Trail: Learning the Java Language
Lesson: Numbers and Strings

Answers to Questions and Exercises: Characters and Strings

Questions

Question 1: What is the initial capacity of the following string builder?

```
StringBuilder sb = new StringBuilder("Able was I ere I saw Elba.");
```

Answer 1: It's the length of the initial string + 16: 26 + 16 = 42.

Question 2: Consider the following string:

```
String hannah = "Did Hannah see bees? Hannah did.";
```

Question 2a: What is the value displayed by the expression `hannah.length()`?
Answer 2a: 32.

Question 2b: What is the value returned by the method call `hannah.charAt(12)`?
Answer 2b: `e`.

Question 2c: Write an expression that refers to the letter `b` in the string referred to by `hannah`.
Answer 2c: `hannah.charAt(15)`.

Question 3: How long is the string returned by the following expression? What is the string?

```
"Was it a car or a cat I saw?".substring(9, 12)
```

Answer 3: It's 3 characters in length: `car`. It does not include the space after car.

Question 4: In the following program, called `ComputeResult`, what is the value of `result` after each numbered line executes?

```
public class ComputeResult {
    public static void main(String[] args) {
        String original = "software";
        StringBuilder result = new StringBuilder("hi");
        int index = original.indexOf('a');

/*1*/   result.setCharAt(0, original.charAt(0));
/*2*/   result.setCharAt(1, original.charAt(original.length()-1));
/*3*/   result.insert(1, original.charAt(4));
/*4*/   result.append(original.substring(1,4));
/*5*/   result.insert(3, (original.substring(index, index+2) + " "));

        System.out.println(result);
    }
}
```

Answer 4:

1. si
2. se
3. swe
4. sweoft
5. swear oft

Exercises

Exercise 1: Show two ways to concatenate the following two strings together to get the string `"Hi, mom."`:

```
String hi = "Hi, ";
String mom = "mom.";
```

Answer 1: `hi.concat(mom)` and `hi + mom`.

Exercise 2: Write a program that computes your initials from your full name and displays them.
Answer 2: `ComputeInitials`

```
public class ComputeInitials {
    public static void main(String[] args) {
        String myName = "Fred F. Flintstone";
        StringBuffer myInitials = new StringBuffer();
        int length = myName.length();

        for (int i = 0; i < length; i++) {
            if (Character.isUpperCase(myName.charAt(i))) {
                myInitials.append(myName.charAt(i));
            }
        }
        System.out.println("My initials are: " + myInitials);
    }
}
```

Exercise 3: An anagram is a word or a phrase made by transposing the letters of another word or phrase; for example, "parliament" is an anagram of "partial men," and "software" is an anagram of "swear oft." Write a program that figures out whether one string is an anagram of another string. The program should ignore white space and punctuation.
Answer 3: `Anagram`

```
public class Anagram {

    /**
     * Tests whether the passed-in strings are anagrams --
     * containing the exact same number of each letter.
     * Punctuation, case, and order don't matter.
     *
     * @return true if the strings are anagrams; otherwise, false
     */
    public static boolean areAnagrams(String string1,
                                      String string2) {

        String workingCopy1 = removeJunk(string1);
        String workingCopy2 = removeJunk(string2);

            workingCopy1 = workingCopy1.toLowerCase();
```

```
            workingCopy2 = workingCopy2.toLowerCase();

            workingCopy1 = sort(workingCopy1);
            workingCopy2 = sort(workingCopy2);

        return workingCopy1.equals(workingCopy2);
    }

    /**
     * Removes punctuation & spaces -- everything except
     * letters from the passed-in string.
     *
     * @return a stripped copy of the passed-in string
     */
    protected static String removeJunk(String string) {
        int i, len = string.length();
        StringBuilder dest = new StringBuilder(len);
                char c;

            for (i = (len - 1); i >= 0; i--) {
                c = string.charAt(i);
                if (Character.isLetter(c)) {
                        dest.append(c);
                }
            }

        return dest.toString();
    }

    /**
     * Sorts the passed-in string.
     *
     * @return a sorted copy of the passed-in string
     */
    protected static String sort(String string) {
            char[] charArray = string.toCharArray();

            java.util.Arrays.sort(charArray);

        return new String(charArray);
    }

    public static void main(String[] args) {
        String string1 = "Cosmo and Laine:";
        String string2 = "Maid, clean soon!";

        System.out.println();
        System.out.println("Testing whether the following "
                           + "strings are anagrams:");
        System.out.println("    String 1: " + string1);
        System.out.println("    String 2: " + string2);
        System.out.println();

        if (areAnagrams(string1, string2)) {
            System.out.println("They ARE anagrams!");
        } else {
            System.out.println("They are NOT anagrams!");
        }
        System.out.println();
    }
}
```

Problems with the examples? Try Compiling and Running the Examples: FAQs.
Complaints? Compliments? Suggestions? Give us your feedback.

Previous page: Questions and Exercises: Characters and Strings