

AP Computer Science A - Unit 3 Progress Check: MCQ Part B

Question 1

Consider the following class definition.

```
public class MyClass
{
    private int x;

    public MyClass(int start)
    {
        x = start;
    }

    public void updateX(int amount)
    {
        x += amount;
    }
}
```

Suppose that `tester` is a properly instantiated reference to a `MyClass` object and `num` is an `int` value. Which of the following best describes the conditions under which the value of the instance variable `x` is unchanged as a result of the call `tester.updateX(num)`?

- (A) When `num` is 0
 - (B) When `num` is 1
 - (C) When `num` is positive
 - (D) When `num` is negative
-

Question 2

Consider the following class definition.

```
public class XYPoint
{
    private int x;
    private int y;

    public XYPoint(int xVal, int yVal)
    {
        x = xVal;
        y = yVal;
    }

    public String getPoint()
    {
        /* missing code */
    }
}
```

The following code segment appears in a class other than `XYPoint`.

```
XYPoint p1 = new XYPoint(3, -2);
XYPoint p2 = new XYPoint(4, 0);
System.out.println(p1.getPoint());
System.out.println(p2.getPoint());
```

This code segment is intended to produce the following output:

```
(3,-2)
(4,0)
```

Which of the following statements can be used to replace `/* missing code */` so that this code segment produces the intended output?

- (A) `System.out.println("(x,y)");`
 - (B) `System.out.println("(+x+", "+y+")");`
 - (C) `return "(x,y)";`
 - (D) `return "(" + x + ", " + y + ")";`
-

Question 3

Consider the following class definition.

```
public class Fraction
{
    private int numerator;
    private int denominator;

    public Fraction(int n, int d)
    {
        numerator = n;
        denominator = d;
    }

    public double getDecimal()
    {
        return (double) numerator / denominator;
    }
}
```

Suppose that `fr` is a properly instantiated reference to a `Fraction` object. Which of the following best describes the conditions under which the call `fr.getDecimal()` will fail to return a value?

- (A) When the instance variables `numerator` and `denominator` have the same value
 - (B) When the instance variables `numerator` and `denominator` have different values
 - (C) When the instance variable `denominator` is equal to 0
 - (D) When the instance variable `denominator` is negative
-

Question 4

Consider the following method.

```
public String mystery(String message, int x)
{
    message = message.substring(x);
    return message;
}
```

The following code segment appears in a method in the same class as `mystery`.

```
String str1 = "abcde";
String str2 = mystery(str1, 3);
```

Which of the following best describes the behavior of this code segment?

- (A) `str1` is modified by the call to `mystery`; `str1` and `str2` are equal in value after executing the code segment.
 - (B) `str1` is modified by the call to `mystery`; `str1` and `str2` have differing values after executing the code segment.
 - (C) `str1` is not modified by the call to `mystery`; `str1` and `str2` are equal in value after executing the code segment.
 - (D) `str1` is not modified by the call to `mystery`; `str1` and `str2` have differing values after executing the code segment.
-

Question 5

Consider the following class definition.

```
public class Location
{
    private int row, col;

    public Location()
    { /* implementation not shown */ }
    /* There may be other constructors and methods not shown. */
}
```

The following method is intended to create a copy of a `Location` object.

```
public Location clone(Location loc)
{
    Location loc1 = new Location();
    loc1.row = loc.row;
    loc1.col = loc.col;
    return loc1;
}
```

Which of the following best describes the conditions under which the method works as intended?

- (A) The method will run correctly only if it is defined in a class other than `Location`.
 - (B) The method will run correctly only if it is defined inside the `Location` class.
 - (C) The method will always run correctly.
 - (D) The method will never run correctly.
-

Question 6

Consider the following class definition.

```
public class Item
{
    // maintains the price of an item
    private int price;

    public int getPrice()
    {
        return price;
    }
    /* There may be instance variables, constructors, and other methods not shown. */
}
```

The following method appears in a class other than `Item`. The method is intended to calculate and return the sales tax of the item specified by the parameter `myItem`. The sales tax will be calculated as the cost of the item times the value of the parameter `taxRate`.

```
public double getTax(Item myItem, double taxRate)
{
    double cost = /* missing code */;
    return cost * taxRate;
}
```

Which of the following expressions can be used to replace `/* missing code */` so that this method works as intended?

