Completely fill in the box corresponding to your answer choice for each question.

1. ■■■ [B] [C] [D]
2. ■■■ [B] [C] [D]
3. ■■■ [B] [C] [D]
4. ■■■ [B] [C] [D]
5. ■■■ [B] [C] [D]
6. ■■■ [B] [C] [D]
7. [A] ■■■ [C] [D]
8. [A] ■■■ [C] [D]
9. ■■■ [B] [C] [D]
10. ■■■ [B] [C] [D]
11. [A] ■■■ [C] [D]
12. [A] ■■■ [C] [D]
13. [A] [B] ■■■ [D]
14. [A] ■■■ [C] [D]
15. [A] ■■■ [C] [D]
16. ■■■ [B] [C] [D]
17. [A] [B] ■■■ [D]
18. [A] ■■■ [C] [D]
19. ■■■ [B] [C] [D]
20. ■■■ [B] [C] [D]
21. [A] [B] [C] ■■■
22. ■■■ [B] [C] [D]
23. [A] [B] [C] ■■■
24. ■■■ [B] [C] [D]
25. ■■■ [B] [C] [D]
26. [A] ■■■ [C] [D]
27. ■■■ [B] [C] [D]
28. ■■■ [B] [C] [D]
29. ■■■ [B] [C] [D]
30. ■■■ [B] [C] [D]
31. ■■■ [B] [C] [D]
32. [A] [B] [C] ■■■
33. ■■■ [B] [C] [D]
34. ■■■ [B] [C] [D]
35. [A] [B] ■■■ [D]
36. ■■■ [B] [C] [D]
37. [A] ■■■ [C] [D]

Number missed: _______ Final Score: _______
public class Kitten {
    private String name = "";
    
    public Kitten(String name) {
        name = name;
    }
    
    public String toString() {
        return "Kitten: " + name;
    }
    
    public boolean equals(Object other) {
        if (this == other) return true;
        if (null == other) return false;
        if (!(other instanceof Kitten)) return false;
        Kitten that = (Kitten) other;
        return this.name.equals(that.name);
    }
}

Assume the following statements have been executed:

Object maggie = new Kitten("Maggie");
Object fiona = new Kitten("Fiona");
Object fiona2 = new Kitten("Fiona");

[3] 1. What is the value of \texttt{maggie}?
    \textbf{A.} the address of a \texttt{Kitten} object
    \textbf{B.} \texttt{null}
    \textbf{C.} automatically set to 0
    \textbf{D.} undefined

[3] 2. What is printed on the console after the following statement is executed?
    \texttt{System.out.println(maggie.toString());}
    \textbf{A.} Kitten:
    \textbf{B.} Kitten: null
    \textbf{C.} Kitten: Maggie

[3] 3. What is the value of the expression \texttt{fiona.equals(fiona2)}?
    \textbf{A.} true
    \textbf{B.} false

[3] 4. What is the value of the expression \texttt{fiona.equals(maggie)}?
    \textbf{A.} true
    \textbf{B.} false

[3] 5. After executing \texttt{Kitten[]} \texttt{kittens = new Kitten[5];}, what is the value of \texttt{kittens[0]}?
    \textbf{A.} null
    \textbf{B.} the address of a \texttt{Kitten} object
    \textbf{C.} automatically set to 0
    \textbf{D.} undefined
public class Doberman {
    private static int dobieCount = 0;
    private String name;

    public Doberman(String name) {
        this.name = name;
        dobieCount++;
    }
    public String reportDobieCount() {
        return name + " says there are " + dobieCount + " dobies."
    }
    public boolean equals(Doberman other) {
        return this.name.equals(other.name);
    }
}

6. If no Doberman instances have been created, what is true about the following line from another class?
   System.out.println("dobieCount: " + Doberman.dobieCount);
   A. It will not compile.
   B. It will compile but will cause a ClassCastException at run-time.
   C. It will print “dobieCount: 0”

7. What would be printed to the console after executing the following statements?
   Doberman fido = new Doberman("Fido");
   Doberman chloe = new Doberman("Chloe");
   System.out.println(chloe.reportDobieCount());
   Doberman prince = new Doberman("Prince");
   A. Chloe says there are 1 dobies.
   B. Chloe says there are 2 dobies.
   C. Chloe says there are 3 dobies.

