

Writing Classes MCQ, Part A

This quiz has 15 questions.

1. The `Fraction` class below will contain two `int` attributes for the numerator and denominator of a fraction. The class will also contain a method `fractionToDecimal` that can be accessed from outside the class.

```
public class Fraction {
    /* missing code */
    // constructor not shown
    // other methods not shown
}
```

Which of the following replacements for `/* missing code */` is the most appropriate implementation of the class?

- (A) `private int numerator;`
`private int denominator;`
`private double fractionToDecimal() {`
 `return (double)numerator/denominator;`
`}`
- (B) `private int numerator;`
`private int denominator;`
`public double fractionToDecimal() {`
 `return (double)numerator/denominator;`
`}`
- (C) `public int numerator;`
`public int denominator;`
`private double fractionToDecimal() {`
 `return (double)numerator/denominator;`
`}`
- (D) `public double fractionToDecimal() {`
 `private int numerator;`
 `private int denominator;`
 `return (double)numerator/denominator;`
`}`
- (E) `public double fractionToDecimal() {`
 `public int numerator;`
 `public int denominator;`
 `return (double)numerator/denominator;`
`}`

2. The `Thing` class below will contain a `String` attribute, a constructor, and the `helper` method, which will be kept internal to the class.

```
public class Thing {
    /* missing code */
}
```

Which of the following replacements for `/* missing code */` is the most appropriate implementation of the class?

- (A) `private String str;`
`private Thing(String s)`
 `{ /* implementation not shown */ }`
`private void helper()`
 `{ /* implementation not shown */ }`
- (B) `private String str;`
`public Thing(String s)`
 `{ /* implementation not shown */ }`
`private void helper()`
 `{ /* implementation not shown */ }`
- (C) `private String str;`
`public Thing(String s)`
 `{ /* implementation not shown */ }`
`public void helper()`
 `{ /* implementation not shown */ }`
- (D) `public String str;`
`private Thing(String s)`
 `{ /* implementation not shown */ }`
`private void helper()`
 `{ /* implementation not shown */ }`
- (E) `public String str;`
`public Thing(String s)`
 `{ /* implementation not shown */ }`
`public void helper()`
 `{ /* implementation not shown */ }`

Writing Classes MCQ, Part A

3. The `Employee` class will contain a `String` attribute for an employee's name and a `double` attribute for the employee's salary.

Which is the most appropriate implementation of the class?

- Ⓐ

```
public class Employee {  
    public String name;  
    public double salary;  
    // constructor and methods not shown  
}
```
- Ⓑ

```
public class Employee {  
    public String name;  
    private double salary;  
    // constructor and methods not shown  
}
```
- Ⓒ

```
public class Employee {  
    private String name;  
    private double salary;  
    // constructor and methods not shown  
}
```
- Ⓓ

```
private class Employee {  
    public String name;  
    public double salary;  
    // constructor and methods not shown  
}
```
- Ⓔ

```
private class Employee {  
    private String name;  
    private double salary;  
    // constructor and methods not shown  
}
```

4. Consider the following definition of the class `Student`.

```
public class Student {  
    private int gradeLevel;  
    private String name;  
    private double GPA;  
    public Student(int lvl,  
                  String nm, double gr)  
    {  
        gradeLevel = lvl;  
        name = nm;  
        GPA = gr;  
    }  
}
```

Which of the following object initializations will compile without error?

- Ⓐ

```
Student max =  
    new Student("Max", 10, 3.75);
```
- Ⓑ

```
Student max =  
    new Student(3.75, "Max", 10);
```
- Ⓒ

```
Student max =  
    new Student(3.75, 10, "Max");
```
- Ⓓ

```
Student max =  
    new Student(10, "Max", 3.75);
```
- Ⓔ

```
Student max =  
    new Student(10, 3.75, "Max");
```

Writing Classes MCQ, Part A

5. Consider the following class definition. Each object of the class `Employee` will store the employee's name as `name`, the number of weekly hours worked as `wkHours`, and hourly rate of pay as `payRate`.

```
public class Employee {
    private String name;
    private int wkHours;
    private double payRate;
    public Employee(String nm,
                    int hrs, double rt)
    {
        name = nm;
        wkHours = hrs;
        payRate = rt;
    }
    public Employee(String nm,
                    double rt)
    {
        name = nm;
        wkHours = 20;
        payRate = rt;
    }
}
```

Which of the following code segments, found in a class other than `Employee`, could be used to correctly create an `Employee` object representing an employee who worked for 20 hours at a rate of \$18.50 per hour?

