1. switch(season){
    case Season.WINTER:
    …
    break;
    case Season.SPRING:
    …
    break;
    case Season.SUMMER:
    …
    break;
    case Season.FALL:
    …
   }

   Which assertion clause do you add to the default clause of the switch block in the sample code above in order to catch the condition, when season is not one of the given four values?

   a. default:
      assert true: season;
   [✓ b.] default:
      assert false: season;
   c. default:
      throw new AssertionException(season);
   d. default:
      assertion true: season;
   e. default:
      assertion false: season;

2.

```
try {
  // some code here that throws IOException
}
catch (IOException ex){
      System.out.println(“Line 1: Sample Error!”);
      throw ex;
}
finally {
      System.out.println(“Line 2: Example Error!”);
}
```

What is the output of the sample code above?

[✓ a.] Line 1: Sample Error!
Line 2: Example Error!
b. Line 2: Example Error!
c. Line 2: Example Error!
Line 1: Sample Error!
d. Line 1: Sample Error!
e. Sample Error and Example Error!

3.

```
int a=5;
int b=0;
try {
      int c = a/b;
}
catch (FileNotFoundException a| IOException b|
ArithmeticException c)
{
     System.out.println(c.getMessage(“Cannot do this!”));
     System.out.println(b.getMessage(“Nope!”));
     System.out.println(a.getMessage(“Wrong Again!”));
}
```

What is the outcome of the sample code above?

a. Wrong Again!
[✓ b.] Cannot do this!
c. Nope!
d. Nope!
Wrong Again!
e. Cannot do this!
Nope!

4.
```
public double SquareRoot(double value) throws
ArithmeticException {
 if (value >= 0) return Math.sqrt(value);
       else throw new ArithmeticException();
 }
 public double func(int x) {
       double y = (double) x;
 try {
       y = SquareRoot( y );
 } catch(ArithmeticException e) {
       y = 0;
 } finally {
       --y;
 }
 return y;
 }
```

Referring to the sample code above, what value is returned when you invoke method func(4)?

a. -2.0
b. -1.0
c. 0
[✓ d.] 1.0
e. 2.0

5. Sample Code

```
try{
     //code throwing exception
}catch(IllegalArgumentException iae){
     //code throwing exception
}catch(Exception e){

     //code throwing exception

}catch(DataFormatException dfe){
}
```

How do you rewrite the sample code above to reduce size?

```
a) try{
     //code throwing exception
     }
     catch(IllegalArgumentException iae, Exception e,
DataFormatException dfe){
     }
```

[✓ b)]

```
b) try{

     //code throwing exception

     }catch(IllegalArgumentException iae | Exception e |
     DataFormatException dfe) {
     }
```

```
c) try{
        //code throwing exception
        }
        catch(IllegalArgumentException iae),
             (Exception e),
             (DataFormatException dfe){
        }
```

```
d) try{
        //code throwing exception
        }
        catch(IllegalArgumentException iae; Exception e;
        DataFormatException dfe){
        }
```

```
e) try{
        //code throwing exception
        }
        catch("IllegalArgumentException iae" - "Exception e" -
        "DataFormatException dfe"){
        }
```

6. Sample Code

```
try{
     //code
}catch (Exception ex){
if( ex instanceof SubSubException){
     //code
}else if(ex instanceof SubException){
     //code
}else{
     //code
}
}
```

Why does the sample code above produce an error?

[✓ a.] The code is not structured properly for a try catch exception.
b. The second if condition will never be evaluated, and its body is effectively unreachable code.
c. The code is not valid java code.
d. The catch block cannot contain if-then-else statements. The if-then-else statement needs to be run first.
e. The code is ordered in a way that will not cycle through all the proper checks.

7.

```
Line 1 try
Line 2 {
Line 3    //some code
Line 4 }
Line 5 catch {Exception e}
Line 6 {
Line 7    if (e instanceof FooException)
Line 8      throw e;
Line 9 }
```

How do you fix the sample code above for it to throw an exception?

[✓ a.] Change Line 5 to catch (Exception e).
b. Change Line 8 to throw catch (Exception e).
c. Insert throw e after Line 5.
d. Insert try {//some code} after Line 5.
e. Change Line 7 to if {e instanceof FooException}.

8.
```
public class TestEx {
static class Ex1 extends Exception {}
static class Ex2 extends Ex1 {}
static class Ex3 extends Exception {}
static void method1() throws Ex1,Ex2,Ex3 {
throw  new Ex2(); }
public static void main(String args[]) {
try {
method1();
} catch (Ex3 e) {
System.out.print("C");
} catch (Ex2 e) {
System.out.print("B");
} catch (Ex1 e) {
System.out.print("A");
} catch (Exception e) {
System.out.print("D");
} finally {
System.out.println("F");
}
}
}
```
What is the output of the sample code above?

a. AF
b. BAF
[✓ c. BF]
d. CF
e. DF

9. Sample Code

```
public double SquareRoot(double value) throws
ArithmeticException
{
      if (value >= 0)
            return Math.sqrt(value);
      else
           throw new ArithmeticException();
}
public double func(int x) {
       double y = (double) x;
      y *= -9.0;
 try {
       y = SquareRoot( y );
 } catch(ArithmeticException e) {
      y /= 3;
} finally {
      y += 10;
 }
      return y;
 }
```

Referring to the sample code above, what value is returned when you invoke method func(9)?

a. -27
[✓ b. -17]
c. 7
d. 9
e. NaN