

## Question:01

```
10 public class BankAccount {
11     protected String accountNumber;
12     protected String accountHolder;
13     protected double balance;
14
15
16     public BankAccount(String accountNumber, String accountHolder, double balance) {
17         this.accountNumber = accountNumber;
18         this.accountHolder = accountHolder;
19         this.balance = balance;
20     }
21     public void deposit(double amount) {
22         balance += amount;
23         System.out.println("Deposited TZS " + amount + ". New balance: TZS " + balance);
24     }
25     public void withdraw(double amount) {
26         if (balance >= amount) {
27             balance -= amount;
28             System.out.println("Withdrew TZS " + amount + ". New balance: TZS " + balance);
29         } else {
30             System.out.println("Insufficient funds to withdraw " + amount);
31         }
32     }
33     public double getBalance() {
34         return balance;
35     }
36     public void displayAccountInfo() {
37         System.out.println("Account Number: " + accountNumber);
38         System.out.println("Account Holder: " + accountHolder);
39         System.out.println("Balance: TZS " + balance);
40     }
41     public double calculateInterest() {
42         return 0;
43     }
44 }
45 public class SavingsAccount extends BankAccount {
46     private double interestRate;
47     public SavingsAccount(String accountNumber, String accountHolder, double balance, double interestRate) {
48         super(accountNumber, accountHolder, balance);
49         this.interestRate = interestRate;
50     }
51     @Override
52     public double calculateInterest() {
53         return balance * interestRate;
54     }
55     @Override
56     public void withdraw(double amount) {
57         if (balance - amount >= 10000) {
58             super.withdraw(amount);
59         } else {
60             System.out.println("Cannot withdraw. Minimum balance of TZS 10,000 must be maintained.");
61         }
62     }
63     public void applyInterest() {
64         double interest = calculateInterest();
65         balance += interest;
66         System.out.println("Interest of TZS " + interest + " applied. New balance: TZS " + balance);
67     }
68 }
69 public class CurrentAccount extends BankAccount {
70     private double overdraftLimit;
71     public CurrentAccount(String accountNumber, String accountHolder, double balance, double overdraftLimit) {
72         super(accountNumber, accountHolder, balance);
73         this.overdraftLimit = overdraftLimit;
74     }
75     @Override
76     public double calculateInterest() {
77         return 0;
78     }
79     @Override
80     public void withdraw(double amount) {
81         if (balance + overdraftLimit >= amount) {
82             balance -= amount;
83             System.out.println("Withdrew TZS " + amount + ". New balance: TZS " + balance);
84         } else {
85             System.out.println("Insufficient funds, including overdraft limit.");
86         }
87     }
88     public boolean isOverdrawn() {
89         return balance < 0;
90     }
91 }
92 public class FixedDepositAccount extends BankAccount {
93     private double interestRate;
94     private int maturityMonths;
95     private boolean isMatured;
```

```

96     public FixedDepositAccount(String accountNumber, String accountHolder, double balance, double interestRate, int maturityMonths) {
97         super(accountNumber, accountHolder, balance);
98         this.interestRate = interestRate;
99         this.maturityMonths = maturityMonths;
100        this.isMatured = false;
101    }
102    @Override
103    public double calculateInterest() {
104        return balance * interestRate * (maturityMonths / 12.0);
105    }
106    @Override
107    public void withdraw(double amount) {
108        if (isMatured) {
109            super.withdraw(amount);
110        } else {
111            System.out.println("Account not matured yet. Cannot withdraw.");
112        }
113    }
114    public void checkMaturity() {
115        isMatured = true;
116        System.out.println("Account is now matured.");
117    }
118    public double getMaturityAmount() {
119        return balance + calculateInterest();
120    }
121 }
122 import java.util.ArrayList;
123 public class Bank {
124     private ArrayList<BankAccount> accounts = new ArrayList<>();
125     public void addAccount(BankAccount account) {
126         accounts.add(account);
127     }