- I. `Employee e1 = new Employee("Lili", 20, 18.5);`
- II. `Employee e2 = new Employee("Steve", 18.5);`
- III. `Employee e3 = new Employee("Carol", 20);`
- (A) I only
- (B) III only
- (C) I and II only
- (D) I and III only
- (E) I, II, and III

6. Consider the following class definition.

```
public class Person {
    private String name;
    /* missing constructor */
}
```

The statement below, which is located in a method in a different class, creates a new `PERSON` object with its attribute `name` initialized to "Washington".

```
Person p =
    new Person("Washington");
```

Which of the following can be used to replace `/* missing constructor */` so that the object `p` is correctly created?

- (A) `private Person() { name = n; }`
- (B) `private Person(String n) { name = n; }`
- (C) `public Person() { name = n; }`
- (D) `public Person(String n) { name = n; }`
- (E) `public Person(String name) { String n = name; }`

Writing Classes MCQ, Part A

7. Consider the following method `substringFound`, which is intended to return `true` if a substring, `key`, is located at a specific index of the string `phrase`. Otherwise it should return `false`.

```
public boolean substringFound(
    String phrase,
    String key,
    int index)
{
    String part =
        phrase.substring(
            index,
            index+key.length());
    return part.equals(key);
}
```

Which of the following is the best precondition for `index` so that the method will return the appropriate result in all cases and a runtime error can be avoided?

- (A) `0 <= index < phrase.length()`
- (B) `0 <= index < key.length()`
- (C) `0 <= index < phrase.length() + key.length()`
- (D) `0 <= index <= phrase.length() - key.length()`
- (E) `0 <= index < phrase.length() - index`

8. The method `addItUp(m, n)` is intended to print the sum of the integers greater than or equal to `m` and less than or equal to `n`.

```
/* missing precondition */
public static int
addItUp( int m, int n) {
    int sum = 0;
    for(int j=m; j<=n; j++) {
        sum += j;
    }
    return sum;
}
```

Which of the following is the most appropriate precondition for the method?

- (A) `/* Precondition: m <= n */`
- (B) `/* Precondition: n <= m */`
- (C) `/* Precondition: m>=0 and n>=0 */`
- (D) `/* Precondition: m<=0 and n<=0 */`
- (E) `/* Precondition: m<=0 and n>= 0 */`

9. Consider the following method.

```
/* missing precondition */
public void someMethod(int j, int k,
    String oldS) {
    String newS =
        oldS.substring(j,k);
    System.out.println("New String: " +
        newS);
}
```

Which of the following is the most appropriate precondition for `someMethod` so that the call to `substring` does not throw an exception?

- (A) `/* Precondition: 0 <= oldS.length() */`
- (B) `/* Precondition: 0<j and 0<k */`
- (C) `/* Precondition: 0<=j and 0<=k */`
- (D) `/* Precondition: j <= k */`
- (E) `/* Precondition: 0<=j<=k<=oldS.length() */`

Writing Classes MCQ, Part A

10. Consider the following class declaration.

```
public class Student {
    private String name;
    private int age;
    public Student(String n, int a) {
        name = n;
        age = a;
    }
    public boolean isOlderThan5() {
        if(age>5) {
            return true;
        }
    }
}
```

Which of the following best described the reason this code segment will not compile?

- (A) The return type for the `isOlderThan5` method should be `void`.
- (B) The return type for the `isOlderThan5` method should be `String`.
- (C) The return type for the `isOlderThan5` method should be `int`.
- (D) The `isOlderThan5` method is missing a `return` statement for the case when `age` is less than or equal to 5.
- (E) The `isOlderThan5` method should receive variable `age` as a parameter.

