

WEEK 9

1. Create a class Vehicle, write a method cost() in this class. Create two classes Bus and Train which have their own display() methods and inherit from Vehicle class. Create objects of Bus and Train class and call cost() and display() methods.

The image shows a Java code editor with four tabs: Main.java, Bus.java, Train.java, and Vehicle.java. The Main.java tab is active, displaying the following code:

```
1 package week9_1;
2
3 public class Main {
4     public static void main(String[] args) {
5         Bus myBus = new Bus();
6         myBus.display();
7         myBus.cost();
8
9         System.out.println();
10
11        Train myTrain = new Train();
12        myTrain.display();
13        myTrain.cost();
14    }
15 }
```

The Bus.java tab is selected, showing:

```
1 package week9_1;
2
3 class Bus extends Vehicle { 2 usages
4     public void display() {
5         System.out.println("This is a Bus.");
6     }
7 }
```

The Train.java tab is selected, showing:

```
1 package week9_1;
2
3 class Train extends Vehicle { 2 usages
4     public void display() {
5         System.out.println("This is a Train.");
6     }
7 }
```

The Vehicle.java tab is selected, showing:

```
1 package week9_1;
2
3 class Vehicle { 2 usages 2 inheritors
4     public void cost() { 2 usages
5         System.out.println("The cost of the vehicle is...");
6     }
7 }
```

At the bottom, there are two output boxes:

This is a Bus.
The cost of the vehicle is...

This is a Train.
The cost of the vehicle is...

WEEK 9

2. Create class University which has data members- name and ranking. Create class Faculty that extends University class has data member- name and method- Details(). Create a new class Department which is derived from Faculty and has data member- name, chairman and method- Details() and Display() where Display() method calls Details() methods of both Faculty and Department class in its body. Create an object of Department class to Display() method and University ranking.

The screenshot shows a Java code editor with two tabs visible: "University.java" (selected) and "Faculty.java".

University.java:

```
1 package week9_2;
2
3 @class University { 1 usage 2 inheritors
4     String name; 2 usages
5     int ranking; 2 usages
6
7     public University(String name, int ranking) { 1 usage
8         this.name = name;
9         this.ranking = ranking;
10    }
11 }
12
```

Faculty.java:

```
1 package week9_2;
2
3 @class Faculty extends University { 1 usage 1 inheritor
4     String facultyName; 2 usages
5
6     public Faculty(String uniName, int ranking, String facultyName) { 1 usage
7         super(uniName, ranking);
8         this.facultyName = facultyName;
9     }
10
11 @public void Details() { 2 usages 1 override
12     System.out.println("Faculty: " + this.facultyName);
13 }
14 }
```

WEEK 9

```
>Main.java  x  Department.java  x  Faculty.java  University.java
1 package week9_2;
2
3 class Department extends Faculty { 2 usages
4     String deptName; 2 usages
5     String chairman; 2 usages
6
7     public Department(String uniName, int ranking, String facultyName, String deptName, String chairman) {
8         super(uniName, ranking, facultyName);
9         this.deptName = deptName;
10        this.chairman = chairman;
11    }
12    @†
13    public void Details() { 2 usages
14        System.out.println("Department: " + this.deptName);
15        System.out.println("Chairman: " + this.chairman);
16    }
17    public void Display() { 1 usage
18        System.out.println("--- Complete Details ---");
19        System.out.println("University: " + this.name);
20        super.Details();
21        this.Details();
22    }
23 }
```

```
>Main.java  x  Department.java  Faculty.java  University.java
1 package week9_2;
2
3 public class Main {
4     public static void main(String[] args) {
5         Department csDept = new Department( uniName: "AMU", ranking: 8,
6                                            facultyName: "F/O Science", deptName: "CS Dept.", chairman: "Mr. A.R. Faridi");
7
8         csDept.Display();
9
10        System.out.println("-----");
11        System.out.println("University Ranking: " + csDept.ranking);
12    }
13 }
```

```
--- Complete Details ---
University: AMU
Faculty: F/O Science
Department: CS Dept.
Chairman: Mr. A.R. Faridi
-----
University Ranking: 8
```

- 3.** Create class Account (Data members- Id, Account_holder_name, Address; Methodsdeposit(), withdraw()). Create two static methods in Account calculateSimpleInterest() and calculateCompoundInterest() and implement them.