128     public double getTotalDeposits() {
129         double total = 0;
130         for (BankAccount account : accounts) {
131             total += account.getBalance();
132         }
133         return total;
134     }
135     public double getTotalInterest() {
136         double totalInterest = 0;
137         for (BankAccount account : accounts) {
138             totalInterest += account.calculateInterest();
139         }
140         return totalInterest;
141     }
142     public void displayAllAccounts() {
143         for (BankAccount account : accounts) {
144             account.displayAccountInfo();
145             System.out.println("-----");
146         }
147     }
148 }
149 }
150 public class Exercise13_Polymorphism {
151     public static void transferMoney(BankAccount from, BankAccount to, double amount) {
152         if (from.getBalance() >= amount) {
153             from.withdraw(amount);
154             to.deposit(amount);
155             System.out.println("Transferred TZS " + amount + " from " + from.accountNumber + " to " + to.accountNumber);
156         } else {
157             System.out.println("Insufficient balance for transfer.");
158         }
159     }
160     public static void transferMoney(BankAccount from, BankAccount to, double amount, String description) {
161         if (from.getBalance() >= amount) {
162             from.withdraw(amount);
163             to.deposit(amount);
164             System.out.println("Transferred TZS " + amount + " from " + from.accountNumber + " to " + to.accountNumber + ". Description: " + description);
165         } else {
166             System.out.println("Insufficient balance for transfer.");
167         }
168     }
169 }
170 public static void transferMoney(BankAccount from, String toAccountNumber, double amount, Bank bank) {
171     BankAccount to = null;
172     for (BankAccount account : bank.accounts) {
173         if (account.accountNumber.equals(toAccountNumber)) {
174             to = account;
175             break;
176         }
177     }
178 }
```

```
79     if (to != null) {
80         transferMoney(from, to, amount);
81     } else {
82         System.out.println("Account not found: " + toAccountNumber);
83     }
84 }
85
86 public static void main(String[] args) {
87     System.out.println("== BANKING SYSTEM TEST ==\n");
88     SavingsAccount savings = new SavingsAccount("SAV001", "Ali Hassan", 500000, 0.05);
89     CurrentAccount current = new CurrentAccount("CUR001", "Fatma Said", 1000000, 500000);
90     FixedDepositAccount fixed = new FixedDepositAccount("FD001", "Omar Juma", 2000000, 0.08, 12);
91
92     savings.displayAccountInfo();
93     savings.deposit(100000);
94     savings.withdraw(50000);
95     savings.applyInterest();
96     System.out.println("Interest earned: TZS " + savings.calculateInterest());
97
98     Bank bank = new Bank();
99     bank.addAccount(savings);
100    bank.addAccount(current);
101    bank.addAccount(fixed);
102
103    transferMoney(savings, current, 50000);
104    transferMoney(current, savings, 30000, "Rent payment");
105
106    System.out.println("\n== END OF TEST ==");
107 }
108
109 }
```

## Question:02

```
8 public class Circle {  
9     private double radius;  
10    private String color;  
11    public Circle() {  
12        this.radius = 1.0;  
13        this.color = "red";  
14    }  
15    public Circle(double radius) {  
16        this.radius = radius;  
17        this.color = "red";  
18    }  
19    public Circle(double radius, String color) {  
20        this.radius = radius;  
21        this.color = color;  
22    }  
23    public double getRadius() {  
24        return radius;  
25    }  
26    public void setRadius(double radius) {  
27        this.radius = radius;  
28    }  
29    public String getColor() {  
30        return color;  
31    }  
32    public void setColor(String color) {  
33        this.color = color;  
34    }  
35    public double getArea() {  
36        return Math.PI * radius * radius;  
37    }  
38    @Override  
39    public String toString() {  
40        return "Circle[radius=" + radius + ", color=" + color + "]";  
41    }  
42 }  
43 public class Cylinder extends Circle {  
44     private double height;  
45     public Cylinder() {  
46         super();  
47         this.height = 1.0;  
48     }  
49     public Cylinder(double radius) {  
50         super(radius);  
51         this.height = 1.0;  
52     }  