- (A) `Item.price`
 - (B) `myItem.price`
 - (C) `Item.getPrice()`
 - (D) `myItem.getPrice()`
-

Question 7

Consider the following `StringFinder` class.

```
public class StringFinder
{
    private String fullString;
    private static String target = "x";

    public StringFinder(String fs)
    {
        fullString = fs;
    }

    public static void updateTarget(String newTarget)
    {
        target = newTarget;
    }

    public int targetLocation()
    {

```

```

    {
        int temp = fullString.indexOf(target);
        target = "z";
        return temp;
    }
}

```

The following code segment appears in a class other than `StringFinder`.

```

StringFinder music = new StringFinder("jazz");
StringFinder fire = new StringFinder("blaze");
StringFinder animal = new StringFinder("zebra");
int musicLoc = music.targetLocation();
StringFinder.updateTarget("a");
int fireLoc = fire.targetLocation();
int animalLoc = animal.targetLocation();
System.out.println(musicLoc + " " + fireLoc + " " + animalLoc);

```

What is printed as a result of executing this code segment?

- (A) -1 -1 -1
 - (B) -1 2 0
 - (C) -1 2 4
 - (D) 0 2 4
-

Question 8

Consider the following class definition.

```

public class Widget
{
    private int number;
    private static String word = "start";

    public Widget()
    { /* implementation not shown */ }
}

```

The following code segment appears in a class other than `Widget`.

```
int result = Widget.doSomething();
```

Which of the following implementations of `doSomething` will allow this code segment to run without error when added to the `Widget` class?

- (A)

```

public int doSomething()
{
    return number;
}

```

- (B)

```

public int doSomething()
{

```

```

    return word.length();
}

• (C)

public static int doSomething()
{
    return number;
}

• (D)

public static int doSomething()
{
    return word.length();
}

```

Question 9

Consider the following class declaration.

```

public class TestObject
{
    private double var1;
    private static int var2 = 0;

    public TestObject(double p)
    {
        var1 = p;
        var2++;
    }

    public void printTestObject()
    {
        System.out.println(var1 + "," + var2);
    }
}

```

The following code segment appears in a class other than `TestObject`. Assume that no other `TestObject` objects have been created.

```

TestObject obj1 = new TestObject(2.5);
TestObject obj2 = new TestObject(10.2);
obj1.printTestObject();

```

What is printed as a result of executing this code segment?

- (A) 2.5,0
 - (B) 2.5, 1
 - (C) 2.5, 2
 - (D) 10.2, 2
-

Question 10

Consider the following class definition.

```
public class ComputeObject
{
    private int limit;
    private int val;

    public ComputeObject()
    {
        limit = 7;
        val = 10;
    }

    public int sumProd(int limit)
    {
        int total = 0;
        for (int val = 0; val < limit; val++)
        {
            total += val;
        }
        total *= val;
        return total;
    }
}
```

The following code segment appears in a class other than `ComputeObject`.

```
ComputeObject s = new ComputeObject();
System.out.println(s.sumProd(5));
```

Which of the following best describes the behavior of this code segment?

- (A) The `sumProd` method accumulates the sum of the integers 0 through 4, multiplies the sum by 5, and returns the value 50, which is then printed.
 - (B) The `sumProd` method accumulates the sum of the integers 0 through 4, multiplies the sum by 10, and returns the value 100, which is then printed.
 - (C) The `sumProd` method accumulates the sum of the integers 0 through 6, multiplies the sum by 7, and returns the value 147, which is then printed.
 - (D) The `sumProd` method accumulates the sum of the integers 0 through 6, multiplies the sum by 10, and returns the value 210, which is then printed.
-

Question 11

Consider the following class definition.

```
public class Stars
{
    private int size;
    private int width;

    public Stars(int n, int w)
    {
```

```

        size = n;
        width = w;
    }

    public void draw(int size)
    {
        for (int i = 1; i <= size; i++)
        {
            line(i, width * 1);
        }
    }

    public void line(int n, int width)
    {
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < width; j++)
            {
                System.out.print("*");
            }
        }
        System.out.println();
    }
}

```

The following code segment appears in a class other than `Stars`.

```

Stars s = new Stars(3, 2);
s.draw(5);

```

What is printed as a result of executing this code segment?

