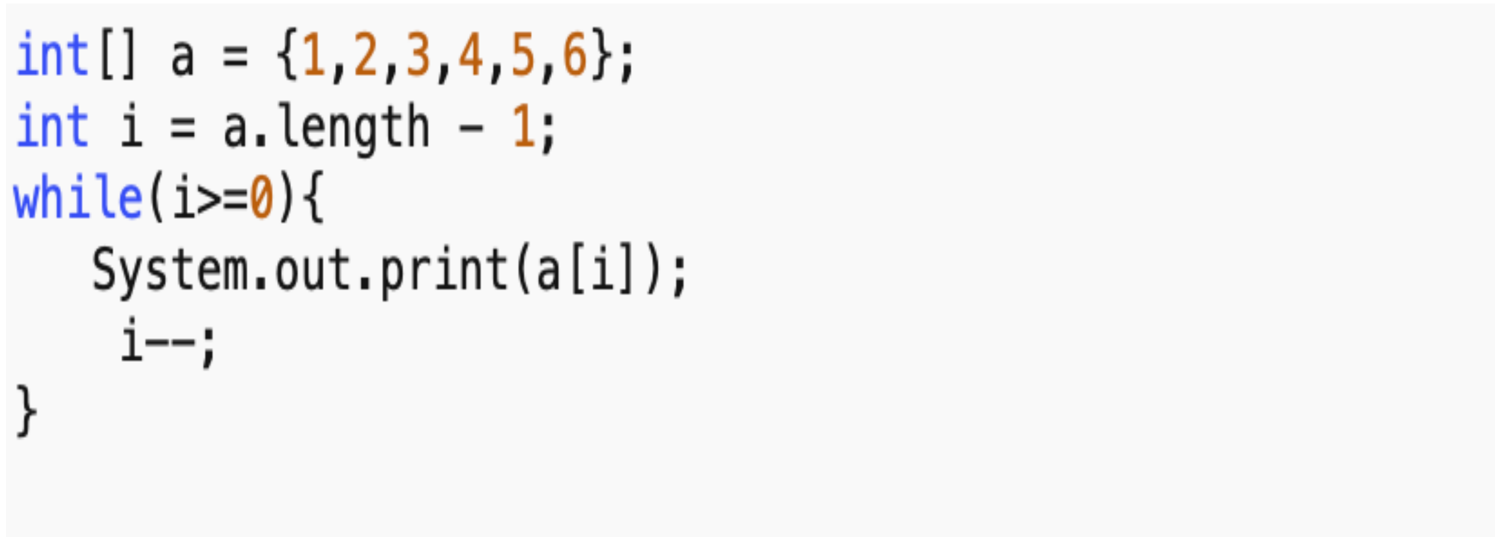
QUIZ 2

Covers: Whole Semester

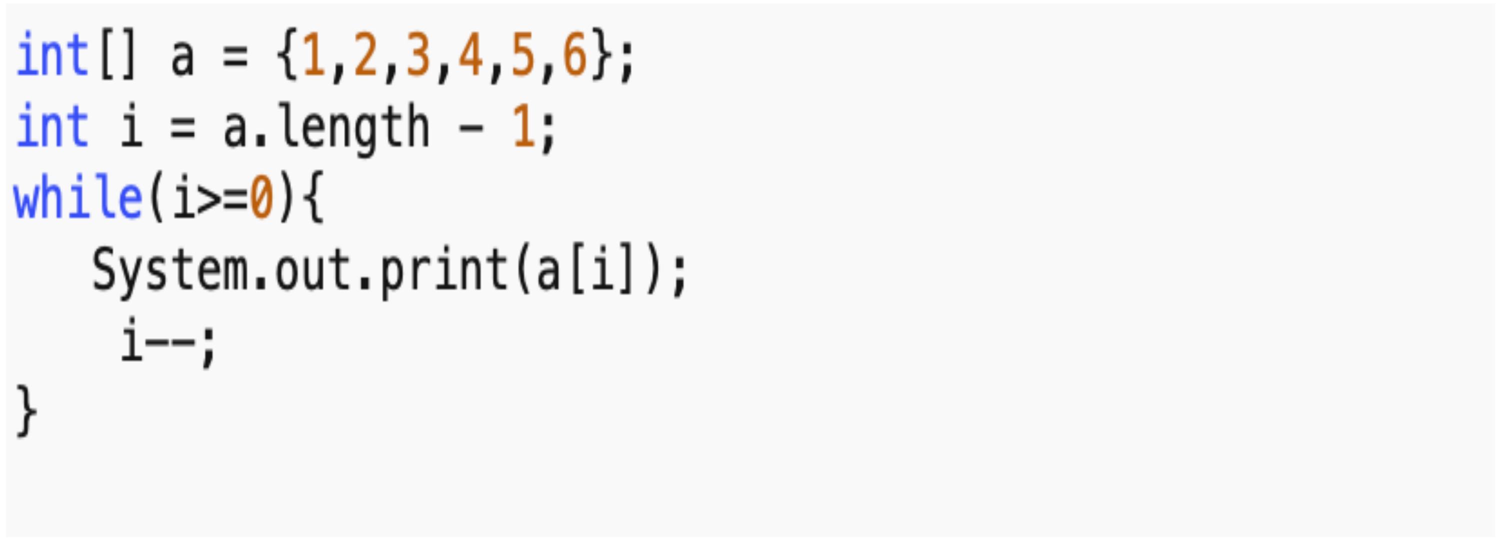
1. What would the following code display?



```java
int[] a = {1,2,3,4,5,6};
int i = a.length - 1;
while(i>=0){
   System.out.print(a[i]);
    i--;
}
```

a) 123456
b) Exception at runtime
c) 654321
d) Nothing
e) 65432
f) 12345

1. What would the following code display?



```
int[] a = {1,2,3,4,5,6};
int i = a.length - 1;
while(i>=0){
   System.out.print(a[i]);
    i--;
}
```

a) 123456
b) Exception at runtime
c) [✓ 654321]
d) Nothing
e) 65432
f) 12345

Pretty obvious once you have understood that it's looping in decreasing order and when you have the boundaries right.

2. A and E are classes, B and D are interfaces. Which of the following statements are correct?

a) interface F implements B,D { }
b) interface F implements D { }
c) interface F extends D { }
d) interface F extends E { }
e) interface F extends B, D { }

2. A and E are classes, B and D are interfaces. Which of the following statements are correct?

a) interface F implements B,D { }
b) interface F implements D { }
c) interface F extends D { }
d) interface F extends E { }
e) interface F extends B, D { }

"To implement" means to provide the code for a method specified in an interface. An interface can only extend, and extend **one** other interface (no multiple inheritance in Java)

3. What is inheritance?

a) It is the process where one object acquires properties of another
b) Inheritance is the ability of an object to take on many forms
c) Inheritance is a technique to define different methods of the same type
d) None of the above

3. What is inheritance?

a) It is the process where one object acquires properties of another

b) Inheritance is the ability of an object to take on many forms

c) Inheritance is a technique to define different methods of the same type

d) None of the above

b refers to "polymorphism" and c (although it says nothing about method names) *might* refer to overloading.