53     public Cylinder(double radius, double height) {
```

```

54     super(radius);
55     this.height = height;
56   }
57   public Cylinder(double radius, double height, String color) {
58     super(radius, color);
59     this.height = height;
60   }
61   public double getHeight() {
62     return height;
63   }
64   public void setHeight(double height) {
65     this.height = height;
66   }
67   public double getVolume() {
68     return getArea() * height;
69   }
70   @Override
71   public String toString() {
72     return "Cylinder[" + super.toString() + ", height=" + height + "]";
73   }
74 }
75 public class Lab1CircleCylinder {
76   public static void main(String[] args) {
77     System.out.println("=====");
78     System.out.println(" Lab 1: Circle and Cylinder Hierarchy");
79     System.out.println("=====\n");
80     System.out.println("--- Section 1: Basic Object Creation ---");
81     Circle c1 = new Circle(5.0, "blue");
82     System.out.println("Circle: " + c1);
83     System.out.println("Area: " + c1.getArea());
84     Cylinder cyl = new Cylinder(5.0, 10.0, "green");
85     System.out.println("\nCylinder: " + cyl);
86     System.out.println("Base Area: " + cyl.getArea());
87     System.out.println("Volume: " + cyl.getVolume());
88     System.out.println("\n--- Section 2: Upcasting ---");
89     Circle c2 = new Cylinder(3.0, 7.0, "yellow");
90     System.out.println("c2 is a: " + c2.getClass().getSimpleName());
91     System.out.println("c2.toString(): " + c2);
92     System.out.println("c2.getArea(): " + c2.getArea());
93     System.out.println("c2.getRadius(): " + c2.getRadius());
94     System.out.println("\n--- Section 3: Downcasting ---");
95     Circle c3 = new Cylinder(4.0, 8.0, "purple");
96     Cylinder cy2 = (Cylinder) c3;
97     System.out.println("After downcast: " + cy2);
98     System.out.println("Now we can call getVolume(): " + cy2.getVolume());
99
100    System.out.println("\n--- Section 4: instanceof Operator ---");
101
102    Circle[] shapes = {
103      new Circle(2.0, "red"),
104      new Cylinder(3.0, 5.0, "blue"),
105      new Circle(4.0, "green"),
106      new Cylinder(1.0, 10.0, "orange")
107    };
108
109    for (Circle shape : shapes) {
110      System.out.println(shape);
111      if (shape instanceof Cylinder) {
112        Cylinder temp = (Cylinder) shape;
113        System.out.println(" -> This is a Cylinder! Volume = " + temp.getVolume());
114      } else {
115        System.out.println(" -> This is just a Circle. Area = " + shape.getArea());
116      }
117    }
118
119    System.out.println("=====");
120    System.out.println(" End of Lab 1");
121    System.out.println("=====");
122  }
123 }
124

```

## Question:03

```
8 public class Person {
9     private String name;
10    private String address;
11    public Person(String name, String address) {
12        this.name = name;
13        this.address = address;
14    }
15    public String getName() {
16        return name;
17    }
18    public String getAddress() {
19        return address;
20    }
21    public void setAddress(String address) {
22        this.address = address;
23    }
24    @Override
25    public String toString() {
26        return "Person[name=" + name + ", address=" + address + "]";
27    }
28 }
29 public class Student extends Person {
30     private String program;
31     private int year;
32     private double fee;
33     public Student(String name, String address, String program, int year, double fee) {
34         super(name, address);
35         this.program = program;
36         this.year = year;
37         this.fee = fee;
38     }
39     public String getProgram() {
40         return program;
41     }
42     public void setProgram(String program) {
43         this.program = program;
44     }
45     public int getYear() {
46         return year;
47     }
48     public void setYear(int year) {
49         this.year = year;
50     }
51 }
52
53
54     public double getFee() {
55         return fee;
56     }
57
58     public void setFee(double fee) {
59         this.fee = fee;
60     }
61
62     @Override
63     public String toString() {
64         return "Student[Person[name=" + getName() + ", address=" + getAddress() + "], program=" + program + ", year=" + year + ", fee=" + fee + "]";
65     }
66 }
67 public class Staff extends Person {
68     private String department;
69     private double salary;
70     public Staff(String name, String address, String department, double salary) {
