Exercises: Primitive Variables

Code Reading

1. Consider the following variable identifiers. Mark whether the identifier is valid or invalid. If it is invalid, state why.

<table>
<thead>
<tr>
<th>valid</th>
<th>invalid (starts with number)</th>
<th>valid</th>
<th>invalid (space)</th>
<th>invalid (keyword)</th>
</tr>
</thead>
<tbody>
<tr>
<td>xCoordinate</td>
<td>2ndName</td>
<td>FINAL_AMOUNT</td>
<td>my string</td>
<td>class</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Consider the following lines of code. At the end of the code, which variables have been declared, and which have been initialized?

```java
public class Exercise {
    public static void main(String[] args) {
        String firstName, lastName;
        String hometown = "La Crosse", state;
        firstName = "James";
    }
}
```

**Solution:** Declared: `firstName, lastName, hometown, state`  
Initialized: `hometown, firstName`
3. Consider the following lines of code. At each line, state the current values of all the initialized variables.

```java
public class Exercise {

    public static void main(String[] args) {

        String firstName, lastName;
        String hometown = "La Crosse", state; hometown = "La Crosse"

        firstName = "James"; hometown = "La Crosse", firstName = "James"
        lastName = "Smith"; hometown = "La Crosse", firstName = "James", lastName = "Smith"
        state = firstName; hometown = "La Crosse", firstName = "James", lastName = "Smith", state = "James"
        lastName = state + lastName; hometown = "La Crosse", firstName = "James", lastName = "JamesSmith", state = "James"
        hometown = lastName; hometown = "JamesSmith", firstName = "James", lastName = "JamesSmith", state = "James"

    }
}
```
Code Writing

4. For the following parts, (a) declare a String variable called state, and (b) initialize the variable to ‘Wisconsin’. On (c), declare and initialize the variable all on one line.

(a) **Solution:** String state;

(b) **Solution:** state = "Wisconsin";

(c) **Solution:** String state = "Wisconsin";

5. For the following parts, (a) declare a Scanner variable called scan, and (b) instantiate the variable to read from System.in. On (c), declare and instantiate the variable all on one line.

(a) **Solution:** Scanner scan;

(b) **Solution:** scan = new Scanner(System.in);

(c) **Solution:** Scanner scan = new Scanner(System.in);

6. Below are two variables, a and b. Write code to assign the value of variable a to variable b. You should only use the variables provided in your solution, not any string literals (e.g., “one” or “two”).

```java
String a = "one";
String b = "two";

b = a;
```
7. Below, write code to declare and instantiate a Scanner variable called readIn that reads from System.in. Print a prompt for the user to input their major, and then use the nextLine() method to read in their answer and store it in a String variable called major. Consider how you might ensure there is a space between the prompt and the user’s response. Test your solution by typing it into Eclipse.

Solution:

```java
Scanner readIn = new Scanner(System.in);
System.out.print(“What is your major? ”);

// two options for answers
// declare and initialize on separate lines
String major;
major = readIn.nextLine();

// declare and initialize on one line
String major = readIn.nextLine();
```