4. What is polymorphism

a) Polymorphism is a technique to define different objects of the same type
b) Polymorphism is the ability of an object to take on many forms
c) Polymorphism is a technique to define different methods of the same type
d) None of the above

4. What is polymorphism

a) Polymorphism is a technique to define different objects of the same type
b) Polymorphism is the ability of an object to take on many forms
c) Polymorphism is a technique to define different methods of the same type
d) None of the above

Poly = several, morph = shape (in Greek)

5. What are instance variables?

a) Instance variables are static variables within a class but outside any method

b) Instance variables are variables defined inside methods, constructors or blocks

c) Instance variables are variables within a class but outside any method

d) None of the above

5. What are instance variables?

a) Instance variables are static variables within a class but outside any method

b) Instance variables are variables defined inside methods, constructors or blocks

c) [✓ Instance variables are variables within a class but outside any method]

d) None of the above

static variables may be class variables (outside a method) or variables that you only see within a method but retain their values across calls (they aren't on the stack); b refers to *local* variables, allocated on the stack.

6. When we refer to a “member of a class” we mean:

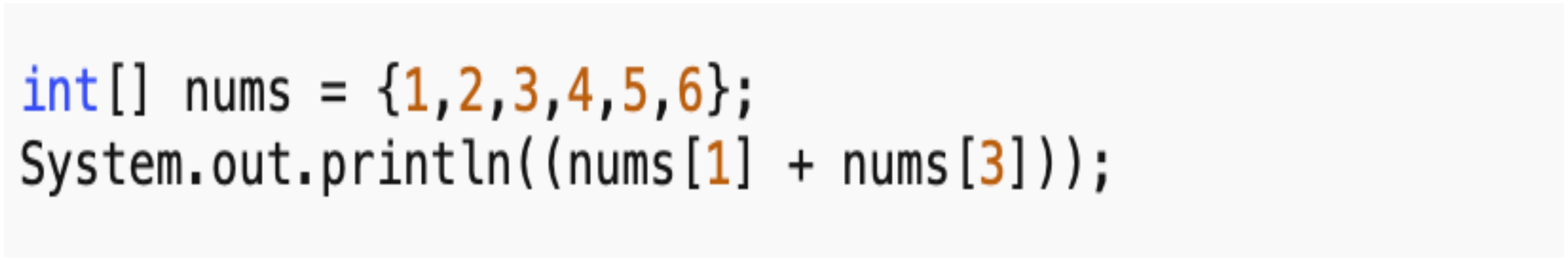
a) An attribute
b) A method
c) An attribute or method
d) A sub-class

6. When we refer to a “member of a class” we mean:

a) An attribute
b) A method
c) An attribute or method
d) A sub-class

"member" is a generic term for whatever belongs to a larger group (group member, family member ...).
Anything inside a class is a "member".

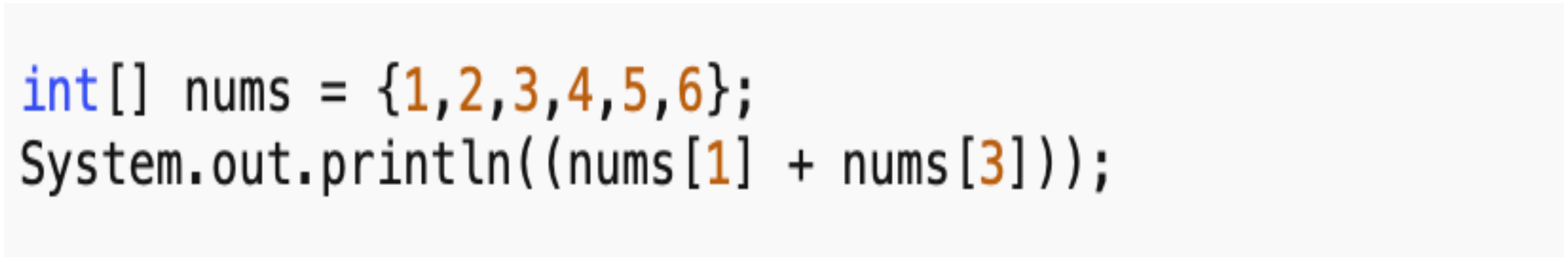
7. What would be displayed after the following code?



```
int[] nums = {1,2,3,4,5,6};
System.out.println((nums[1] + nums[3]));
```

a) 6
b) 2+4
c) 1+3
d) 4

7. What would be displayed after the following code?



```java
int[] nums = {1,2,3,4,5,6};
System.out.println((nums[1] + nums[3]));
```

a) 6

b) 2+4

c) 1+3

d) 4

Easy…

8. If classes Student, Staff, and Faculty extend the class Person, which of the following make sense?

a) Faculty[] faculties = {new Person(), new Staff(), new Student()};
b) Staff[] staff={new Person(), new Faculty(), new Student()};
c) Person[] persons={new Faculty(), new Staff(), new Student()};

8. If classes Student, Staff, and Faculty extend the class Person, which of the following make sense?

a) Faculty[] faculties = {new Person(), new Staff(), new Student()};

b) Staff[] staff={new Person(), new Faculty(), new Student()};

c) Person[] persons={new Faculty(), new Staff(), new Student()};

A reference to the parent can be assigned a reference to a child, not the reverse. Every Faculty, Student and Staff is a Person.

9. What is the output of the following program?

a) 5

b) 18

c) 3

d) Compile error

e) Runtime error

```java
class Recursion {
  int func(int n) {
    int result;
     if (n == 0) {
        result = 3;
     } else {
        result = func(n - 1);
    }
    return result;
  }
}

public class Prog {
  public static void main(String args[]) {
     Recursion obj = new Recursion() ;
     System.out.print(obj.func(5));
  }
}
```