71         super(name, address);
72         this.department = department;
73         this.salary = salary;
74     }
75     public String getDepartment() {
76         return department;
77     }
78     public void setDepartment(String department) {
79         this.department = department;
80     }
81     public double getSalary() {
82         return salary;
83     }
84
85     public void setSalary(double salary) {
86         this.salary = salary;
87     }
88
89     @Override
90     public String toString() {
91         return "Staff[Person[name=" + getName() + ", address=" + getAddress() + "], department=" + department + ", salary=" + salary + "]";
92     }
93 }
94 public class Lab2_PersonStudentStaff {
95     public static void main(String[] args) {
96         System.out.println("=====Lab 2: Person, Student, and Staff Hierarchy");
97         System.out.println(" Lab 2: Person, Student, and Staff Hierarchy");
98         System.out.println("=====\\n");
99 }
```

```
98
99     System.out.println("-----\n");
100    System.out.println("--- Section 1: Creating Objects ---");
101    Person p1 = new Person("Amina Hassan", "Stonetown, Zanzibar");
102    System.out.println(p1);
103    Student s1 = new Student("Juma Ali", "Chwaka, Zanzibar", "BITA", 2, 1500000);
104    Student s2 = new Student("Fatma Omar", "Mbweni, Zanzibar", "BCS", 1, 1800000);
105    System.out.println(s1);
106    System.out.println(s2);
107    Staff staff1 = new Staff("Dr. Khalid Salum", "Vuga, Zanzibar", "SCCMS", 3500000);
108    System.out.println(staff1);
109    System.out.println("\n--- Section 2: Inheritance in Action ---");
110    System.out.println("Student name: " + s1.getName());
111    System.out.println("Student address: " + s1.getAddress());
112    System.out.println("Student program: " + s1.getProgram());
113
114    System.out.println("\nStaff name: " + staff1.getName());
115    System.out.println("Staff department: " + staff1.getDepartment());
116    s1.setAddress("Fumba, Zanzibar");
117    System.out.println("\nAfter address change: " + s1);
118    System.out.println("\n--- Section 3: Polymorphism ---");
119    Person[] people = {
120        new Person("Bakari Juma", "Mwanakwerekwe, Zanzibar"),
121        new Student("Zainab Moh'd", "Kiembe Samaki, Zanzibar", "BITA", 3, 1500000),
122        new Student("Hassan Said", "Amani, Zanzibar", "BCS", 1, 1800000),
123        new Staff("Prof. Mwanaisha Ali", "Mazizini, Zanzibar", "SCCMS", 4500000)
124    };
125
126    System.out.println("All people at SUZA:");
```

```
127
128    System.out.println("All people at SUZA:");
129    for (Person p : people) {
130        System.out.println(" " + p);
131    }
132    System.out.println("\n--- Section 4: instanceof and Type Checking ---");
133    int studentCount = 0;
134    int staffCount = 0;
135
136    for (Person p : people) {
137        if (p instanceof Student) {
138            Student s = (Student) p;
139            System.out.println(s.getName() + " is a Student in " + s.getProgram() + " Year " + s.getYear());
140            studentCount++;
141        } else if (p instanceof Staff) {
142            Staff st = (Staff) p;
143            System.out.println(st.getName() + " is Staff in " + st.getDepartment());
144            staffCount++;
145        } else {
146            System.out.println(p.getName() + " is a Person (visitor/other)");
147        }
148    }
149
150    System.out.println("\nSummary: " + studentCount + " students, " + staffCount + " staff members");
151    System.out.println("=====");
```

## Question:04

```
8 public class Point {  
9     private double x = 0.0;  
10    private double y = 0.0;  
11    public Point() {  
12    }  
13    public Point(double x, double y) {  
14        this.x = x;  
15        this.y = y;  
16    }  
17    public double getX() {  
18        return x;  
19    }  
20    public void setX(double x) {  
21        this.x = x;  
22    }  
23  
24    public double getY() {  
25        return y;  
26    }  
27  
28    public void setY(double y) {  
29        this.y = y;  
30    }  
31    public void setXY(double x, double y) {  
32        this.x = x;  
33        this.y = y;  
34    }  
35    public double[] getXY() {  
36        return new double[]{x, y};  
37    }  
38    @Override  
39    public String toString() {  
40        return "(" + x + ", " + y + ")";  
41    }  
42 }  
43 public class MovablePoint extends Point {  
44  
45    private double xSpeed = 0.0;  
46    private double ySpeed = 0.0;  