- (A)

```

*
**
***

```

- (B)

```

**
*****

```

- (C)

```

*
**
***
****

```

- (D)

```

**
****
*****
*****
*****

```

Question 12

Consider the following class, which models a bank account. The `deposit` method is intended to update the account balance by a given amount; however, it does not work as intended.

```
public class BankAccount
{
    private String accountOwnerName;
    private double balance;
    private int accountNumber;

    public BankAccount(String name, double initialBalance, int acctNum)
    {
        accountOwnerName = name;
        balance = initialBalance;
        accountNumber = acctNum;
    }

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

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

- (A) The `deposit` method must have a return statement.
 - (B) In the `deposit` method, the variable `balance` should be replaced by the variable `initialBalance`.
 - (C) In the `deposit` method, the variable `balance` is declared as a local variable and is different from the instance variable `balance`.
 - (D) The variable `balance` must be passed to the `deposit` method.
-

Question 13

Consider the following class definition.

```
public class Player
{
    private int base;
    private int bonus;

    public Player()
    { /* implementation not shown */ }

    public int getTotal()
    {
        return base + bonus;
    }

    public boolean hasHigherScore (Player other)
```

```

    {
        /* missing code */
    }
}

```

The `Player` class contains a `getTotal` method, which returns the total score for the player. The class also contains a `hasHigherScore` method, which has a `Player` object as a parameter. The `hasHigherScore` method, when called on a given `Player` object, is intended to return `true` if the total score of the given `Player` is greater than the total score of the `Player` specified by the parameter. It is intended to return `false` otherwise. For example, suppose `p1` and `p2` are valid `Player` objects. If `p1` has a higher total score than `p2`, `p1.hasHigherScore(p2)` should return `true`.

Which of the following can be used to replace `/* missing code */` so that the `hasHigherScore` method works as intended?

- (A) `return this > other;`
 - (B) `return getTotal(this) > getTotal(other);`
 - (C) `return this.getTotal() > getTotal();`
 - (D) `return this.getTotal() > other.getTotal();`
-

Question 14

Consider the following class declaration.

```

public class Thing
{
    private int val;

    public Thing(int v)
    {
        val = v;
    }

    public int getVal()
    {
        return val;
    }

    public String mystery(Thing other)
    {
        if (this == other)
        {
            return "yes";
        }
        else if (this.val == other.getVal())
        {
            return "maybe";
        }
        else return "no";
    }
}

```

The following code segment appears in a class other than `Thing`.

```

Thing apple = new Thing(5);
Thing banana = new Thing(5);
System.out.println(apple.mystery(banana));
System.out.println(banana.mystery(banana));

```

What, if anything, is printed as a result of executing this code segment?

- (A)

yes
yes

- (B)

maybe
yes

- (C)

maybe
maybe

- (D) Nothing is printed because the **Thing** class does not compile.

Question 15

Consider the following class definition.

```

public class Friend
{
    private String name; // Line 3

    public Friend(String name)
    {
        name = name; // Line 7
    }

    public String getName()
    {
        return name; // Line 12
    }
}

```

The following code segment appears in a class other than **Friend**. It is intended to print the string “Jessie” but does not work as intended because of an error in the **Friend** class.

```

Friend bestie = new Friend("Jessie");
System.out.println(bestie.getName());

```

Which of the following changes can be made to the **Friend** class so that this code segment works as intended?

- (A) Changing line 3 to `private String this.name;`
- (B) Changing line 7 to `this.name = name;`
- (C) Changing line 7 to `name = this.name;`
- (D) Changing line 12 to `return this.name;`