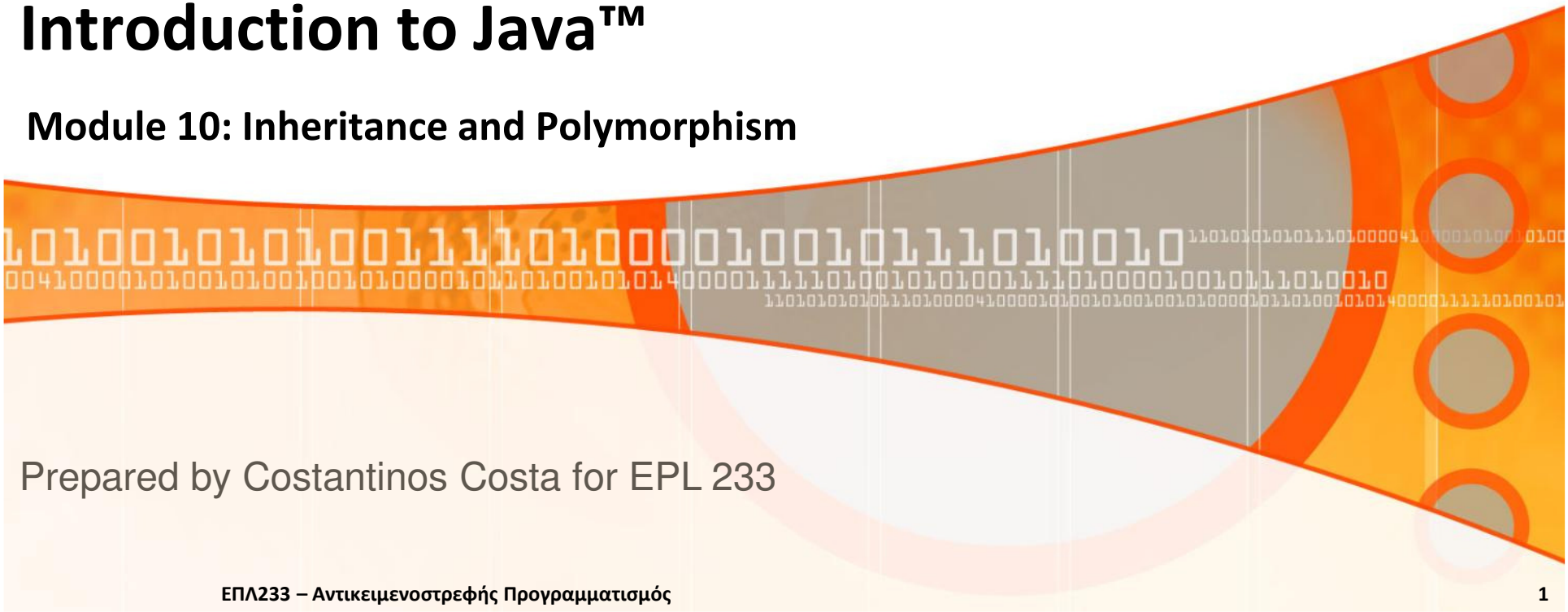
Introduction to Java™

Module 10: Inheritance and Polymorphism



Prepared by Costantinos Costa for EPL 233

Multiple-choice Questions On Inheritance And Polymorphism (Classes)

```
public class BankAccount {
    private double myBalance;
    public BankAccount() {
    myBalance = 0;
    }

    public BankAccount(double balance) {
    myBalance = balance;
    }

    public void deposit(double amount) {
    myBalance += amount;
    }

    public void withdraw(double amount) {
    myBalance -= amount;
    }

    public double getBalance() {
    return myBalance;
    }
}
```

```
public class SavingsAccount extends
BankAccount {
    private double myInterestRate;
    public SavingsAccount() { /* implementation
    not shown */}

    public SavingsAccount(double balance, double
    rate) { /*implementation notshown */}
    // Add interest to balance
    public void addInterest(){ /* implementation
    not shown */}}
```

```
public class CheckingAccount extends
BankAccount {
    private static final double FEE = 2.0;
    private static final double MIN_BALANCE =
    50.0;

    public CheckingAccount(double balance) { /*
    implementation not shown */}
    /* FEE of $2 deducted if withdrawal leaves
    balance less than MIN_BALANCE.Allows for
    negative balance.
     */
    public void withdraw(double amount) { /*
    implementation not shown */
    }}
```

Multiple-choice Questions On Inheritance And Polymorphism

1. Of the methods shown, how many different nonconstructor methods can be invoked by a SavingsAccount object?

   A. 1
   B. 2
   C. 3
   D. 4
   E. 5

2. Which of the following correctly implements the default constructor of the SavingsAccount class?

```
I. myInterestRate = 0;
   super();

II. super();
    myInterestRate = 0;

III. super();
```