47  
48    public MovablePoint() {  
49    }  
50    public MovablePoint(double xSpeed, double ySpeed) {  
51        this.xSpeed = xSpeed;  
52        this.ySpeed = ySpeed;  
53    }  
54 }
```

```

55     super(x, y);
56     this.xSpeed = xSpeed;
57     this.ySpeed = ySpeed;
58 }
59 public double getXSpeed() {
60     return xSpeed;
61 }
62
63 public void setXSpeed(double xSpeed) {
64     this.xSpeed = xSpeed;
65 }
66
67 public double getYSpeed() {
68     return ySpeed;
69 }
70
71 public void setYSpeed(double ySpeed) {
72     this.ySpeed = ySpeed;
73 }
74 public void setSpeed(double xSpeed, double ySpeed) {
75     this.xSpeed = xSpeed;
76     this.ySpeed = ySpeed;
77 }
78 public double[] getSpeed() {
79     return new double[]{xSpeed, ySpeed};
80 }
81 public MovablePoint move() {
82     setX(getX() + xSpeed);
83     setY(getY() + ySpeed);
84     return this;
85 }
86 @Override
87 public String toString() {
88     return super.toString() + " speed=(" + xSpeed + ", " + ySpeed + ")";
89 }
90 }
91 public class Lab3_PointMovablePoint {
92     public static void main(String[] args) {
93         System.out.println("=====");
94         System.out.println(" Lab 3: Point and MovablePoint");
95         System.out.println("=====\n");
96
97         System.out.println("--- Section 1: Point Objects ---");
98
99         Point p1 = new Point();
100        System.out.println("Default point: " + p1);

```

```

101
102     Point p2 = new Point(3.0, 4.0);
103     System.out.println("Point at (3, 4): " + p2);
104
105     p2.setX(5.0);
106     p2.setY(6.0);
107     System.out.println("After setX(5), setY(6): " + p2);
108
109     double[] coords = p2.getXY();
110     System.out.println("getXY() = [" + coords[0] + ", " + coords[1] + "]");
111     System.out.println("\n--- Section 2: MovablePoint Objects ---");
112     MovablePoint mp1 = new MovablePoint(0.0, 0.0, 2.0, 3.0);
113     System.out.println("Initial position: " + mp1);
114     System.out.println("X coordinate: " + mp1.getX());
115     System.out.println("Y coordinate: " + mp1.getY());
116     System.out.println("\n--- Section 3: Movement ---");
117     System.out.println("Before move: " + mp1);
118     mp1.move();
119     System.out.println("After 1st move: " + mp1);
120     mp1.move();

```

```

1 System.out.println("After 2nd move: " + mp1);
2 mp1.move();
3 System.out.println("After 3rd move: " + mp1);
4 mp1.setSpeed(1.0, -1.0);
5 System.out.println("\nSpeed changed to (1.0, -1.0)");
6 mp1.move();
7 System.out.println("After move: " + mp1);
8 mp1.move();
9 System.out.println("After move: " + mp1);
10 System.out.println("\n--- Section 4: Polymorphism ---");
11 Point p3 = new MovablePoint(1.0, 1.0, 0.5, 0.5);
12 System.out.println("p3 (Point ref): " + p3);
13 System.out.println("p3 class: " + p3.getClass().getSimpleName());
14
15 MovablePoint mp2 = (MovablePoint) p3;
16 mp2.move();
17 System.out.println("After downcast and move: " + mp2);
18 System.out.println("p3 also changed: " + p3);
19 }
20 System.out.println("\n--- Section 5: Simple Movement Simulation ---");
21
22 MovablePoint[] points = {
23     new MovablePoint(0.0, 0.0, 1.0, 1.0),
24     new MovablePoint(10.0, 0.0, -1.0, 0.5),
25     new MovablePoint(5.0, 5.0, 0.0, -2.0)
26 };
27
28 System.out.println("Starting positions:");
29 for (int i = 0; i < points.length; i++) {
30     System.out.println(" Point " + (i + 1) + ": " + points[i]);
31 }
32
33
34 for (int step = 1; step <= 5; step++) {
35     System.out.println("\nStep " + step + ":" );
36     for (int i = 0; i < points.length; i++) {
37         points[i].move();
38         System.out.println(" Point " + (i + 1) + ": " + points[i]);
39     }
40 }
41
42 System.out.println("\n=====");
43 System.out.println(" End of Lab 3");
44 System.out.println("=====");
45 }
```

## Question:05

```
8 public class Circle {
9     private double radius;
10    private String color;
11    public Circle() {
12        this.radius = 1.0;
13        this.color = "red";
14    }
15    public Circle(double radius) {
16        this.radius = radius;
17        this.color = "red";
18    }
19    public Circle(double radius, String color) {
20        this.radius = radius;
21        this.color = color;
22    }
23    public double getRadius() {
24        return radius;
25    }
26    public void setRadius(double radius) {
27        this.radius = radius;
28    }
29    public String getColor() {
30        return color;
31    }
32    public void setColor(String color) {