WEEK 9

The screenshot shows a Java code editor with two tabs: Main.java and Account.java. The Account.java tab is active, displaying the following code:

```
1 package week9_3;
2
3 class Account { 4 usages
4     int id; 2 usages
5     String Account_holder_name; 2 usages
6     String Address; 2 usages
7     private double balance; 7 usages
8
9     public Account(int id, String name, String address, double initialBalance) { 1 usage
10         this.id = id;
11         this.Account_holder_name = name;
12         this.Address = address;
13         this.balance = initialBalance;
14     }
15
16     public void deposit(double amount) { 1 usage
17         if (amount > 0) {
18             this.balance += amount;
19             System.out.printf("Successfully deposited %.2f. New balance is %.2f\n", amount, this.balance);
20         } else {
21             System.out.println("Deposit amount must be positive.");
22         }
23     }
24
25     public void withdraw(double amount) {
26         if (amount <= 0) {
27             System.out.println("Withdrawal amount must be positive.");
28         } else if (this.balance >= amount) {
29             this.balance -= amount;
30             System.out.printf("Successfully withdrew %.2f. New balance is %.2f\n", amount, this.balance);
31         } else {
32             System.out.println("Withdrawal failed. Insufficient funds.");
33         }
34     }
35
36     public void displayDetails() { 2 usages
37         System.out.println("--- Account " + this.id + " ---");
38         System.out.println("Holder: " + this.Account_holder_name);
39         System.out.println("Address: " + this.Address);
40         System.out.printf("Current Balance: %.2f\n", this.balance);
41         System.out.println("-----");
42     }
43
44     public static double calculateSimpleInterest(double p, double r, double t) { 1 usage
45         return (p * r * t) / 100.0;
46     }
47
48     public static double calculateCompoundInterest(double p, double r, double t ) { 1 usage
49         double rateAsDecimal = r / 100.0;
50         double amount = p * Math.pow(1 + rateAsDecimal, t);
51         return amount - p;
52     }
53 }
54 }
```

WEEK 9

```
Main.java x Account.java
1 package week9_3;
2
3 public class Main {
4     public static void main(String[] args) {
5
6         Account account1 = new Account( id: 101, name: "Amit Sharma", address: "Agra, UP", initialBalance: 25000.0);
7
8         System.out.println("Using non-static methods:");
9         account1.displayDetails();
10        account1.deposit( amount: 10000.0);
11        account1.withdraw( amount: 5000.0);
12        account1.displayDetails();
13
14        System.out.println();
15
16        System.out.println("Using static methods:");
17        double p = 50000;
18        double r = 7.5;
19        double t = 5;
20
21        double si = Account.calculateSimpleInterest(p, r, t);
22        System.out.printf("Simple Interest for %.2f at %.2f% for %.1f years is: %.2f\n", p, r, t, si);
23
24        double ci = Account.calculateCompoundInterest(p, r, t);
25        System.out.printf("Compound Interest for %.2f at %.2f% for %.1f years (compounded annually) is: %.2f\n"
26    }
27 }
```

```
Using non-static methods:
--- Account 101 ---
Holder: Amit Sharma
Address: Agra, UP
Current Balance: 25000.00
-----
Successfully deposited 10000.00. New balance is 35000.00
Successfully withdrew 5000.00. New balance is 30000.00
--- Account 101 ---
Holder: Amit Sharma
Address: Agra, UP
Current Balance: 30000.00
-----

Using static methods:
Simple Interest for 50000.00 at 7.50% for 5.0 years is: 18750.00
Compound Interest for 50000.00 at 7.50% for 5.0 years (compounded annually) is: 21781.47
```

4. Create class Account (Data members- Id, Account_holder_name, Address; Methodsdeposit(), withdraw()). Declare deposit() and withdraw() as abstract methods. Declare Account class as abstract. (Create constructor in Account as well).

5. Create two children of Account- Saving (Data Members- Min_balance; Methodsdisplay(), deposit(), withdraw()) and

WEEK 9

Current (Data Members- Max_withdrawl_limit; Methods- display(),deposit(), withdraw()) . Create constructors for both classes. Implementation of deposit() and withdraw() should be specific to Saving and Current class. Create objects of Saving and Current class and display them

