**Aishwarya TJ**

**AN2**

1. Code

class BankAccount {

private String accountNumber;

private String accountHolderName;

private double balance;

private static int totalAccounts = 0;

public BankAccount(String accountHolderName, double initialDeposit) {

this.accountHolderName = accountHolderName;

this.balance = initialDeposit;

this.accountNumber = generateAccountNumber();

totalAccounts++;

}

private static String generateAccountNumber() {

return "ACC" + String.format("%03d", totalAccounts + 1);

}

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

System.out.println(accountHolderName + " deposited " + amount + ". New balance: " + balance);

} else {

System.out.println("Deposit amount must be positive.");

}

}

public void withdraw(double amount) {

if (amount > 0) {

if (amount <= balance) {

balance -= amount;

System.out.println(accountHolderName + " withdrew " + amount + ". New balance: " + balance);

} else {

System.out.println("Insufficient balance for withdrawal.");

}

} else {

System.out.println("Withdrawal amount must be positive.");

}

}

public void checkBalance() {

System.out.println("Balance for " + accountHolderName + " (" + accountNumber + "): " + balance);

}

public static int getTotalAccounts() {

return totalAccounts;

}

public void displayAccountInfo() {

System.out.println("Account Number: " + accountNumber);

System.out.println("Account Holder: " + accountHolderName);

System.out.println("Balance: " + balance);

System.out.println("--------------------------");

}

}

public class BankSystem {

public static void main(String[] args) {

BankAccount[] accounts = new BankAccount[3];

accounts[0] = new BankAccount("Alice", 1000);

accounts[1] = new BankAccount("Bob", 1500);

accounts[2] = new BankAccount("Charlie", 2000);

accounts[0].deposit(500);

accounts[1].withdraw(300);

accounts[2].checkBalance();

for (int i = 0; i < accounts.length; i++) {

accounts[i].displayAccountInfo();

}

System.out.println("Total Accounts Created: " + BankAccount.getTotalAccounts());

}

}

1. Code

class Book {

private String bookId;

private String title;

private String author;

private boolean isAvailable;

private static int totalBooks = 0;

private static int availableBooks = 0;

public Book(String title, String author) {

this.title = title;

this.author = author;

this.isAvailable = true;

this.bookId = generateBookId();

totalBooks++;

availableBooks++;

}

private static String generateBookId() {

return "B" + String.format("%03d", totalBooks + 1);

}

public String getBookId() {

return bookId;

}

public boolean isAvailable() {

return isAvailable;

}

public void issueBook() {

if (isAvailable) {

isAvailable = false;

availableBooks--;

System.out.println(title + " has been issued.");

} else {

System.out.println(title + " is not available.");

}

}

public void returnBook() {

if (!isAvailable) {

isAvailable = true;

availableBooks++;

System.out.println(title + " has been returned.");

}

}

public void displayBookInfo() {

System.out.println("ID: " + bookId + ", Title: " + title + ", Author: " + author + ", Available: " + isAvailable);

}

public static int getTotalBooks() {

return totalBooks;

}

public static int getAvailableBooks() {

return availableBooks;

}

}

class Member {

private String memberId;

private String memberName;

private String[] booksIssued;

private int bookCount;

private static int totalMembers = 0;

public Member(String memberName, int maxBooks) {

this.memberName = memberName;

this.booksIssued = new String[maxBooks];

this.bookCount = 0;

this.memberId = generateMemberId();

totalMembers++;

}

private static String generateMemberId() {

return "M" + String.format("%03d", totalMembers + 1);

}

public void borrowBook(Book book) {

if (book.isAvailable() && bookCount < booksIssued.length) {

book.issueBook();

booksIssued[bookCount] = book.getBookId();

bookCount++;

System.out.println(memberName + " borrowed " + book.getBookId());

} else {

System.out.println(memberName + " cannot borrow this book.");

}

}

public void returnBook(String bookId, Book[] books) {

for (int i = 0; i < bookCount; i++) {

if (booksIssued[i].equals(bookId)) {

for (Book b : books) {

if (b.getBookId().equals(bookId)) {

b.returnBook();

System.out.println(memberName + " returned " + bookId);

break;

}

}

for (int j = i; j < bookCount - 1; j++) {

booksIssued[j] = booksIssued[j + 1];

}

booksIssued[bookCount - 1] = null;

bookCount--;

return;

}

}

System.out.println(memberName + " does not have this book.");

}

public void displayMemberInfo() {

System.out.print("ID: " + memberId + ", Name: " + memberName + ", Books Issued: ");

for (int i = 0; i < bookCount; i++) {

System.out.print(booksIssued[i] + " ");

}

System.out.println();

}

}

public class LibrarySystem {

public static void main(String[] args) {

Book[] books = new Book[3];

books[0] = new Book("The Hobbit", "J.R.R. Tolkien");

books[1] = new Book("1984", "George Orwell");

books[2] = new Book("Clean Code", "Robert C. Martin");

Member[] members = new Member[2];

members[0] = new Member("Alice", 2);

members[1] = new Member("Bob", 2);

members[0].borrowBook(books[0]);

members[0].borrowBook(books[1]);

members[1].borrowBook(books[1]);

members[0].displayMemberInfo();

members[1].displayMemberInfo();

books[0].displayBookInfo();

books[1].displayBookInfo();

books[2].displayBookInfo();

members[0].returnBook("B001", books);

members[1].borrowBook(books[0]);

members[0].displayMemberInfo();

members[1].displayMemberInfo();

System.out.println("Total Books: " + Book.getTotalBooks());