33        this.color = color;
34    }
35    public double getArea() {
36        return Math.PI * radius * radius;
37    }
38    @Override
39    public String toString() {
40        return "Circle[radius=" + radius + ", color=" + color + "]";
41    }
42}
43 public class Cylinder extends Circle {
44     private double height;
45     public Cylinder() {
46         super();
47         this.height = 1.0;
48     }
49     public Cylinder(double radius) {
50         super(radius);
51         this.height = 1.0;
52     }
53     public Cylinder(double radius, double height) {
```

```

54     super(radius);
55     this.height = height;
56   }
57   public Cylinder(double radius, double height, String color) {
58     super(radius, color);
59     this.height = height;
60   }
61   public double getHeight() {
62     return height;
63   }
64   public void setHeight(double height) {
65     this.height = height;
66   }
67   public double getVolume() {
68     return getArea() * height;
69   }
70   @Override
71   public String toString() {
72     return "Cylinder[" + super.toString() + ", height=" + height + "]";
73   }
74 }
75 public class Lab1_CircleCylinder {
76   public static void main(String[] args) {
77     System.out.println("=====");
78     System.out.println(" Lab 1: Circle and Cylinder Hierarchy");
79     System.out.println("=====\n");
80     System.out.println("--- Section 1: Basic Object Creation ---");
81     Circle c1 = new Circle(5.0, "blue");
82     System.out.println("Circle: " + c1);
83     System.out.println("Area: " + c1.getArea());
84     Cylinder cyl1 = new Cylinder(5.0, 10.0, "green");
85     System.out.println("\nCylinder: " + cyl1);
86     System.out.println("Base Area: " + cyl1.getArea());
87     System.out.println("Volume: " + cyl1.getVolume());
88     System.out.println("\n--- Section 2: Upcasting ---");
89     Circle c2 = new Cylinder(3.0, 7.0, "yellow");
90     System.out.println("c2 is a: " + c2.getClass().getSimpleName());
91     System.out.println("c2.toString(): " + c2);
92     System.out.println("c2.getArea(): " + c2.getArea());
93     System.out.println("c2.getRadius(): " + c2.getRadius());
94     System.out.println("\n--- Section 3: Downcasting ---");
95     Circle c3 = new Cylinder(4.0, 8.0, "purple");
96     Cylinder cy2 = (Cylinder) c3;
97     System.out.println("After downcast: " + cy2);
98     System.out.println("Now we can call getVolume(): " + cy2.getVolume());
99 }

100    System.out.println("\n--- Section 4: instanceof operator ---");
101
102    Circle[] shapes = {
103      new Circle(2.0, "red"),
104      new Cylinder(3.0, 5.0, "blue"),
105      new Circle(4.0, "green"),
106      new Cylinder(1.0, 10.0, "orange")
107    };
108
109    for (Circle shape : shapes) {
110      System.out.println(shape);
111      if (shape instanceof Cylinder) {
112        Cylinder temp = (Cylinder) shape;
113        System.out.println(" -> This is a Cylinder! Volume = " + temp.getVolume());
114      } else {
115        System.out.println(" -> This is just a Circle. Area = " + shape.getArea());
116      }
117    }
118
119    System.out.println("\n=====");
120    System.out.println(" End of Lab 1");
121    System.out.println("=====");
122  }
123 }
```

## Question:06

```
8 public class Author {
9
10    private String name;
11    private String email;
12    private char gender;
13    public Author(String name, String email, char gender) {
14        this.name = name;
15        this.email = email;
16        this.gender = gender;
17    }
18    public String getName() {
19        return name;
20    }
21
22    public String getEmail() {
23        return email;
24    }
25
26    public void setEmail(String email) {
27        this.email = email;
28    }
29
30    public char getGender() {
31        return gender;
32    }
33    @Override
34    public String toString() {
35        return String.format("Author[name=%s, email=%s, gender=%c]", name, email, gender);
36    }
37}
38 public class Book {
39    private String name;
40    private Author author;
41    private double price;
42    private int qty;
43    public Book(String name, Author author, double price) {
44        this.name = name;
45        this.author = author;
46        this.price = price;
47        this.qty = 0;
48    }
49
50    public Book(String name, Author author, double price, int qty) {
51        this.name = name;
52        this.author = author;
53        this.price = price;
54        this.qty = qty;
55    }