8. What would be printed to the console after executing the following statements?
   ArrayList daringDobermans = new ArrayList();
   daringDobermans.add(new Doberman("Chloe"));
   System.out.println(daringDobermans.contains(new Doberman("Chloe")));
   A. true
   B. false

9. What would be printed to the console after executing the following statements?
   ArrayList daringDobermans = new ArrayList();
   Doberman chloe = new Doberman("Chloe");
   daringDobermans.add(chloe);
   System.out.println(daringDobermans.contains(chloe));
   A. true
   B. false

10. Given Doberman chloe = new Doberman("Chloe"), what would chloe.toString() return?
    A. Something like “Doberman@deadbeef”
    B. “Chloe”
    C. null
public class Super {
    protected int x = 1;
}

public class Duper extends Super {
    protected int y = 2;

    public Duper(int n) { x += y + n; }

    public String toString() { return new Integer(x).toString(); }
}

public class Andes {
    static int a = 0;
    static boolean incA() { return ++a > 0; }

    public static void main(String[] args) {
        boolean b = Boolean.parseBoolean(args[0]);
        System.out.println(b && incA() ? new Duper(a) : new Duper(a + 1));
    }
}

[3] 11. What is printed when java Andes true is executed on the command line?
    A. 3
    B. 4
    C. 5

[3] 12. What is printed when java Andes false is executed on the command line?
    A. 3
    B. 4
    C. 5

For the next two questions, change line 3 in Andes.java to

    static boolean incA() { return a++ > 0; }

[3] 13. What is printed when java Andes true is executed on the command line?
    A. 3
    B. 4
    C. 5

[3] 14. What is printed when java Andes false is executed on the command line?
    A. 3
    B. 4
    C. 5

[3] 15. Will the expression new Duper() compile?
    A. Yes
    B. No
Assume `Trooper` is defined as follows:

```java
public class Trooper {
    private String name;
    private boolean mustached;
    public Trooper(String name, boolean hasMustache) {
        this.name = name; this.mustached = hasMustached;
    }
    public String getName() { return name; }
    public boolean hasMustache() { return mustached; }

    public boolean equals(Trooper other) {
        if (this == other) return true;
        if (null == other || !(other instanceof Trooper)) return false;
        Trooper that = (Trooper) other;
        return this.name.equals(that.name) && this.mustached == that.mustached;
    }
    public int hashCode() { return 1; }
}
```

And the following has been executed in the same scope as the code in the questions below:

```java
ArrayList<Trooper> troopers = new ArrayList<>();
troopers.add(new Trooper("Farva", true));
troopers.add(new Trooper("Farva", true));
troopers.add(new Trooper("Rabbit", false));
troopers.add(new Trooper("Mac", true));
```

16. What would be the result of the statement `Collections.sort(troopers)`?
   A. The code will not compile.
   B. `troopers` will be sorted in order by name.
   C. `troopers` will be sorted in order by mustache, then name.
   D. `troopers` will not have any duplicate elements.

17. After executing the statement `Set<Trooper> trooperSet = new HashSet<>(troopers)`, what would be the value of `trooperSet.contains(new Trooper("Mac", true))`?
   A. The code will not compile.
   B. `true`
   C. `false`
   D. `void`

18. Given the definitions of `troopers` and `trooperSet` above, what would `trooperSet.size()` return?
   A. 3
   B. 4

19. After the statement `Set<String> stringSet = new HashSet<>(Arrays.asList("meow", "meow"))` executes, what would be the value of `stringSet.size()`?
   A. 1
   B. 2

    A. true
    B. false
Given the following class definitions:

```java
public abstract class Animal {
    public abstract void speak();
    public int legs() { return 4; }
}

public class Mammal extends Animal {
    public void speak() { System.out.println("Hello!"); }
}

public class Canine extends Mammal {
    public void speak() { System.out.println("Grr!"); }
}

public class Dog extends Canine {
    public void speak(String to) { System.out.println("Woof, " + to); }
}

public class Cat extends Mammal {
    public void speak() { System.out.println("Meow!"); }
}
```

21. Say we write a subclass of `Mammal` named `Kangaroo` in which we want to override the `legs` method. Which of the following methods overrides `legs`?
   - A. `public void legs() { System.out.println(2); }`
   - B. `public Object legs() { return new Integer(2); }`
   - C. `public double legs() { return 2; }`
   - D. None of the above.

22. Which of the following is an invocation of the method `public void pet(Canine c)`?
   - A. `pet(new Dog())`
   - B. `pet(new Cat())`
   - C. `pet(new Mammal())`
   - D. `pet(new Animal())`

23. Assuming `Mammal fido = new Dog();` has been executed, what does `fido.speak()` print?
   - A. Hello!
   - B. Woof! Woof!
   - C. Meow!
   - D. None of the above.