System.out.println("Available Books: " + Book.getAvailableBooks());

}

}

1. Code

class Employee {

private String empId;

private String empName;

private String department;

private double baseSalary;

private String empType;

private static int totalEmployees = 0;

public Employee(String empName, String department, double baseSalary) {

this.empId = generateEmpId();

this.empName = empName;

this.department = department;

this.baseSalary = baseSalary;

this.empType = "Full-Time";

totalEmployees++;

}

public Employee(String empName, String department, double hourlyRate, int hours) {

this.empId = generateEmpId();

this.empName = empName;

this.department = department;

this.baseSalary = hourlyRate \* hours;

this.empType = "Part-Time";

totalEmployees++;

}

public Employee(String empName, String department, double contractAmount, boolean isContract) {

this.empId = generateEmpId();

this.empName = empName;

this.department = department;

this.baseSalary = contractAmount;

this.empType = "Contract";

totalEmployees++;

}

private static String generateEmpId() {

return "E" + String.format("%03d", totalEmployees + 1);

}

public double calculateSalary(double bonus) {

if (empType.equals("Full-Time")) {

return baseSalary + bonus;

}

return baseSalary;

}

public double calculateSalary(int hours, double hourlyRate) {

if (empType.equals("Part-Time")) {

return hours \* hourlyRate;

}

return baseSalary;

}

public double calculateSalary() {

return baseSalary;

}

public double calculateTax(double salary) {

if (empType.equals("Full-Time")) {

return salary \* 0.2;

} else if (empType.equals("Part-Time")) {

return salary \* 0.1;

} else {

return salary \* 0.15;

}

}

public void generatePaySlip(double salary, double tax) {

System.out.println("----- Pay Slip -----");

System.out.println("Employee ID: " + empId);

System.out.println("Name: " + empName);

System.out.println("Department: " + department);

System.out.println("Type: " + empType);

System.out.println("Salary: " + salary);

System.out.println("Tax Deducted: " + tax);

System.out.println("Net Salary: " + (salary - tax));

System.out.println("---------------------");

}

public void displayEmployeeInfo() {

System.out.println("ID: " + empId + ", Name: " + empName + ", Dept: " + department + ", Type: " + empType);

}

public static int getTotalEmployees() {

return totalEmployees;

}

}

public class PayrollSystem {

public static void main(String[] args) {

Employee e1 = new Employee("Alice", "HR", 40000);

Employee e2 = new Employee("Bob", "IT", 500, 40);

Employee e3 = new Employee("Charlie", "Finance", 30000, true);

double salary1 = e1.calculateSalary(5000);

double tax1 = e1.calculateTax(salary1);

e1.generatePaySlip(salary1, tax1);

double salary2 = e2.calculateSalary(40, 500);

double tax2 = e2.calculateTax(salary2);

e2.generatePaySlip(salary2, tax2);

double salary3 = e3.calculateSalary();

double tax3 = e3.calculateTax(salary3);

e3.generatePaySlip(salary3, tax3);

System.out.println("Total Employees: " + Employee.getTotalEmployees());

}

}

1. Code

class Vehicle {

private String vehicleId;

private String brand;

private String model;

private double rentPerDay;

private boolean isAvailable;

private static int totalVehicles = 0;

private static double totalRevenue = 0;

private static String companyName;

private static int rentalDays = 0;

public Vehicle(String vehicleId, String brand, String model, double rentPerDay) {

this.vehicleId = vehicleId;

this.brand = brand;

this.model = model;

this.rentPerDay = rentPerDay;

this.isAvailable = true;

totalVehicles++;

}

public double calculateRent(int days) {

double rent = rentPerDay \* days;

totalRevenue += rent;

rentalDays += days;

return rent;

}

public void rentVehicle(int days) {

if (isAvailable) {

double rent = calculateRent(days);

isAvailable = false;

System.out.println(brand + " " + model + " rented for " + days + " days. Rent: " + rent);

} else {

System.out.println(brand + " " + model + " is not available.");

}

}

public void returnVehicle() {

if (!isAvailable) {

isAvailable = true;

System.out.println(brand + " " + model + " has been returned.");

}

}

public void displayVehicleInfo() {

System.out.println("ID: " + vehicleId + ", Brand: " + brand + ", Model: " + model + ", Rent/Day: " + rentPerDay + ", Available: " + isAvailable);

}

public static void setCompanyName(String name) {

companyName = name;

}

public static double getTotalRevenue() {

return totalRevenue;

}

public static double getAverageRentPerDay() {

if (rentalDays == 0) return 0;

return totalRevenue / rentalDays;

}

public static void displayCompanyStats() {

System.out.println("Company: " + companyName);

System.out.println("Total Vehicles: " + totalVehicles);

System.out.println("Total Revenue: " + totalRevenue);

System.out.println("Average Rent Per Day: " + getAverageRentPerDay());

}

}

public class VehicleRentalSystem {

public static void main(String[] args) {

Vehicle.setCompanyName("SuperDrive Rentals");

Vehicle v1 = new Vehicle("V001", "Toyota", "Camry", 1000);

Vehicle v2 = new Vehicle("V002", "Honda", "Civic", 800);

Vehicle v3 = new Vehicle("V003", "Yamaha", "R15", 500);

v1.rentVehicle(5);

v2.rentVehicle(3);

v1.returnVehicle();

v3.rentVehicle(2);

v1.displayVehicleInfo();

v2.displayVehicleInfo();

v3.displayVehicleInfo();

Vehicle.displayCompanyStats();

}

}