ANSWER KEY

1. **C**
2. **A**
3. **B**
4. **E**
5. **A**
6. **D**
7. **B**
8. **E**
9. **C**
10. **D**
11. **E**
12. **D**
13. **C**
14. **A**

ANSWERS EXPLAINED

1. **(C)** Segment III works because if you enter an age of 90, say, `category` will correctly be assigned `"Senior"`, and none of the other `else` pieces of code will be executed. Similarly, if you enter an age corresponding to an adult or a child, only the correct assignment is made. Segment I fails because if you enter an age of 90, `category` will be assigned `"Senior"`, but then will be changed to `"Adult"` when the age passes the second test. Segment II uses incorrect syntax. The segment will work if you change the second test to

   ```
   if (age >= 18 && age <= 64)
   ```

2. **(A)** The algorithm prints the current index of `"o"` in the string, and then creates a new substring containing all remaining characters following that `"o"`. Here is the series of substrings and the corresponding output for each (the symbol ␣ denotes a blank character):

   ```
   How␣do␣you␣do?    1
   w␣do␣you␣do?      3
   ␣you␣do?          2
   u␣do?             3
   ```

3. **(B)** Here is a description of the algorithm:

   Make a copy of `phrase` in `newPhrase`.
   Find the first occurrence of `ch` in `newPhrase` (`pos` is the index).
   If you found it, concatenate to `str` the characters in `newPhrase` from `0` to `pos-1`.
   Change `newPhrase` to contain all characters from `ch` to the end, excluding `ch`.
   Repeat the process until there are no more occurrences of `ch` in `newPhrase`.

   So Line 12 is wrong because `newPhrase.substring(0,pos-1)` will not include the character at `pos-1`, which means that the string returned will lose a character that is *not* equal to `ch`.

4. **(E)** The program has found a stand-alone word if the characters `before` and `after` are both blank. Choice E tests that they are not letters between `"a"` and `"z"`, i.e., they must be blank. Choices A and B fail because you must use `compareTo` for inequality tests on strings. Choices C and D allow at least one of `before` and `after` to be a letter, which would mean that `word` was not a stand-alone word.