The screenshot shows a Java code editor with two tabs open: Account.java and Current.java. The Current.java tab is active.

```
>Main.java    Account.java ×  Current.java    Saving.java
1 package week9_4;
2
3 abstract class Account { 2 usages 2 inheritors
4     int id; 7 usages
5     String accountHolderName; 3 usages
6     String address; 3 usages
7
8     public Account(int id, String name, String address) { 2 usages
9         this.id = id;
10        this.accountHolderName = name;
11        this.address = address;
12    }
13
14     public abstract void deposit(double amount); 2 usages 2 implementations
15     public abstract void withdraw(double amount); 2 usages
16 }
17
18
>Main.java    Account.java ×  Current.java    Saving.java
1 package week9_4;
2
3 class Current extends Account { 2 usages
4     // Data member specific to Current account
5     double maxWithdrawLimit; 2 usages
6
7     /**
8      * Constructor for the Current class.
9      */
10    public Current(int id, String name, String address, double maxWithdrawal) { 1 usage
11        super(id, name, address); // Initialize parent class members
12        this.maxWithdrawLimit = maxWithdrawal;
13    }
14
15    @Override 2 usages
16    public void deposit(double amount) {
17        System.out.printf("Depositing %.2f into Current Account #%-d\n", amount, this.id);
18    }
19
20    @Override
21    public void withdraw(double amount) {
22        System.out.printf("Withdrawing %.2f from Current Account #%-d\n", amount, this.id);
23    }
24
25    public void display() {
26        System.out.println("--- Current Account Details ---");
27        System.out.println("Account ID: " + this.id);
28        System.out.println("Holder Name: " + this.accountHolderName);
29        System.out.println("Address: " + this.address);
30        System.out.println("Max Withdrawal Limit: " + this.maxWithdrawLimit);
31        System.out.println("-----");
32    }
33
34 }
```

WEEK 9

The screenshot shows a Java code editor with two tabs open: `Saving.java` and `Main.java`. The `Saving.java` tab is active, displaying the following code:

```
1 package week9_4;
2
3 class Saving extends Account { 2 usages
4     // Data member specific to Saving account
5     double minBalance; 2 usages
6
7     public Saving(int id, String name, String address, double minBalance) { 1 usage
8         super(id, name, address); // Initialize parent class members
9         this.minBalance = minBalance;
10    }
11
12    @Override 2 usages
13    public void deposit(double amount) {
14        System.out.printf("Depositing %.2f into Saving Account #%-d\n", amount, this.id);
15    }
16
17    @Override
18    public void withdraw(double amount) {
19        System.out.printf("Withdrawing %.2f from Saving Account #%-d\n", amount, this.id);
20    }
21
22    public void display() {
23        System.out.println("--- Saving Account Details ---");
24        System.out.println("Account ID: " + this.id);
25        System.out.println("Holder Name: " + this.accountHolderName);
26        System.out.println("Address: " + this.address);
27        System.out.println("Minimum Balance Rule: " + this.minBalance);
28        System.out.println("-----");
29    }
30}
31
```

The `Main.java` tab is also visible, showing the following code:

```
1 package week9_4;
2
3 public class Main {
4     public static void main(String[] args) {
5
6         Saving savingAccount = new Saving( id: 101, name: "Shaheen", address: "AMU", minBalance: 1000.0 );
7
8         Current currentAccount = new Current( id: 202, name: "Shafin", address: "AMU", maxWithdrawal: 100000.0 );
9
10        savingAccount.display();
11        savingAccount.deposit( amount: 15000 );
12        savingAccount.withdraw( amount: 4500 );
13
14        System.out.println();
15
16        currentAccount.display();
17        currentAccount.deposit( amount: 50000 );
18        currentAccount.withdraw( amount: 25000 );
19    }
20}
```

WEEK 9

```
--- Saving Account Details ---
Account ID: 101
Holder Name: Shaheer
Address: AMU
Minimum Balance Rule: 1000.0
-----
Depositing 15000.00 into Saving Account #101
Withdrawning 4500.00 from Saving Account #101

--- Current Account Details ---
Account ID: 202
Holder Name: Shafin
Address: AMU
Max Withdrawal Limit: 100000.0
-----
Depositing 50000.00 into Current Account #202
Withdrawning 25000.00 from Current Account #202
```

6. Create a class Shape with a method area(). Create two derived classes Rectangle and Circle that extend Shape. Each class should override the area() method to calculate the area of the respective shape. Create objects of Rectangle and Circle and call their area() methods.