9. What is the output of the following program?

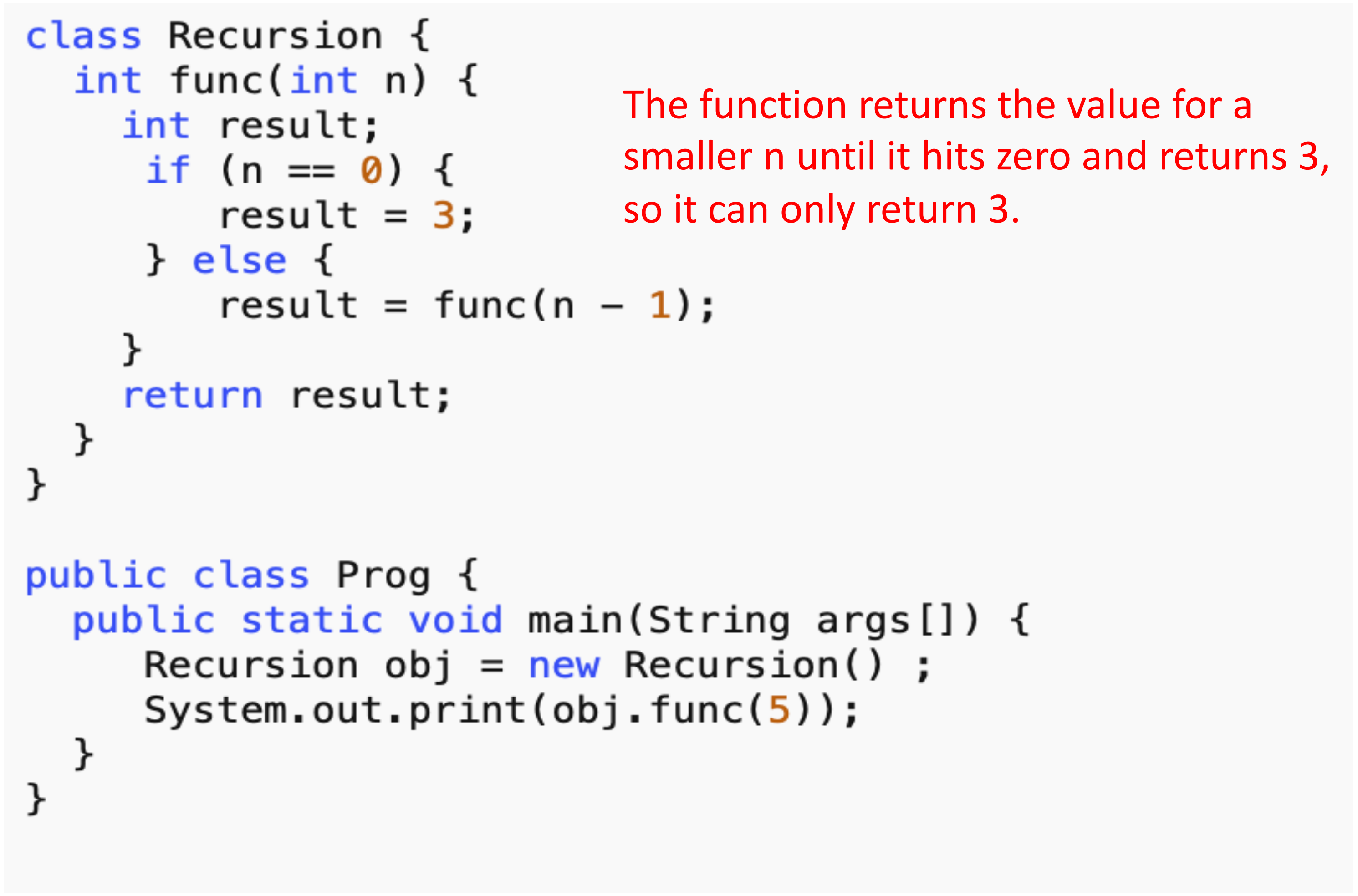
a) 5

b) 18

c) 3

d) Compile error

e) Runtime error



```
class Recursion {
  int func(int n) {
    int result;
     if (n == 0) {
        result = 3;
     } else {
        result = func(n - 1);
    }
    return result;
  }
}

public class Prog {
  public static void main(String args[]) {
     Recursion obj = new Recursion() ;
     System.out.print(obj.func(5));
  }
}
```

The function returns the value for a smaller n until it hits zero and returns 3, so it can only return 3.

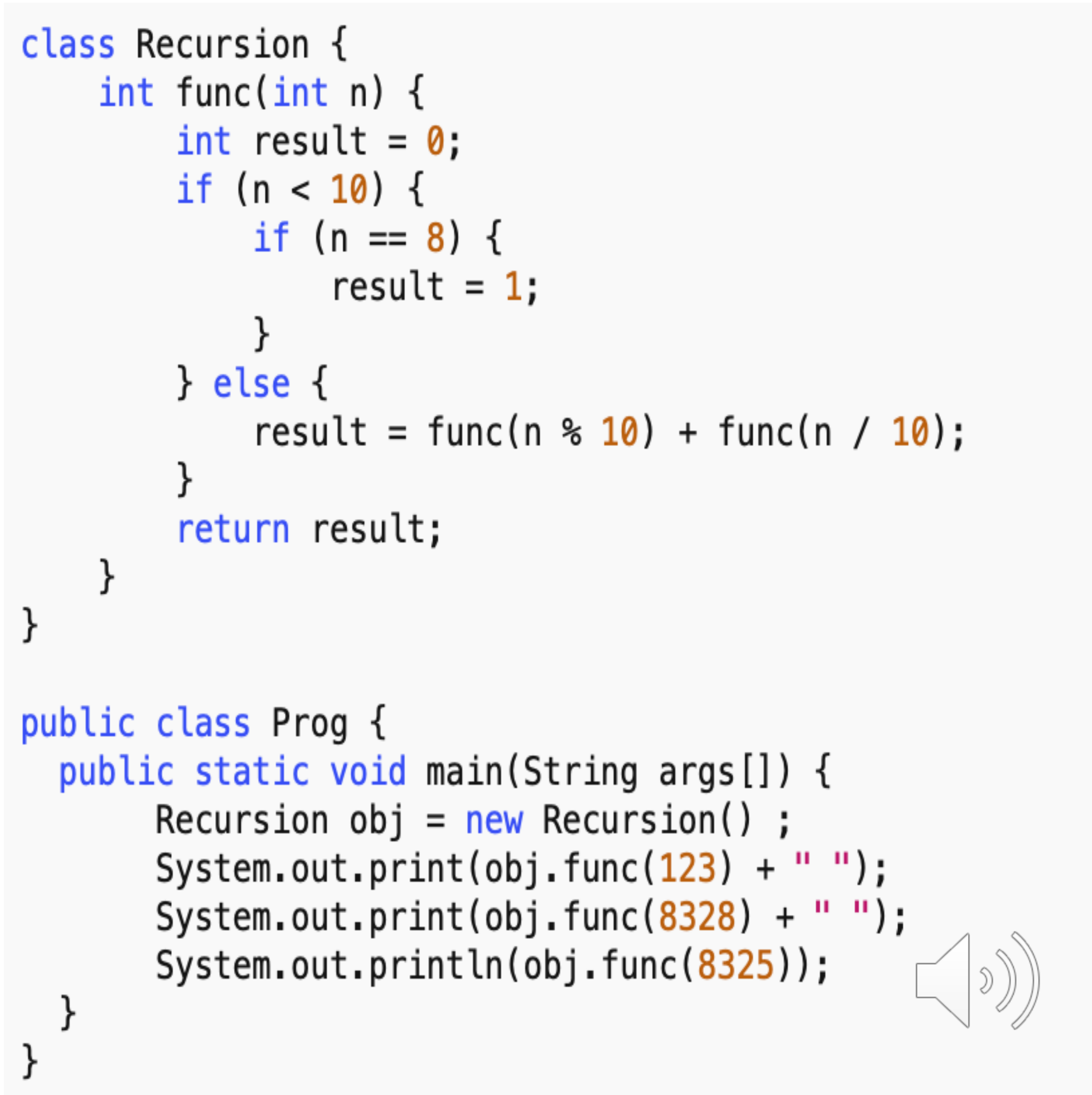
10. What is the output of the following program?

a) 0 0 0

b) 0 2 1

c) 0 1 0

d) Compile error



```java
class Recursion {
    int func(int n) {
        int result = 0;
        if (n < 10) {
            if (n == 8) {
                result = 1;
            }
        } else {
            result = func(n % 10) + func(n / 10);
        }
        return result;
    }
}

public class Prog {
  public static void main(String args[]) {
       Recursion obj = new Recursion() ;
       System.out.print(obj.func(123) + " ");
       System.out.print(obj.func(8328) + " ");
       System.out.println(obj.func(8325));
  }
}
```

