private final List<Folder> subFolders = new ArrayList<>();
    private final List<FSFile> files = new ArrayList<>();

    public Folder(String name) {
        this.name = name;
    }

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

    public Folder addSubFolder(String name) {
        Folder f = new Folder(name);
        subFolders.add(f);
        return f;
    }

    public void addFile(FSFile file) { files.add(file); }

    public Folder findSubFolder(String name) {
        for (Folder f : subFolders) {
            if (f.getName().equals(name)) return f;
        }
        return null;
    }

    public boolean deleteSubFolder(String name) {
        Iterator<Folder> it = subFolders.iterator();
```

```
        while (it.hasNext()) {
            Folder f = it.next();
            if (f.getName().equals(name)) {
                it.remove();
                return true;
            }
        }
        return false;
    }

    public void print() {
        print("");
    }

    private void print(String indent) {
        System.out.println(indent + name + "/");
        String childIndent = indent + "  ";
        for (Folder f : subFolders) {
            f.print(childIndent);
        }
        for (FSFile file : files) {
            file.print(childIndent);
        }
    }
}
```

File: FSDriver.java

```
public class FSDriver {
    public static void main(String[] args) {
        Folder root = new Folder("php_demo1");
        Folder source = root.addSubFolder("Source Files");

        Folder app = source.addSubFolder("app");
        app.addFile(new FSFile("app.php"));

        Folder publicFolder = source.addSubFolder("public");
        publicFolder.addFile(new FSFile("index.php"));

        Folder config = source.addSubFolder("config");
        config.addFile(new FSFile("config.php"));

        root.print();
        System.out.println();

        source.deleteSubFolder("app");
        root.print();
```

```
        System.out.println();

        source.deleteSubFolder("public");
        root.print();
    }
}
```

File: FSFile.java

```
public class FSFile {
    private String name;

    public FSFile(String name) {
        this.name = name;
    }

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

    public void print(String indent) {
        System.out.println(indent + "- " + name);
    }
}
```

Output Screenshots:


PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
demo1/
Include Path/
Remote Files/
Source Files/
.phalcon/
app/
config/
controllers/
library/
migrations/
models/
views/
cache/
public/
- .htaccess
- .htrouter.php
- index.html
demo1/
Include Path/
Remote Files/
Source Files/
.phalcon/
cache/
public/
- .htaccess
- .htrouter.php
- index.html
demo1/
Include Path/
Remote Files/
Source Files/
.phalcon/
cache/