A. II only
B. I and II only
C. II and III only
D. III only
E. I, II, and III

3. Which is a correct implementation of the constructor with parameters in the SavingsAccount class?

A.
```
myBalance = balance;
myInterestRate = rate;
```

B.
```
getBalance() = balance;
myInterestRate = rate;
```

C.
```
super();
myInterestRate = rate;
```

D.
```
super(balance);
myInterestRate = rate;
```

E.
```
super(balance, rate);
```

4. Which is a correct implementation of the CheckingAccount constructor?

I.
```
super(balance);
```

II.
```
super();
deposit(balance);
```

III.
```
deposit(balance);
```

A. I only
B. II only
C. III only
D. II and III only
E. I, II, and III

5. Which is correct implementation code for the withdraw method in the CheckingAccount class?

A.
```
super.withdraw(amount);
if (myBalance < MIN_BALANCE)
 super.withdraw(FEE);
```

B.
```
withdraw(amount);
if (myBalance < MIN_BALANCE)
 withdraw(FEE);
```

C.
```
super.withdraw(amount);
if (getBalance() < MIN_BALANCE)
 super.withdraw(FEE);
```

D.
```
withdraw(amount);
if (getBalance() < MIN_BALANCE)
 withdraw(FEE);
```

E.
```
myBalance -= amount;
if (myBalance < MIN_BALANCE)
 myBalance -= FEE;
```

Task 1

- Write an inheritance hierarchy for classes **Quadrilateral**, **Trapezoid**, **Parallelogram**, **Rectangle** and **Square**.
- Use **Quadrilateral** as the **superclass** of the hierarchy. Specify the instance variables and methods for each class.
- The **private** instance variables of **Quadrilateral** should be the **x-y** coordinate pairs for the four endpoints of the Quadrilateral.
- Write a program that instantiates objects of your classes and outputs each object's area (**except** Quadrilateral).

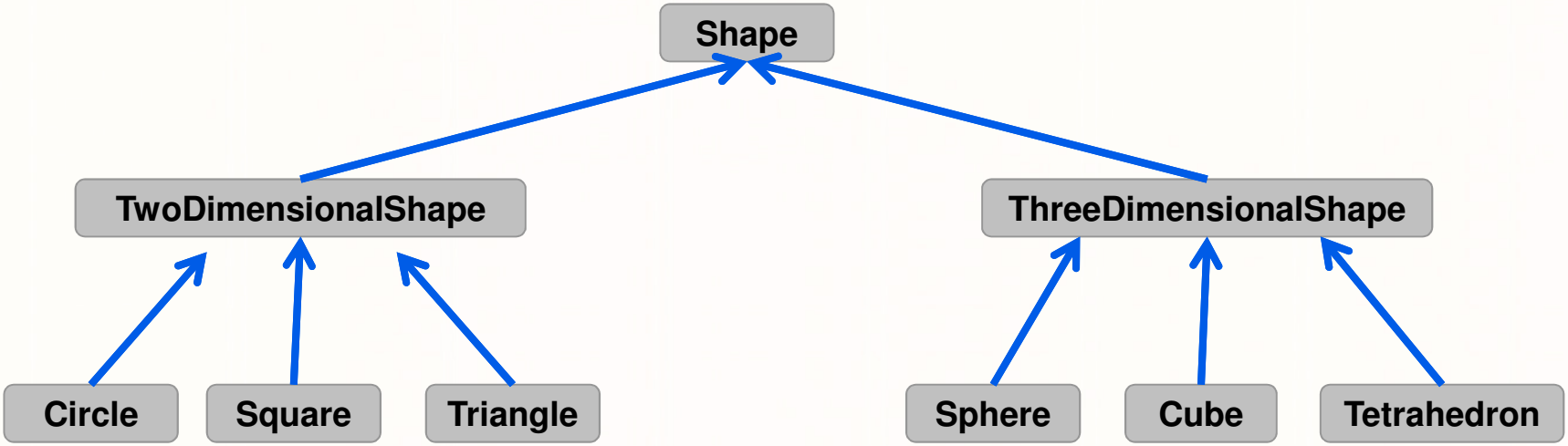
- Hints:
  - Create and use a Point class to represent the corners of the shapes.
  - Your output should appear as follows:

```
Coordinates of Quadrilateral are:
( 1.1, 1.2 ), ( 6.6, 2.8 ), ( 6.2, 9.9 ), ( 2.2, 7.4 )

Coordinates of Trapezoid are:
( 0.0, 0.0 ), ( 10.0, 0.0 ), ( 8.0, 5.0 ), ( 3.3, 5.0 )
Height is: 5.0
Area is: 36.75
```

Task 2

- Implement the Shape hierarchy shown below.



- Hint: Use keyword *"abstract"*.
- Output:

```
Circle: [22, 88] radius: 4
Circle's area is 50
Square: [71, 96] side: 10
Square's area is 100
Sphere: [8, 89] radius: 2
Sphere's area is 50
Sphere's volume is 33
Cube: [79, 61] side: 8
Cube's area is 384
Cube's volume is 512
```

Task 2

- *Each TwoDimensionalShape* should contain method **getArea** to calculate the **area** of the two-dimensional shape.
- *Each ThreeDimensionalShape* should have methods **getArea** and **getVolume** to calculate the surface **area** and **volume**, respectively, of the three-dimensional shape.
- Create a program that uses an array of Shape references to objects of each concrete class in the hierarchy. The program should print a text description of the object to which each array element refers. Also, in the loop that processes all the shapes in the array, determine whether each shape is a TwoDimensionalShape or a ThreeDimensionalShape. If it is a **TwoDimensionalShape**, display its **area**. If it is a **ThreeDimensionalShape**, display its **area** and **volume**