10. What is the output of the following program?

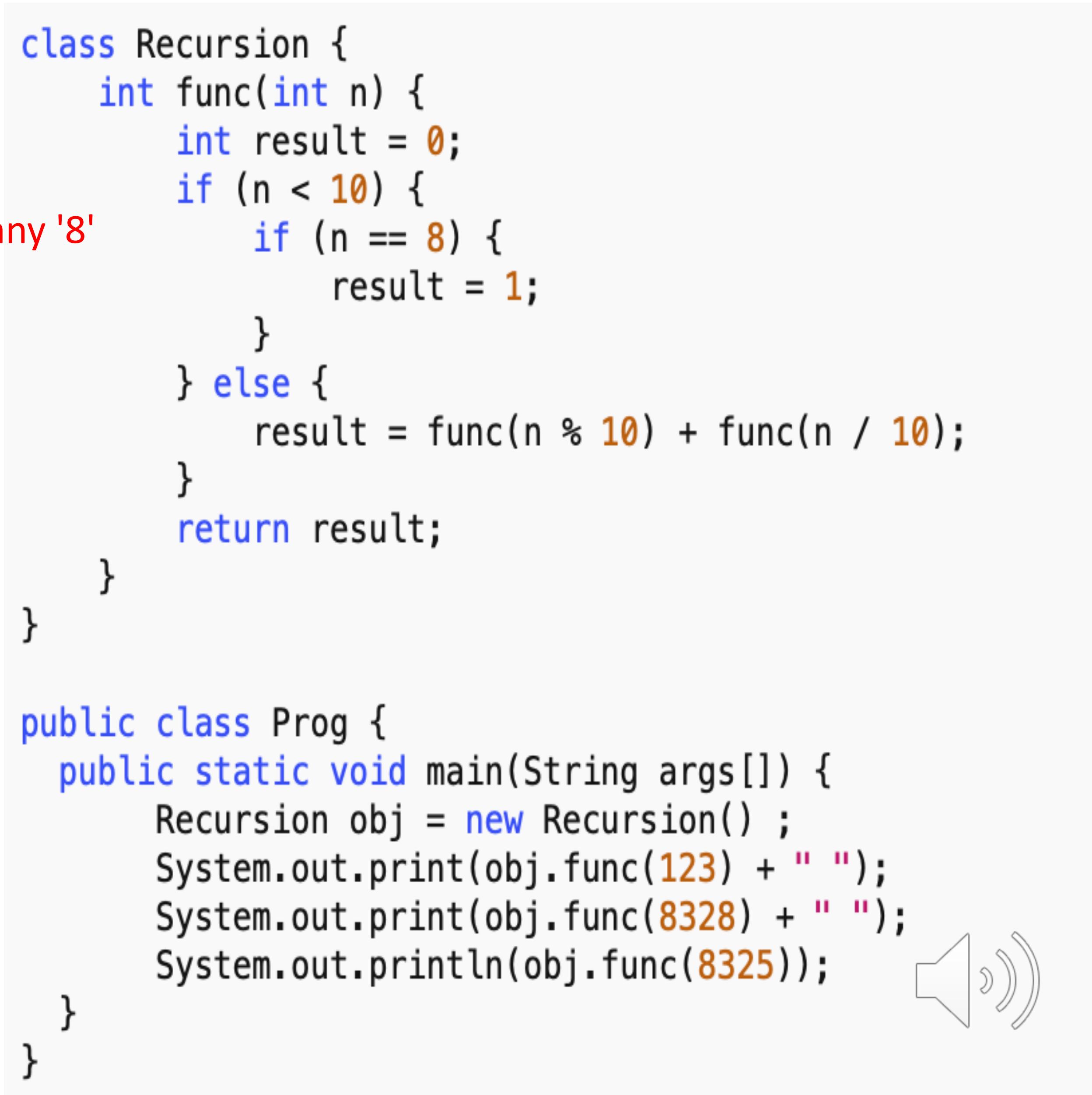
a) 0 0 0

b) [✓ 0 2 1]

c) 0 1 0

d) Compile error

The function actually counts how many '8' there are in the number.



```java
class Recursion {
    int func(int n) {
        int result = 0;
        if (n < 10) {
            if (n == 8) {
                result = 1;
            }
        } else {
            result = func(n % 10) + func(n / 10);
        }
        return result;
    }
}

public class Prog {
  public static void main(String args[]) {
        Recursion obj = new Recursion() ;
        System.out.print(obj.func(123) + " ");
        System.out.print(obj.func(8328) + " ");
        System.out.println(obj.func(8325));
  }
}
```

11. Which of the following is a valid annotation definition?

a) public @annotation MyAnnotation{ }
b) private @interface MyAnnotation{ }
c) public @interface MyAnnotation{ }
d) public @MyAnnotation{ }

11. Which of the following is a valid annotation definition?

a) public @annotation MyAnnotation{ }
b) private @interface MyAnnotation{ }
c) public @interface MyAnnotation{ }
d) public @MyAnnotation{ }

An annotation is a special interface, and there is no reason to make it private.

12. Which of the following belongs to meta-annotations?

a) @Override

b) @Retention

c) @Deprecated

d) @SuppressWarnings()

12. Which of the following belongs to meta-annotations?

a) @Override
b) @Retention
c) @Deprecated
d) @SuppressWarnings()

A meta-annotation is an annotation about annotation. The other annotations refer to the annotated code.

FYI: The @Retention annotation indicates how long annotations with the annotated type are to be retained.

Can be one of:
- CLASS: annotations are to be recorded in the class file by the compiler but need not be retained by the VM at run time.
- RUNTIME: Annotations are to be recorded in the class file by the compiler and retained by the VM at run time, so they may be read reflectively.
- SOURCE: Annotations are to be discarded by the compiler.

13. Which of the following are valid retention policy types in Java (multiple answers)?

a) SOURCE
b) CLASS
c) RUNTIME
d) CODE
e) TOOLS

13. Which of the following are valid retention policy types in Java (multiple answers)?

a) SOURCE
b) CLASS
c) RUNTIME
d) CODE
e) TOOLS

CODE and TOOLS don't exist.