The screenshot shows a Java code editor with four tabs at the top: Main.java, Shape.java, Circle.java, and Rectangle.java. The Main.java tab is active, displaying the following code:

```
1 package week9_6;
2
3 public class Main {
4     public static void main(String[] args) {
5         Rectangle rect = new Rectangle(length: 12.5, width: 4.0);
6         Circle circle = new Circle(radius: 7.5);
7
8         rect.area();
9         circle.area();
10    }
11 }
```

The Shape.java tab is also visible, showing the following code:

```
1 package week9_6;
2
3 class Shape {
4     public void area() {
5         System.out.println("Area of Shape is...");
```

WEEK 9

```
>Main.java ×  Shape.java ×  Circle.java ×  Rectangle.java
1 package week9_6;
2
3 class Circle extends Shape { 2 usages
4     private double radius; 3 usages
5
6     public Circle(double radius) { 1 usage
7         this.radius = radius;
8     }
9
10    @Override 2 usages
11    public void area() {
12        double result = Math.PI * radius * radius;
13        System.out.printf("Area of Circle is: %.2f\n", result);
14    }
15}
16

>Main.java  Shape.java  Circle.java ×  Rectangle.java ×
1 package week9_6;
2
3 class Rectangle extends Shape { 2 usages
4     private double length; 2 usages
5     private double width; 2 usages
6
7     public Rectangle(double length, double width) { 1 usage
8         this.length = length;
9         this.width = width;
10    }
11
12    @Override 2 usages
13    public void area() {
14        double result = length * width;
15        System.out.println("Area of Rectangle is: " + result);
16    }
17}
```

Area of Rectangle is: 50.0

Area of Circle is: 176.71

7. Create a class Employee with data members: name, salary, and a method showDetails(). Create a class Manager that extends Employee with an additional data member department. Override the showDetails() method in Manager to display all details, including department. Create an object of Manager and call showDetails().

WEEK 9

The screenshot shows a Java code editor with three tabs: Main.java, Employee.java, and Manager.java. The Main.java tab is active, displaying the following code:

```
in.java x Employee.java Manager.java
package week9_7;

public class Main {
    public static void main(String[] args) {
        Manager m1 = new Manager("Mohd Nadeem", 120000.0, "CS");
        System.out.println("Manager Info:");
        m1.showDetails();
    }
}
```

The Employee.java tab shows:

```
i.java Employee.java Manager.java
package week9_7;

class Employee {
    String name;
    double salary;
}

public Employee(String name, double salary) {
    this.name = name;
    this.salary = salary;
}

public void showDetails() {
    System.out.println("Name: " + this.name);
    System.out.println("Salary: " + this.salary);
}
```

The Manager.java tab shows:

```
Main.java Employee.java Manager.java
package week9_7;

class Manager extends Employee {
    String department;

    public Manager(String name, double salary, String department) {
        super(name, salary);
        this.department = department;
    }

    @Override
    public void showDetails() {
        super.showDetails();
        System.out.println("Department: " + this.department);
    }
}
```

A callout box highlights the output of the Manager.showDetails() method:

```
Manager Info:
Name: Mohd Nadeem
Salary: 120000.0
Department: CS
```

8. Create an abstract class Appliance with data members brand, power and abstract methods turnOn() and turnOff(). Create two derived classes WashingMachine and Refrigerator that provide their own implementations of turnOn() and turnOff(). Create objects of WashingMachine and Refrigerator and call their methods.

WEEK 9

The screenshot shows a Java code editor with three tabs: Main.java, Appliance.java, and Refrigerator.java. The Main.java tab is active.

```
1 package week9_8;
2
3 public class Main {
4     public static void main(String[] args) {
5         WashingMachine w = new WashingMachine( brand: "Bosch", power: 2200);
6         System.out.println("Washing Machine:");
7         w.displayInfo();
8         w.turnOn();
9         w.turnOff();
10
11         System.out.println();
12
13         Refrigerator fridge = new Refrigerator( brand: "Whirlpool", power: 180);
14         System.out.println("Refrigerator: ");
15         fridge.displayInfo();
16         fridge.turnOn();
17         fridge.turnOff();
18     }
19 }
```

The Appliance.java tab is selected.

```
1 package week9_8;
2
3 abstract class Appliance { 2 usages 2 inheritors
4     String brand; 2 usages
5     int power; 2 usages
6
7     public Appliance(String brand, int power) { 2 usages
8         this.brand = brand;
9         this.power = power;
10    }
11
12     public abstract void turnOn(); 2 usages 2 implementations
13     public abstract void turnOff(); 2 usages 2 implementations
14
15     public void displayInfo() { 2 usages
16         System.out.println("Brand: " + this.brand);
17         System.out.println("Power Consumption: " + this.power + "W");
18    }
19 }
```

