

Question – 1:

What is wrong with each of the following programs?

```
1 public class ShowErrors {
2     public static void main(String[] args) {
3         ShowErrors t = new ShowErrors(5);
4     }
5 }
```

(a)

```
1 public class ShowErrors {
2     public static void main(String[] args) {
3         ShowErrors t = new ShowErrors();
4         t.x();
5     }
6 }
```

(b)

```
1 public class ShowErrors {
2     public void method1() {
3         Circle c;
4         System.out.println("What is radius "
5             + c.getRadius());
6         c = new Circle();
7     }
8 }
```

(c)

```
1 public class ShowErrors {
2     public static void main(String[] args) {
3         C c = new C(5.0);
4         System.out.println(c.value);
5     }
6 }
7
8 class C {
9     int value = 2;
10 }
```

(d)

Question -2:

What is wrong in the following code?

```
1 class Test {
2     public static void main(String[] args) {
3         A a = new A();
4         a.print();
5     }
6 }
7
8 class A {
9     String s;
10
11     A(String newS) {
12         s = newS;
13     }
14
15     public void print() {
16         System.out.print(s);
17     }
18 }
```

Question-3:

What is the output of the following code?

```
public class A {
    boolean x;

    public static void main(String[] args) {
        A a = new A();
        System.out.println(a.x);
    }
}
```

Question-4:

(The *Rectangle* class) Following the example of the *Circle* class in Section 9.2, design a class named *Rectangle* to represent a rectangle. The class contains:

- Two **double** data fields named **width** and **height** that specify the width and height of the rectangle. The default values are **1** for both **width** and **height**.
- A no-arg constructor that creates a default rectangle.
- A constructor that creates a rectangle with the specified **width** and **height**.
- A method named **getArea()** that returns the area of this rectangle.
- A method named **getPerimeter()** that returns the perimeter.

Draw the UML diagram for the class and then implement the class. Write a test program that creates two *Rectangle* objects—one with width **4** and height **40** and the other with width **3.5** and height **35.9**. Display the width, height, area, and perimeter of each rectangle in this order.

Question-5:

(The *Stock* class) Following the example of the *Circle* class in Section 9.2, design a class named *Stock* that contains:

- A string data field named **symbol** for the stock's symbol.
- A string data field named **name** for the stock's name.
- A **double** data field named **previousClosingPrice** that stores the stock price for the previous day.
- A **double** data field named **currentPrice** that stores the stock price for the current time.
- A constructor that creates a stock with the specified symbol and name.
- A method named **getChangePercent()** that returns the percentage changed from **previousClosingPrice** to **currentPrice**.

Draw the UML diagram for the class and then implement the class. Write a test program that creates a *Stock* object with the stock symbol **ORCL**, the name **Oracle Corporation**, and the previous closing price of **34.5**. Set a new current price to **34.35** and display the price-change percentage.

Question-6:

(Use the *Random* class) Write a program that creates a *Random* object with seed **1000** and displays the first 50 random integers between **0** and **100** using the *nextInt(100)* method.

Question-7:

(*Stopwatch*) Design a class named *StopWatch*. The class contains:

- Private data fields *startTime* and *endTime* with getter methods.
- A no-arg constructor that initializes *startTime* with the current time.
- A method named *start()* that resets the *startTime* to the current time.
- A method named *stop()* that sets the *endTime* to the current time.
- A method named *getElapsedTime()* that returns the elapsed time for the stopwatch in milliseconds.

Draw the UML diagram for the class and then implement the class. Write a test program that measures the execution time of sorting 100,000 numbers using selection sort.