14. What will the following program output?

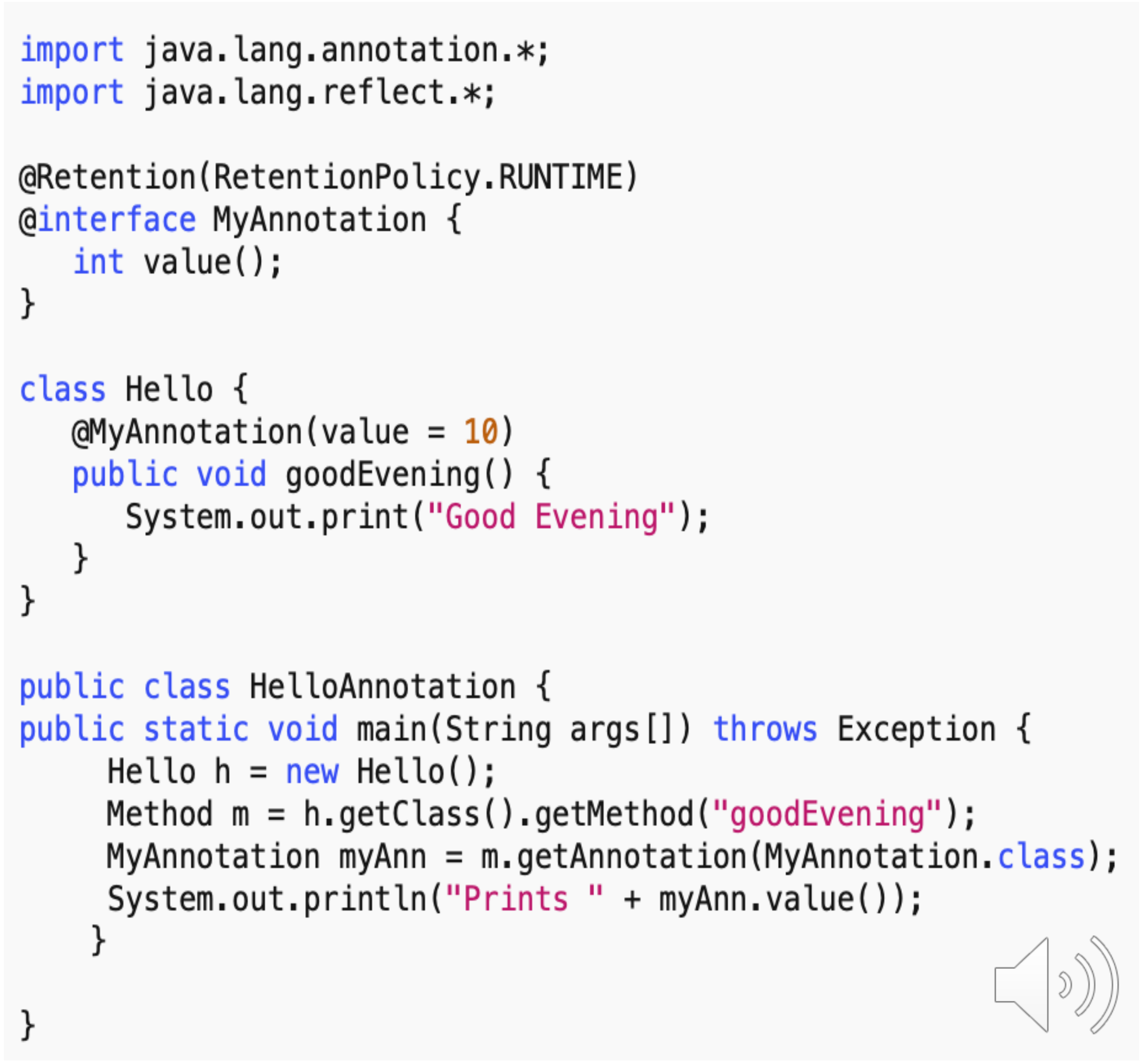
a) Prints HelloAnnotation 10

b) Good Evening Prints 10

c) Prints 10

d) Some other output

e) Output cannot be determined



```java
import java.lang.annotation.*;
import java.lang.reflect.*;

@Retention(RetentionPolicy.RUNTIME)
@interface MyAnnotation {
   int value();
}

class Hello {
   @MyAnnotation(value = 10)
   public void goodEvening() {
      System.out.print("Good Evening");
   }
}

public class HelloAnnotation {
public static void main(String args[]) throws Exception {
     Hello h = new Hello();
     Method m = h.getClass().getMethod("goodEvening");
     MyAnnotation myAnn = m.getAnnotation(MyAnnotation.class);
     System.out.println("Prints " + myAnn.value());
    }

}
```

14. What will the following program output?

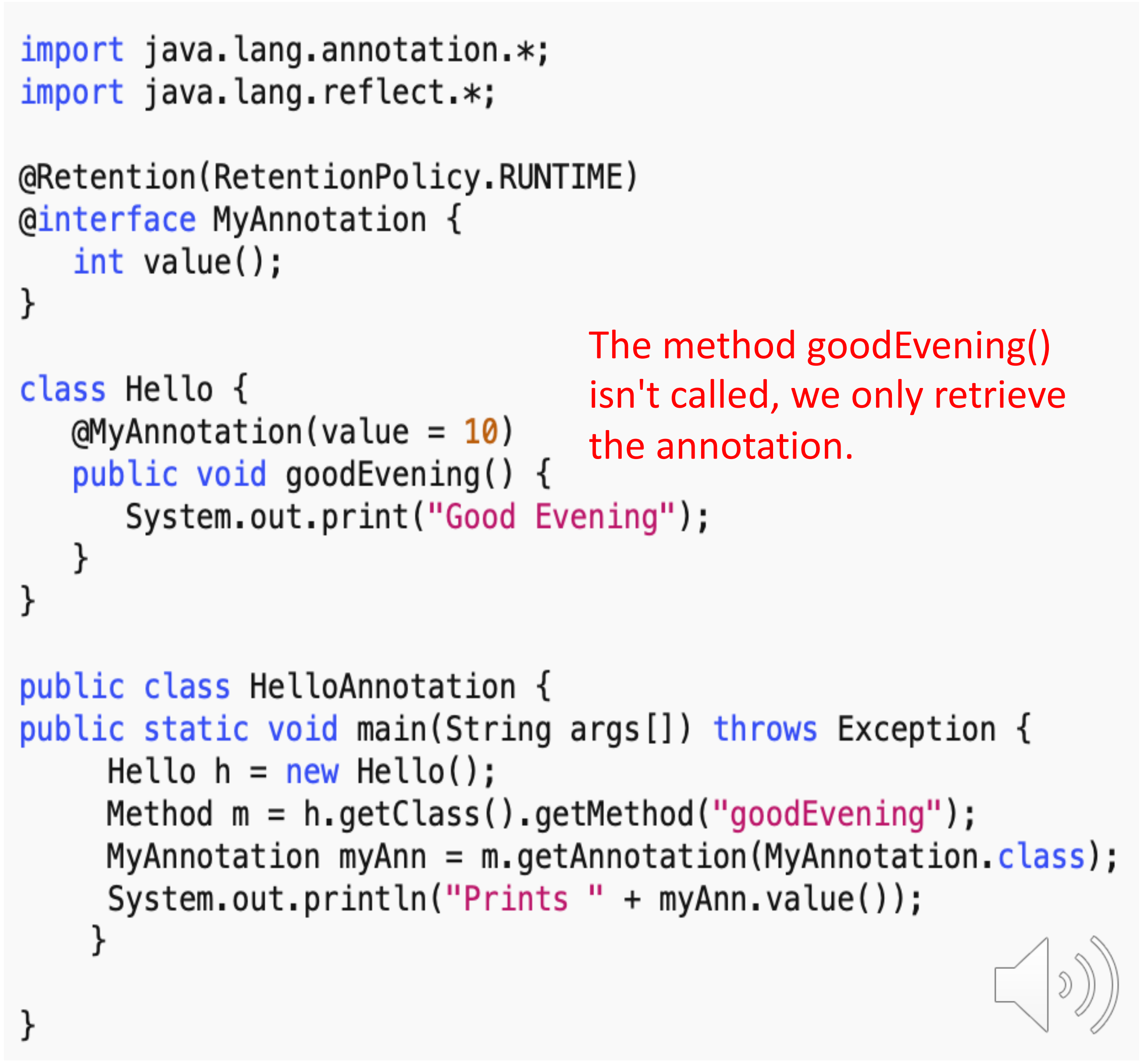
a) Prints HelloAnnotation 10

b) Good Evening Prints 10

c) [✓ Prints 10]

d) Some other output

e) Output cannot be determined



```
import java.lang.annotation.*;
import java.lang.reflect.*;

@Retention(RetentionPolicy.RUNTIME)
@interface MyAnnotation {
   int value();
}

class Hello {
   @MyAnnotation(value = 10)
   public void goodEvening() {
      System.out.print("Good Evening");
   }
}

public class HelloAnnotation {
public static void main(String args[]) throws Exception {
     Hello h = new Hello();
     Method m = h.getClass().getMethod("goodEvening");
     MyAnnotation myAnn = m.getAnnotation(MyAnnotation.class);
     System.out.println("Prints " + myAnn.value());
    }

}
```

