ERRATA FOR BARRON'S COMPUTER SCIENCE A, 7th EDITION.

(1) P 46: The topics in the diagnostic table for questions 7 – 14 should be as follows:

7	`for` loop
8	Program specification
9	Recursion
10	Boolean expressions
11	Hexadecimal
12	`IndexOutOfBoundsException` for `ArrayList`
13	Passing parameters
14	Passing parameters

(2) P 41: The precondition for the `moveForward` method (bottom of the page) should be:

```
Precondition: numLocs >= 0.
```

(3) P 284: The answer explanation for Q 4 should be:

The code segment has the effect of removing all occurrences of 0 from array `arr1`. The algorithm copies the nonzero elements to the front of `arr1`. Then it transfers them to array `arr2`.

(4) P 172: The answer explanation to Q 22 should be:

(C) Segments II and III have errors in the `compareTo` calls. References `c1` and `c2` are of type `Player`, which doesn't have a `compareTo` method, and references `h1` and `h2` are of type `HumanPlayer`, which doesn't have a `compareTo` method.

Note that Segment II can be fixed by downcasting `c1` and `c2` to `ExpertPlayer`:
`if (((ExpertPlayer) c1).compareTo((ExpertPlayer) c2) < 0)`
A cast won't work in Segment III, because you can't cast a `HumanPlayer` to an `ExpertPlayer`.

In Segments I, II, and III, the `getMove` calls are all correct, because `p1`, `p2`, `c1`, and `c2` are all of type `Player` which has a `getMove` method; and `h1` and `h2` are of type `HumanPlayer` which implements `Player` and therefore has a `getMove` method.

(5) P 28 Q 35:

Choice (E) should be:

(E) Every element of array `a` is also in `b`.

On P 51, the answer and explanation for Q 35 are correct. At the end of the explanation, add: "Also, not every element in `b` needs to be in `a`.
For example, if `a = [3, 3, 5]` and `b = [3, 5, 6]`, the method will return `true`.

(6) P 392 Q 18:

Choice (C) should be:

(C) There should not be a method implementation as shown.

On P 423, the answer and explanation for Q 18 should be:

(C) In general, an interface provides method declarations only. No implementations! The methods are automatically `public` and `abstract`, so there is no need to specify this explicitly. Note, however, that in Java 8, *default* methods that have implementations *in the interface* are allowed if the `default` keyword is used at the beginning of the method signature. Default methods will not be tested on the AP exam.

(7) P 188 Q 2:

Implementation III should be:

III `return Double (Math.sqrt(d.doubleValue()));`

On P 200, the final sentence of the answer/explanation for Q 2 should be: Segment III fails because you can't use the `Double` constructor to create a new object without the keyword `new`.

Note that the statement
`return (Double)Math.sqrt(d.doubleValue());`
works because `Math.sqrt(d.doubleValue())` is auto-boxed into a `Double`.
Casting a `Double` to a `Double` is redundant, but it still works!

(8) P 354:

The first line of code in the SHUFFLING algorithm should be:

```
for (int k = arr.length – 1; k > 0; k--)
```

(9) P 10 Q 6:

The precondition should read as follows:

```
Precondition: arr is an array of nonzero length, and is
initialized with int values
```

(10) P 75:

Note that the `ClassCastException` is no longer in the AP Java subset. And the description of it on P142 is now an Optional Topic.

(11) P 394 Q 21:

The question, following the piece of code, should now read as follows:

```
Which is the best precondition for the segment? (You may
assume that a[0]. . . a[-1] represents an empty array.)
(A)  a[0]. . . a[n-1] contains no positive integers.
(B)  a[0]. . . a[n-1] contains no negative integers.
(C)  a[0]. . . a[n-1] contains no nonnegative integers.
(D)  a[0]. . . a[n-1] contains no occurrences of zero.
(E)  The updated value of n is less than or equal to the
     value of n before execution of the segment.
```