11. Consider the following class definition.

```
public class Pet {
    private String name;
    private int age;
    public Pet(String str, int a) {
        name = str;
        age = a;
    }
    public getName() {
        return name;
    }
}
```

Which choice correctly explains why this class definition fails to compile?

- (A) The class is missing a mutator method.
- (B) The class is missing an accessor method.
- (C) The accessor method is missing a return type.
- (D) The accessor method returns a variable other than an instance variable.
- (E) The instance variables should be designated `public` instead of `private`.

Writing Classes MCQ, Part A

12. Consider the following class.

```
public class Help {
    private int h;
    public HELP(int newH) {
        h = newH;
    }
    public double getH() {
        return h;
    }
}
```

The `getH` method is intended to return the value of the instance variable `h`. The following code segment shows an example of creating and using a `Help` object.

```
Help h1 = new Help(5);
int x = h1.getH();
System.out.println(x);
```

Which of the following statements best explains why the `getH` method does not work as intended?

- (A) The `getH` method should have a `double` parameter.
- (B) The `getH` method should have an `int` parameter.
- (C) The `getH` method should return `newH` instead of `h`.
- (D) The `getH` method should have a return type of `int`.
- (E) The `getH` method should have a return type of `void`.

13. Consider the following class definition. The method `appendIt` is intended to take the string passed as a parameter and append it to the instance variable `str`. For example, if `str` contains “week”, the call `appendIt("end")` should set `str` to “weekend”. The method does not work as intended.

```
public class StringThing {
    private String str;
    public StringThing(String s) {
        str = s;
    }
    public void appendIt(String s) {
        str + s;
    }
}
```

Which of the following changes should be made so that the `appendIt` method works as intended?

- (A) The `appendIt` method header should be declared to have a return type `String`.
- (B) The body of `appendIt` should be changed to:
`str = s + str;`
- (C) The body of `appendIt` should be changed to:
`str = str + s;`
- (D) The body of `appendIt` should be changed to:
`return s + str;`
- (E) The body of `appendIt` should be changed to:
`return str + s;`

Writing Classes MCQ, Part A

14. Consider the class definition. The method `levelUp` is intended to increase a `Superhero` object's `strength` attribute by the parameter `amount`. The method does not work as intended.

```
public class Superhero {
    private String name;
    private String secretIdentity;
    private int strength;

    public Superhero(String realName,
                     String codeName) {
        name = realName;
        secretIdentity = codeName;
        strength = 5;
    }

    public int levelUp(int amount) {
        strength += amount;
    }
}
```

Which of the following changes should be made so that the `levelUp` method works as intended?

- (A) The `levelUp` method header should be declared to have a return type `void`.
- (B) In the `levelUp` method header, `amount` should be declared to be type `void`.
- (C) The body of `levelUp` should be changed to:
`strength + amount;`
- (D) The body of `levelUp` should be changed to:
`return strength + amount;`
- (E) The body of `levelUp` should be changed to:
`return amount;`

15. Consider the following class definition, which represents two scores using the instance variables `score1` and `score2`. The method `reset` is intended to set to 0 any score that is less than `threshold`. The method does not work as intended.

```
public class TestClass {
    private int score1;
    private int score2;

    public TestClass(
        int num1, int num2) {
        score1 = num1;
        score2 = num2;
    }

    public void reset(int threshold) {
        if(score1<threshold) {
            score1 = 0;
        } else if (score2<threshold) {
            score2 = 0;
        }
    }
}
```

Which of the following changes can be made so that the `reset` method works as intended?

- (A) In both conditions in the `reset` method, change `<` to `>`.
- (B) In both conditions in the `reset` method, change `<` to `<=`.
- (C) In the `reset` method, for the first `if` statement change `score1` to `num1` and for the second `if` statement change `score2` to `num2`.
- (D) In the `reset` method, change the `else if` to `if`.
- (E) In the `reset` method, change the `else if` to `else`.