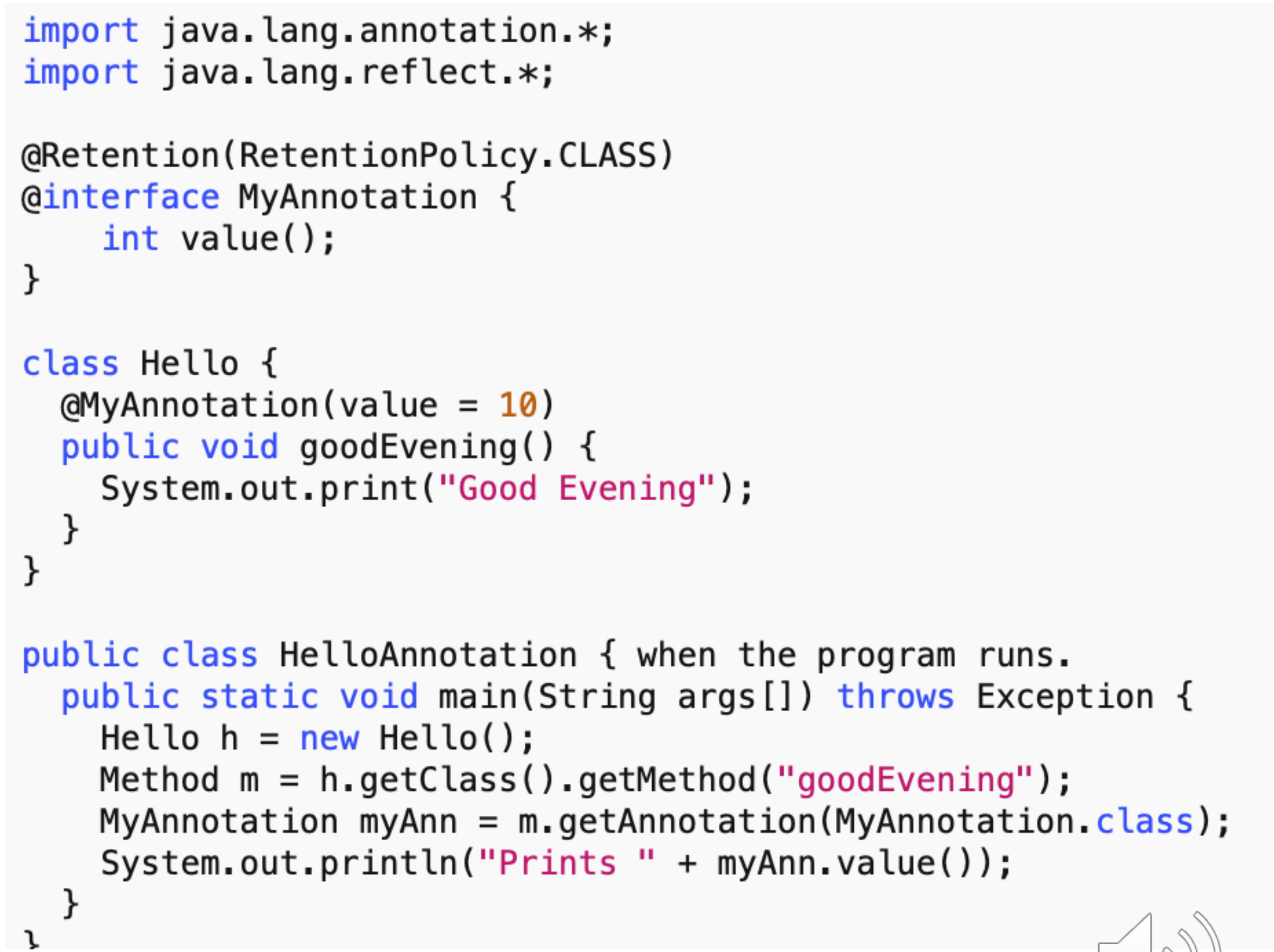
The method goodEvening() isn't called, we only retrieve the annotation.

15. What will be the output of the following program?

a) Prints HelloAnnotation 10

b) Compilation Error

c) Runtime Exception



```java
import java.lang.annotation.*;
import java.lang.reflect.*;

@Retention(RetentionPolicy.CLASS)
@interface MyAnnotation {
    int value();
}

class Hello {
  @MyAnnotation(value = 10)
  public void goodEvening() {
    System.out.print("Good Evening");
  }
}

public class HelloAnnotation { when the program runs.
  public static void main(String args[]) throws Exception {
    Hello h = new Hello();
    Method m = h.getClass().getMethod("goodEvening");
    MyAnnotation myAnn = m.getAnnotation(MyAnnotation.class);
    System.out.println("Prints " + myAnn.value());
  }
}
```

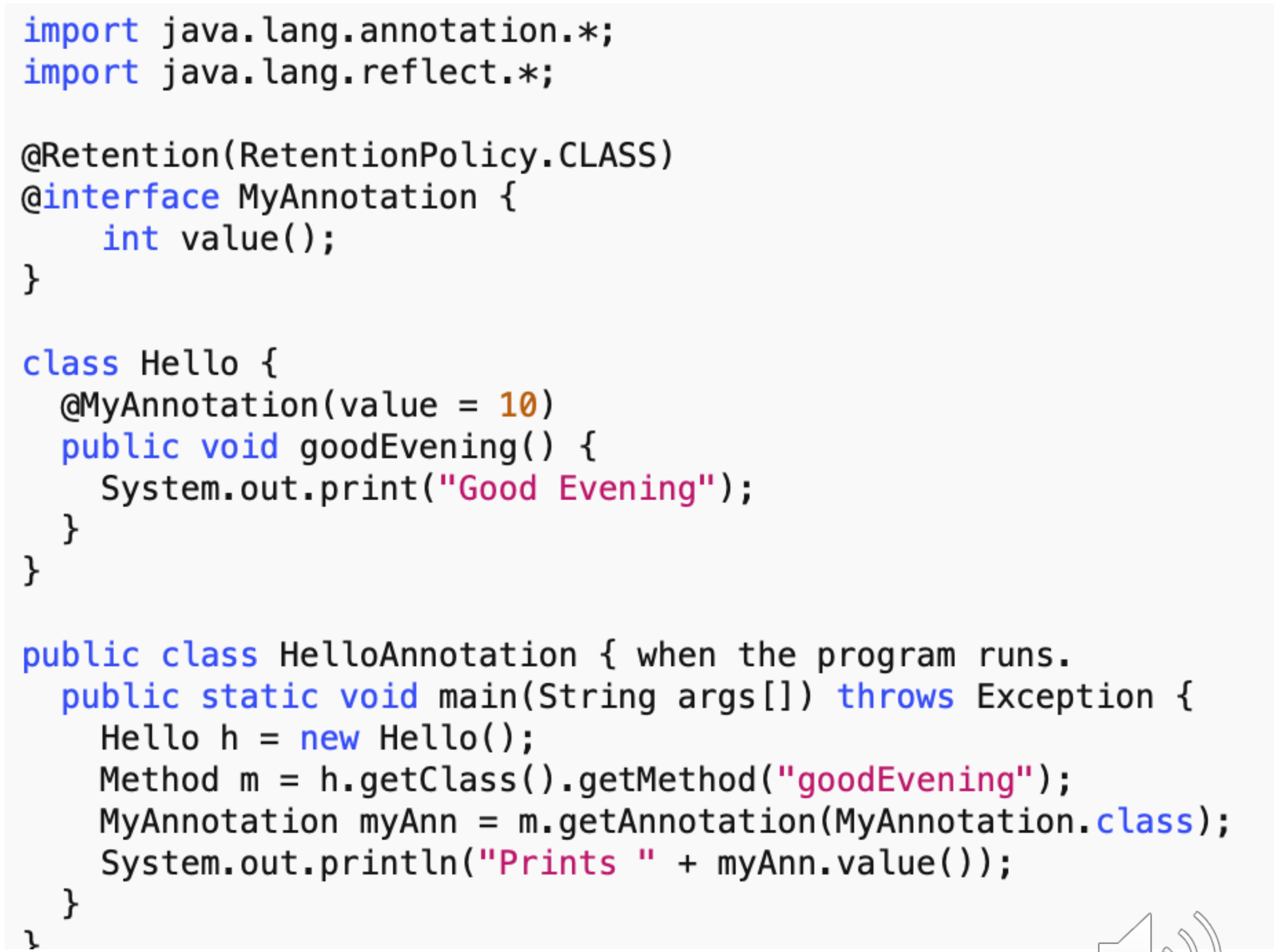
15. What will be the output of the following program?

