Exam 810: Sample Questions

1. Given the code fragment:

```
public class App {
    void calcBill() {
        // Line n1
        new Invoice().print();
    }
}
```

Which code fragment can be inserted at `Line n1` to enable the class compile?

A)
```
private class Invoice {
    void print() {System.out.println("Invoice Printed");}
  }
```

B)
```
public class Invoice {
    void print() {System.out.println("Invoice Printed");}
  }
```

C)
```
class Invoice {
    void print() {System.out.println("Invoice Printed");}
  }
```

D)
```
protected class Invoice {
    void print() {System.out.println("Invoice Printed");}
  }
```

2. Given:

```
public interface MyInt {
    public void method1() {
        System.out.println("method1");
    }
    public default void method2() {
        System.out.println("method2");
    }
    public static void method3() {
        System.out.println("method3");
    }
    public abstract void method4();
}
```

Which statement is true?

A) Only `method4()` compiles
B) Only `method2()` and `method4()` compiles.
C) Only `method2()`, `method3()`, and `method4()` compiles.
D) MyInt.java compiles.

3. Given the code fragment:

```
public static void main(String[] args) {
    List<String> courses = Arrays.asList("Java", "Oracle", "JSF", "EJB");
    // Line n1
    System.out.println(count);
}
```

Which code fragment can be inserted at `Line n1` to enable the code to print `2`?

A) `int count = courses.stream().filter(s -> s.startsWith("J")).count();`
B) `long count = courses.stream().filter(s -> s.startsWith("J")).count();`
C) `int count = courses.filter(s -> s.startsWith("J")).stream().count();`
D) `long count = courses.filter(s -> s.startsWith("J")).stream().count();`

4. Given the code fragment:

```
public class App{
    public static void main(String[] args) {
        String[] fruits = {"banana", "apple", "pears", "grapes"};
        Arrays.sort(fruits, (a, b) -> a.compareTo(b));
        for (String s : fruits) {
            System.out.print(" "+s);
        }
    }
}
```

What is the result?

A) `apple banana grapes pears`
B) `pears grapes banana apple`
C) `banana apple pears grapes`
D) Compilation fails.

5. Given the code fragment:

```
LocalDate date1 = LocalDate.of(2016, Month.JANUARY, 1);
LocalDateTime date2 = LocalDateTime.of(2017, Month.JUNE, 1, 1, 1);
Period p = Period.between(date1, date2);
System.out.print(p.getYears() + ":" + p.getMonths() + ":" + p.getDays());
```

What is the result?

A) `1:5:0`
B) `1:6:0`
C) `0:0:0`
D) Compilation fails.

6. Given that `/report/jun.txt` and `report/data/jundata.txt` files are accessible and given the code fragment:

```
public static void main(String[] args) {
    try (Stream<Path> st1 = Files.find(Paths.get("/report"), 2, (p, a) ->
p.toString().endsWith("txt"));
        Stream<Path> st2 = Files.walk(Paths.get("/report"), 2);) {
            st1.forEach(s -> System.out.println("Found: " + s));

            st2.filter(s -> s.toString()
               .endsWith("txt"))
               .forEach(s -> System.out.println("Walked: " + s));

        } catch (IOException ioe) {
            System.out.println("Exception");
        }
}
```

What is the result?

A)
```
Found: \report\data\jundata.txt
Found: \report\jun.txt
Walked: \report\data\jundata.txt
Walked: \report\jun.txt
```

B)
```
Found: \report\data\jundata.txt
Found: \report\jun.txt
Walked: \report\data\jundata.txt
Walked: \report
Walked: \report\jun.txt
Walked: \report\data\
```

C)
```
Found: \report\jun.txt
Walked: \report\data\jundata.txt
Walked: \report\jun.txt
```

D)
```
Found: \report\jun.txt
Walked: \report
Walked: \report\jun.txt
Walked: \report\data\
Walked: \report\data\jundata.txt
```

7. Given the code fragment:

```
public static void main(String[] args) {
    Stream<Integer> nums = Stream.of(1, 2, 3, 4, 5);
    nums.filter(n -> n % 2 == 1);
    nums.forEach(p -> System.out.print(p));
}
```

What is the result?

A) `135`
B) `12345`
C) Compilation fails.

D) An exception is thrown at runtime.

8. Given the code fragment:

```
class MyResource1 implements Closeable {

    public void close() {
        System.out.print("r1 ");
    }
}

class MyResource2 implements AutoCloseable {

    public void close() throws IOException {
        System.out.print("r2 ");
        throw new IOException();
    }
}

public class App2 {

    public static void main(String[] args) {
        try (MyResource1 r1 = new MyResource1();
                MyResource2 r2 = new MyResource2();) {
            System.out.print("try ");
        } catch (Exception e) {
            System.out.print("catch ");
            for (Throwable t : e.getSuppressed()) {
                System.out.println(t.getClass().getName());
            }
        }
    }
}
```

What is the result?

A) `try r2 r1 catch java.io.IOException`
B) `try r2 r1 catch`
C) `try r1 r2 catch`
D) Compilation fails.

Answers:

1. C
2. C
3. B
4. A
5. D
6. A
7. D
8. B