24. Assuming `Mammal fido = new Dog();` has been executed, what does `(Mammal) fido).speak()` print?
   - A. Grr!
   - B. Hello!
   - C. Woof! Woof!
   - D. Meow!

25. Assuming the statement `Mammal sparky = new Mammal();` has been executed, which of the following statements will compile but cause a `ClassCastException` at run-time?
   - A. `Dog fido = (Dog) sparky;`
   - B. `Mammal fido = new Dog();`
   - C. `Dog fido2 = (Dog) new Dog();`
   - D. `Cat c = new Dog()`
Given the following classes, which have no-arg constructors:

```java
public class A extends Throwable { ... }
public class B extends A { ... }
public class C extends RuntimeException { ... }
```

26. Which of the following will not compile?
A. ```java
   A foo(B b) throws C {
       if (true) throw new C();
       return new B();
   }
   ```
B. ```java
   A baz(B b) throws B {
       if (true) throw new A();
       return new B();
   }
   ```

27. Which of the following will not compile?
A. ```java
   A foo(B b) throws C {
       if (true) throw new B();
       return new B();
   }
   ```
B. ```java
   A bar(B b) throws C {
       if (true) throw new RuntimeException("c");
       return new B("c");
   }
   ```
C. ```java
   A baz(B b) throws A {
       if (true) throw new A("a");
       return new B("c");
   }
   ```

28. Given the method signature `A bar(B q) throws C`, will this code compile?
A m() throws C {
   return bar(new B());
}

A. Yes
B. No

29. Given the method signature `A bar(B q) throws B`, which of the following will not compile?
A. ```java
   A m() throws C {
       return bar(new B());
   }
   ```
B. ```java
   A m() throws Throwable {
       return bar(new B());
   }
   ```
C. All of the above will compile.

30. What is the highest superclass of all exceptions?
A. `java.lang.Object`
B. `java.lang.Throwable`
C. `java.lang.Exception`
Given the following definitions:

```java
public interface Predicate<T> {
    boolean test(T t);
}

static <E> E find(List<E> es, Predicate<E> p) {
    for (E e: es) if (p.test(e)) return e;
    return null;
}

public interface Function<T, R> {
    R apply(T t);
}

static <E, R> List<R> map(List<E> es, Function<E, R> f) {
    List<R> result = new ArrayList<>();
    for (E e: es) result.add(f.apply(e));
    return result;
}
```

and the list:

```java
List<String> words = Arrays.asList("Welcome", "To", "Java", "8");
```

31. Which of the following expressions would return the first word in `words` that starts with an upper case character?
   A. `find(words, s -> Character.isUpperCase(s.charAt(0)))`
   B. `find(map(words, String::split), a -> a[0].isUpperCase())`
   C. `find(words, s -> s.toUpperCase())`
   D. All of the above.

32. Which of the following expressions would return a list of the lengths of the words in `words`?
   A. `map(words, (String s) -> s.length())`
   B. `map(words, String::length)`
   C. `map(map(words, s -> s.split("")), a -> a.length)`
   D. All of the above.

33. Is `Comparable<T>` a functional interface?
   A. Yes
   B. No
34. What is true about this code?

```java
public static int fac(int n) {
    if (n >= 1) return 1;
    else return n * fac(n + 1);
}
// ...
int fac5 = fac(5);
```

A. Compiles and runs without errors or exceptions.
B. Compiles but program terminates with an error or exception.

```java
public static int f(int n) {
    if (n < 0) throw new IllegalArgumentException("n < 0");
    if (n <= 1) {
        return n;
    } else {
        return f(n - 1) + f(n - 2);
    }
}
```

35. Given the method f above, what is f(5)?

A. 0  
B. 4  
C. 5  
D. 120

36. Given the partial `ArrayListQueue` implementation above, which of the following statements for line 5 would implement `enqueue` in O(1) time? Do not consider any particular implementation for `dequeue`.

A. `elems.add(item);`
B. `elems.add(0, item);`
C. `return elems.remove(elems.size() - 1);`

37. Given the partial `ArrayListQueue` implementation above, which of the following statements for line 5 would implement `enqueue` in O(n) time? Do not consider any particular implementation for `dequeue`.

A. `elems.add(item);`
B. `elems.add(0, item);`
C. `return elems.remove(elems.size() - 1);`