a) Prints HelloAnnotation 10

b) Compilation Error

c) Runtime Exception

As the retention isn't RUNTIME, trying to get the annotation returns a null reference when the program runs



```
import java.lang.annotation.*;
import java.lang.reflect.*;

@Retention(RetentionPolicy.CLASS)
@interface MyAnnotation {
    int value();
}

class Hello {
  @MyAnnotation(value = 10)
  public void goodEvening() {
    System.out.print("Good Evening");
  }
}

public class HelloAnnotation { when the program runs.
  public static void main(String args[]) throws Exception {
    Hello h = new Hello();
    Method m = h.getClass().getMethod("goodEvening");
    MyAnnotation myAnn = m.getAnnotation(MyAnnotation.class);
    System.out.println("Prints " + myAnn.value());
  }
}
```

16. What is the output of the following program?

a) false true
b) true false
c) true false
d) false false

```
interface Int{}
class Simple implements Int{}
class SimpleTest {
   public static void main(String[] args) {
     try {
       Class c=Class.forName("Simple");
       System.out.print(c.isInterface() + " ");
       c=Class.forName("Int");
       System.out.println(c.isInterface());
     } catch (Exception e) {
       System.out.println(e);
     }
   }
}
```

16. What is the output of the following program?

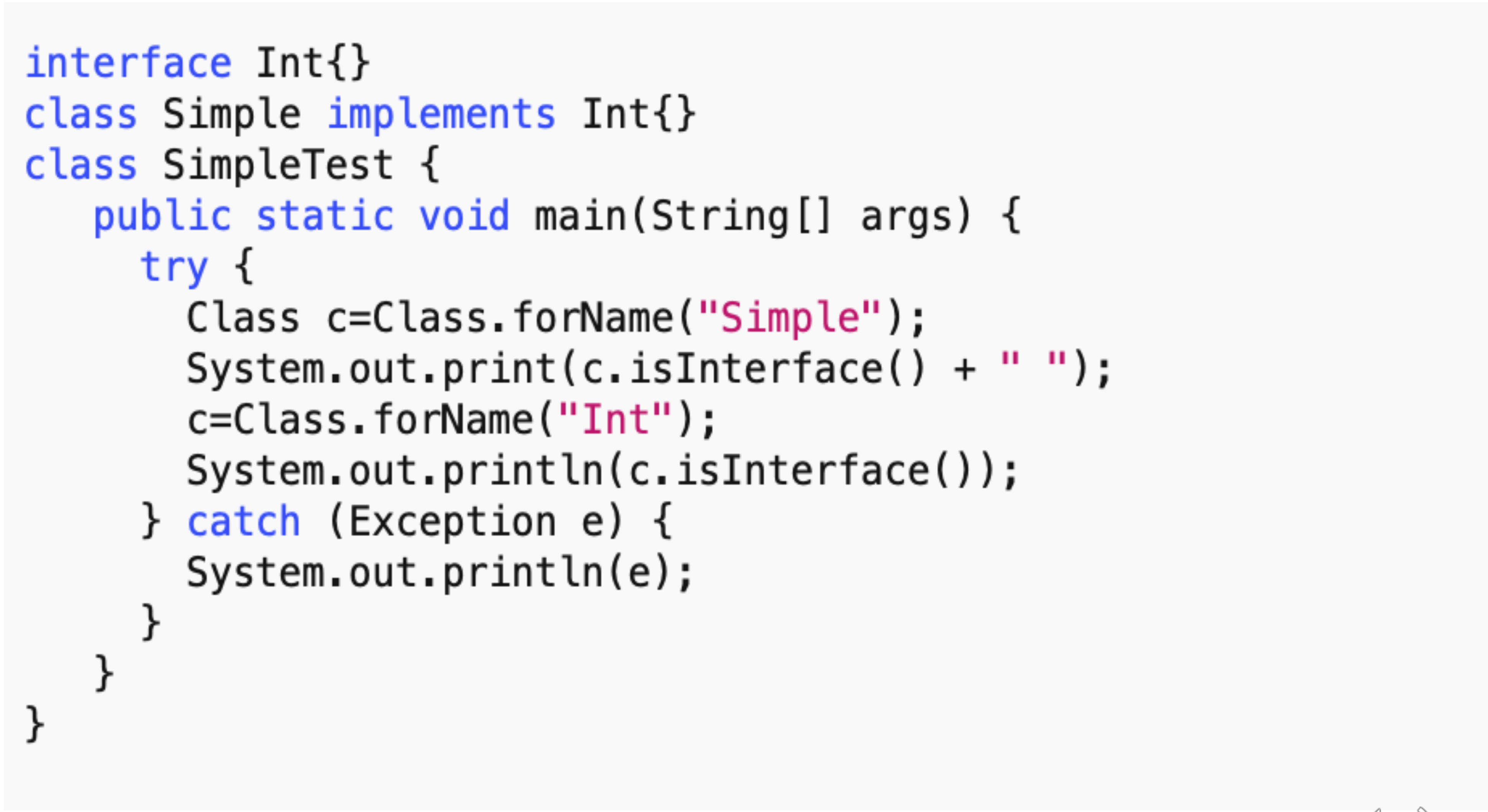
a) false true

b) true false

c) true false

d) false false

An interface kind of extends class (lightweight abstract class)



```
interface Int{}
class Simple implements Int{}
class SimpleTest {
   public static void main(String[] args) {
     try {
       Class c=Class.forName("Simple");
       System.out.print(c.isInterface() + " ");
       c=Class.forName("Int");
       System.out.println(c.isInterface());
     } catch (Exception e) {
       System.out.println(e);
     }
   }
}
```

17. Which of the following can obtain the location of file Reflection.class

Public class Reflection { }

a) Reflection.class.getClassLoader().getResource("Reflection.class")
b) Reflection.getClass().getResource("Reflection.class")
c) Reflection.getClass().getClassLoader().getResource("Reflection.class
   ")
d) Reflection.class.getClassLoader("Reflection.class")

17. Which of the following can obtain the location of file Reflection.class

Public class Reflection { }

a) Reflection.class.getClassLoader().getResource("Reflection.class")
b) Reflection.getClass().getResource("Reflection.class")
c) Reflection.getClass().getClassLoader().getResource("Reflection.class")
d) Reflection.class.getClassLoader("Reflection.class")

getClass() applies to an object, .class to a class. File "Reflection.class" is a resource for the JVM.