The Refrigerator.java tab is selected.

```
1 package week9_8;
2
3 class Refrigerator extends Appliance { 2 usages
4     public Refrigerator(String brand, int power) { 1 usages
5         super(brand, power);
6    }
7
8     @Override 2 usages
9     public void turnOn() {
10        System.out.println("The fridge is on.");
11    }
12
13     @Override 2 usages
14     public void turnOff() {
15        System.out.println("The fridge is off.");
16    }
17 }
```

WEEK 9

```
>Main.java Appliance.java Refrigerator.java WashingMachine.java
1 package week9_8;
2
3 class WashingMachine extends Appliance { 2 usages
4     public WashingMachine(String brand, int power) { 1 usage
5         super(brand, power);
6     }
7
8     @Override 2 usages
9     public void turnOn() {
10        System.out.println("The washing machine is on.");
11    }
12
13     @Override 2 usages
14    public void turnOff() {
15        System.out.println("The washing machine is off.");
16    }
17 }
```

```
Washing Machine:
Brand: Bosch
Power Consumption: 2200W
The washing machine is on.
The washing machine is off.
```

```
Refrigerator:
Brand: Whirlpool
Power Consumption: 180W
The fridge is on.
The fridge is off.
```

9. Create a class MathOperations with two static methods: findGCD(int a, int b) to calculate the greatest common divisor and findLCM(int a, int b) to calculate the least common multiple. Call these methods without creating an object of MathOperations.

WEEK 9

```
Main.java
package week9_9;

public class Main {
    public static void main(String[] args) {
        int number1 = 48;
        int number2 = 18;

        int gcd = MathOperations.findGCD(number1, number2);
        System.out.println("GCD: " + gcd);

        int lcm = MathOperations.findLCM(number1, number2);
        System.out.println("LCM: " + lcm);
    }
}

MathOperations.java
package week9_9;

class MathOperations {
    public static int findGCD(int a, int b) {
        while (b != 0) {
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
    }

    public static int findLCM(int a, int b) {
        if (a == 0 || b == 0) {
            return 0;
        }
        return Math.abs(a * b) / findGCD(a, b);
    }
}
```

GCD: 6
LCM: 144

10. Create a class Student with data members rollNo, name, marks. Add a static variable schoolName. Create static method changeSchoolName() to update schoolName. Demonstrate how the static variable is shared among all objects.

```
1 package week9_10;
2
3 public class Main {
4     public static void main(String[] args) {
5
6         Student student1 = new Student( rollNo: 508, name: "Shaheer", marks: 95.5);
7         Student student2 = new Student( rollNo: 548, name: "Shafin", marks: 89.0);
8
9         System.out.println("Initial Details of All Students\n");
10        student1.displayDetails();
11        student2.displayDetails();
12
13        System.out.println();
14
15        Student.changeSchoolName( newName: "JMI");
16
17        System.out.println("Details After Updating School Name\n");
18        student1.displayDetails();
19        student2.displayDetails();
20    }
21 }
22 }
```

WEEK 9

```
>Main.java      Student.java ×
1 package week9_10;
2
3 class Student { 5 usages
4     int rollNo; 2 usages
5     String name; 2 usages
6     double marks; 2 usages
7     static String schoolName = "AMU"; 2 usages
8
9     public Student(int rollNo, String name, double marks) { 2 usages
10         this.rollNo = rollNo;
11         this.name = name;
12         this.marks = marks;
13     }
14
15     >     public static void changeSchoolName(String newName) { schoolName = newName; }
16
17     public void displayDetails() { 4 usages
18         System.out.println("Roll No: " + this.rollNo);
19         System.out.println("Name: " + this.name);
20         System.out.println("Marks: " + this.marks);
21         System.out.println("School Name: " + schoolName);
22         System.out.println("-----");
23     }
24
25 }
26
27 }
```

Initial Details of All Students

```
Roll No: 508
Name: Shaheer
Marks: 95.5
School Name: AMU
-----
Roll No: 548
Name: Shafin
Marks: 89.0
School Name: AMU
-----
```

Details After Updating School Name

```
Roll No: 508
Name: Shaheer
Marks: 95.5
School Name: JMI
-----
Roll No: 548
Name: Shafin
Marks: 89.0
School Name: JMI
-----
```