56    public String getName() {
57        return name;
58    }
59
60    public Author getAuthor() {
61        return author;
62    }
63
64    public String getAuthorName() {
65        return author.getName();
66    }
67
68    public String getAuthorEmail() {
69        return author.getEmail();
70    }
71
72    public char getAuthorGender() {
73        return author.getGender();
74    }
75
76    public double getPrice() {
77        return price;
78    }
79
80    public void setPrice(double price) {
81        this.price = price;
82    }
83}
```

```

84     public int getQty() {
85         return qty;
86     }
87
88     public void setQty(int qty) {
89         this.qty = qty;
90     }
91
92     @Override
93     public String toString() {
94         return String.format("Book[name=%s, Author[%s], price=%2f, qty=%d]", name, author, price, qty);
95     }
96 }
97
98 public class Lab5_AuthorBookComposition {
99     public static void main(String[] args) {
100         System.out.println("=====");
101         System.out.println(" Lab 5: Author and Book (Composition)");
102         System.out.println("=====\n");
103
104         System.out.println("--- Section 1: Creating Authors ---");
105
106         Author author1 = new Author("Ali Sultan", "ali.sultan@suza.ac.tz", 'm');
107         Author author2 = new Author("Mwanaisha Bakari", "mwanaisha.b@suza.ac.tz", 'f');
108         Author author3 = new Author("Hamad Khamis", "hamad.k@gmail.com", 'm');
109
110         System.out.println(author1);
111         System.out.println(author2);
112         System.out.println(author3);
113
114
115         System.out.println("\n--- Section 2: Creating Books ---");
116
117         Book book1 = new Book("Introduction to Java", author1, 35000, 50);
118         Book book2 = new Book("Data Structures in Java", author2, 42000, 30);
119
120         System.out.println(book1);
121         System.out.println(book2);
122
123
124         System.out.println("\n--- Section 3: Accessing Through Composition ---");
125
126         System.out.println("Book: " + book1.getName());
127         System.out.println("Author name: " + book1.getAuthorName());
128         System.out.println("Author email: " + book1.getAuthorEmail());
129
130

```

```

32     Author bookAuthor = book1.getAuthor();
33     System.out.println("Author object: " + bookAuthor);
34
35
36     System.out.println("\n--- Section 4: Shared Author References ---");
37
38
39     Book book3 = new Book("Advanced Java Programming", author1, 55000, 20);
40     System.out.println("Book 1 author: " + book1.getAuthorName());
41     System.out.println("Book 3 author: " + book3.getAuthorName());
42     System.out.println("Same author? " + (book1.getAuthor() == book3.getAuthor()));
43
44
45     author1.setEmail("ali.sultan.new@suza.ac.tz");
46     System.out.println("\nAfter changing author1's email:");
47     System.out.println("Book 1 author email: " + book1.getAuthorEmail());
48     System.out.println("Book 3 author email: " + book3.getAuthorEmail());
49     System.out.println("Both changed! Because they share the same Author object.");
50
51
52     System.out.println("\n--- Section 5: Creating Book with Anonymous Author ---");
53
54     Book book4 = new Book(
55         "Python for Beginners",
56         new Author("Salma Haji", "salma.h@suza.ac.tz", 'f'),
57         28000,
58         100
59     );
59     System.out.println(book4);

```

```
160     ,
161     System.out.println(book4);
162     System.out.println("Author: " + book4.getAuthorName());
163
164     System.out.println("\n--- Section 6: Book Inventory ---");
165
166     Book[] inventory = {book1, book2, book3, book4};
167
168     System.out.println("SUZA Bookshop Inventory:");
169     System.out.println(String.format("%-30s %-25s %10s %5s",
170         "Title", "Author", "Price(TZS)", "Qty"));
171     System.out.println("-".repeat(75));
172
173     double totalValue = 0;
174     for (Book book : inventory) {
175         System.out.println(String.format("%-30s %-25s ,10.0f %5d",
176             book.getName(), book.getAuthorName(),
177             book.getPrice(), book.getQty()));
178         totalValue += book.getPrice() * book.getQty();
179     }
180     System.out.println("-".repeat(75));
181     System.out.println(String.format("Total inventory value: TZS %,.0f", totalValue));
182
183     System.out.println("\n=====");
184     System.out.println(" End of Lab 5");
185     System.out.println("=====");
186 }
187 }
```