18. Consider the following program:

Which of the following is true?

a) It generates one warning (call of the deprecated constructor) when compiling, nothing when running

b) It generates two warnings (definition of the deprecated constructor and call of the deprecated constructor) when compiling, nothing when running

c) No warning when compiling but it generates two warnings when running

d) No warning when compiling, exception when running

e) One warning when compiling, exception when running

```java
class MyClass {
   private float val;
   @Deprecated
   MyClass(int n) {
      val = n;
    }
   MyClass(float f) {
     val = f;
   }
}

public class TestDeprecated {
   public static void main(String[] args) {
int val = 3;
     MyClass obj = new MyClass(3);
   }
}
```

18. Consider the following program:

Which of the following is true?

a) It generates one warning (call of the deprecated constructor) when compiling, nothing when running

b) It generates two warnings (definition of the deprecated constructor and call of the deprecated constructor) when compiling, nothing when running

c) No warning when compiling but it generates two warnings when running

d) No warning when compiling, exception when running

e) One warning when compiling, exception when running

```
class MyClass {
   private float val;
   @Deprecated
   MyClass(int n) {
      val = n;
    }
   MyClass(float f) {
     val = f;
   }
}

public class TestDeprecated {
   public static void main(String[] args) {
int val = 3;
      MyClass obj = new MyClass(3);
   }
}
```

Only developers USING the deprecated function with javac are warned.

19. What is the main purpose of JDBC?

a) Create a connection to the database.
b) Send SQL statements to the database.
c) Processing data and query results
d) All of the above

19. What is the main purpose of JDBC?

a) Create a connection to the database.
b) Send SQL statements to the database.
c) Processing data and query results
d) All of the above

Easy…

20. Which of the following steps are performed when accessing a database via JDBC?

a) Loading the JDBC driver.
b) Setting up the connection.
c) Executing queries or applying changes to the database
d) Close the connection

20. Which of the following steps are performed when accessing a database via JDBC?

a) Loading the JDBC driver.
b) Setting up the connection.
c) Executing queries or applying changes to the database
d) Close the connection

Note: although 99.9% of JDBC applications load the driver, it's also possible to import it (but then the .jar must be in your classpath when you compile)

21. Which package allows Java to access a database?

a) java.sql
b) java.db
c) java.dbms
d) java.jdbc
e) java.lang
f) java.util

21. Which package allows Java to access a database?

a) java.sql
b) java.db
c) java.dbms
d) java.jdbc
e) java.lang
f) java.util

b, c, d don't exist. e is the default package. f contains a lot of useful stuff, but not JDBC.

22. In general you prefer adding a TableView to a:

a) BorderPane
b) StackPane
c) ScrollPane
d) SplitPane
e) TabPane
f) TitledPane

22. In general you prefer adding a TableView to a:

a) BorderPane
b) StackPane
c) [✓ ScrollPane]
d) SplitPane
e) TabPane
f) TitledPane

A TableView is used to display data retrieved from a data source (often a database, but it could as well be the result of streaming an in-memory collection, or data retrieved from the network). You rarely know how many rows you'll display, and the wisest choice is a ScrollPane.

23. FXCollections:

a) Is a package that contains classes for collections that you should use as backend to some JavaFX widgets.
b) Is a class that contains wrapper methods for regular collections so that you can use them as backend to some JavaFX widgets.
c) Is a class that only contains the ObservableList inner class
d) Is an enum that is used for defining the collections that can be used with JavaFx widgets (for instance a TreeSet cannot be used, and doesn't appear in the enum)

23. FXCollections:

a) Is a package that contains classes for collections that you should use as backend to some JavaFX widgets.

b) Is a class that contains wrapper methods for regular collections so that you can use them as backend to some JavaFX widgets.

c) Is a class that only contains the ObservableList inner class

d) Is an enum that is used for defining the collections that can be used with JavaFx widgets (for instance a TreeSet cannot be used, and doesn't appear in the enum)

In Java, plural is often used to name classes that contain static methods (Arrays, Collections, ...)

24. “static” key word in java indicates:

a) Something that you cannot change
b) Something that is attached to the class, not to a particular object instance
c) Something that cannot throw exceptions
d) Something that cannot be overridden (methods) or extended

24. “static” key word in java indicates:

a) Something that you cannot change
b) Something that is attached to the class, not to a particular object instance
c) Something that cannot throw exceptions
d) Something that cannot be overridden (methods) or extended

Something that you cannot change (variable) or cannot be overriden (method) or extended (class) is preceded by ***final***.

25. The protocol that takes a message from a network to another another network is:

a) FTP
b) TCP
c) IP
d) HTTP

25. The protocol that takes a message from a network to another another network is:

a) FTP

b) TCP

c) IP

d) HTTP

IP = Internet Protocol.
Internet means to net(work)s what *international* means to nations.

FTP = File Transfer Protocol (more like HTTP, but dedicated to exchanging files)

TCP = Transmission Control Protocol, computer-to-computer (uses IP underneath)

HTTP = Hyper Text Transfer Protocol, you know this one.

26. Which of these is **not** a build tool?

a) Ant
b) Junit
c) Make
d) Maven

26. Which of these is **not** a build tool?

a) Ant

b) Junit Testing tool.

c) Make

d) Maven

27. The time taken by a message to travel between computers is called:

a) Bandwidth
b) Routing
c) Latency
d) Delay

27. The time taken by a message to travel between computers is called:

a) Bandwidth

b) Routing

c) Latency

d) Delay

Bandwidth refers to how much data can be sent at the same time, not to speed. Routing is finding a circuit (route) to go to computer to computer; it's about traffic redirection.
Delay is just a general term.

28. You need to have the .jar of the JDBC driver you are using when:

a) Only when you compile the program
b) Only when you run the program
c) Both when you compile and run the program

28. You need to have the .jar of the JDBC driver you are using when:

a) Only when you compile the program
b) Only when you run the program
c) Both when you compile and run the program

b is the good answer for 99.9% of the cases, when you load the driver dynamically. Now, you can also import it, but in that case the good answer would be c. And if you are using JavaDB (Derby), as it's part of the standard JDK/JRE you don't need to specify anything special.
A quiz isn't the place for fine-point distinctions. Always pick the most likely.

29. If you serialize an object that contains references, when you reload it the referenced objects will go at exactly the same place in memory.

a) True
b) False

29. If you serialize an object that contains references, when you reload it the referenced objects will go at exactly the same place in memory.

a) True
b) False [✓]

Memory location is the business of the operating system, not yours. It can put things wherever it likes, as long as it returns references to you.

30. To make an object serializable you:

a) You don’t need to do anything – all objects are serializable by default
b) You only need to say that it implements the Serializable interface
c) You need to say that it implements the Serializable interface and implement the two methods writeObject() and readObject() defined in the interface

30. To make an object serializable you:

a) You don't need to do anything – all objects are serializable by default

b) You only need to say that it implements the Serializable interface

c) You need to say that it implements the Serializable interface and implement the two methods writeObject() and readObject() defined in the interface

It's not done by default because it adds overhead (more code) that not everybody needs.