

4. At a certain high school students receive letter grades based on the following scale.

<u>Integer Score</u>	<u>Letter Grade</u>
93 or above	A
From 84 to 92 inclusive	B
From 75 to 83 inclusive	C
Below 75	F

Which of the following code segments will assign the correct string to `grade` for a given integer `score` ?

I.

```
if (score >= 93)
    grade = "A";
if (score >= 84 && score <= 92)
    grade = "B";
if (score >= 75 && score <= 83)
    grade = "C";
if (score < 75)
    grade = "F";
```

II.

```
if (score >= 93)
    grade = "A";
if (84 <= score <= 92)
    grade = "B";
if (75 <= score <= 83)
    grade = "C";
if (score < 75)
    grade = "F";
```

III.

```
if (score >= 93)
    grade = "A";
else if (score >= 84)
    grade = "B";
else if (score >= 75)
    grade = "C";
else
    grade = "F";
```

- (A) II only
- (B) III only
- (C) I and II only
- (D) I and III only
- (E) I, II, and III

5. Consider the following output.

```
1 1 1 1 1
2 2 2 2
3 3 3
4 4
5
```

Which of the following code segments will produce this output?

- (A)

```
for (int j = 1; j <= 5; j++)
{
    for (int k = 1; k <= 5; k++)
    {
        System.out.print(j + " ");
    }
    System.out.println();
}
```
- (B)

```
for (int j = 1; j <= 5; j++)
{
    for (int k = 1; k <= j; k++)
    {
        System.out.print(j + " ");
    }
    System.out.println();
}
```
- (C)

```
for (int j = 1; j <= 5; j++)
{
    for (int k = 5; k >= 1; k--)
    {
        System.out.print(j + " ");
    }
    System.out.println();
}
```
- (D)

```
for (int j = 1; j <= 5; j++)
{
    for (int k = 5; k >= j; k--)
    {
        System.out.print(j + " ");
    }
    System.out.println();
}
```
- (E)

```
for (int j = 1; j <= 5; j++)
{
    for (int k = j; k <= 5; k++)
    {
        System.out.print(k + " ");
    }
    System.out.println();
}
```

6. A car dealership needs a program to store information about the cars for sale. For each car, they want to keep track of the following information: number of doors (2 or 4), whether the car has air conditioning, and its average number of miles per gallon. Which of the following is the best object-oriented program design?
- (A) Use one class, `Car`, with three instance variables:
`int numDoors, boolean hasAir, and
double milesPerGallon.`
 - (B) Use four unrelated classes: `Car`, `Doors`, `AirConditioning`, and `MilesPerGallon`.
 - (C) Use a class `Car` with three subclasses: `Doors`, `AirConditioning`, and `MilesPerGallon`.
 - (D) Use a class `Car`, with a subclass `Doors`, with a subclass `AirConditioning`, with a subclass `MilesPerGallon`.
 - (E) Use three classes: `Doors`, `AirConditioning`, and `MilesPerGallon`, each with a subclass `Car`.
7. Consider the following declarations.

```
public interface Shape
{
    int isLargerThan(Shape other);
    // Other methods not shown
}
public class Circle implements Shape
{
    // Other methods not shown
}
```

Which of the following method headings of `isLargerThan` can be added to the declaration of the `Circle` class so that it will satisfy the `Shape` interface?

- I. `public int isLargerThan(Shape other)`
 - II. `public int isLargerThan(Circle other)`
 - III. `public boolean isLargerThan(Object other)`
- (A) I only
 - (B) II only
 - (C) III only
 - (D) I and II only
 - (E) I, II, and III

Questions 8–9 refer to the following incomplete class declaration.

```
public class TimeRecord
{
    private int hours;
    private int minutes; // 0 ≤ minutes < 60
    /** Constructs a TimeRecord object.
     * @param h the number of hours
     *         Precondition:  $h \geq 0$ 
     * @param m the number of minutes
     *         Precondition:  $0 \leq m < 60$ 
     */
    public TimeRecord(int h, int m)
    {
        hours = h;
        minutes = m;
    }

    /** @return the number of hours
     */
    public int getHours()
    { /* implementation not shown */ }

    /** @return the number of minutes
     * Postcondition:  $0 \leq \text{minutes} < 60$ 
     */
    public int getMinutes()
    { /* implementation not shown */ }

    /** Adds h hours and m minutes to this TimeRecord.
     * @param h the number of hours
     *         Precondition:  $h \geq 0$ 
     * @param m the number of minutes
     *         Precondition:  $m \geq 0$ 
     */
    public void advance(int h, int m)
    {
        hours = hours + h;
        minutes = minutes + m;
        /* missing code */
    }
    // Other methods not shown
}
```