CIS 255 • Java Programming
Laboratory Assignment 6: Methods with Parameters

In this laboratory assignment you will work with a single `void` method with a parameter to display one of at least three state nicknames.

Background

Many states have at least one nickname commonly associated with them. In this program, you will ask the user to choose from a menu containing several states. You will use a `void` method with a single parameter to receive the user's selection and display its nickname. If the user selects something other than what's included in the menu, the method will display an error message instead.

Step 1: Download the Starter File and Launch TextPad

On Blackboard, click the link for the starter file to download it. Make sure you insert your first initial and last name where indicated (**FLast_Lab06.java**) and that you specify the file location. Once you have downloaded the file, open it in TextPad or a similar Java code editor.

Step 2: Customize Your Code and Add an Import Statement

Edit the comments at the top of the code to include your name, the course number, the laboratory assignment number, and a brief description of the program's contents. You may also wish to include the date (perhaps after the laboratory assignment number). Also, modify the class name to match your file name. As you continue your code, you may go ahead and add comments to the statements or groups of statements that you believe deserve additional explanation, or you may wait until after you have written your entire program to add comments.

This program will deal with interactive input via the console, so add the `import` statement that corresponds to input from the keyboard.

Step 3: Declare Variables in the `main` Method

You will need to declare a variable capable of storing the user's choice from a menu listing several states. The simplest option would be to use a whole number variable so that the user's choice will be a number corresponding to the user's menu choice. Declare a `Scanner` object to receive the user's input as well.

Step 4: Display the Menu and Obtain Input

Use a series of output statements to display the list of states from which you want the user to select. You must have at least three state names from the following (or others if your favorite is not included):

Alabama	The Yellowhammer State	**Kentucky**	The Bluegrass State
Arkansas	The Natural State	**Louisiana**	The Pelican State
Florida	The Sunshine State	**Mississippi**	The Magnolia State
Georgia	The Peach State	**Tennessee**	The Volunteer State

Display each state with a corresponding number that the user can enter to choose that state. For example:

```
Choose from the following states:
1. Alabama
2. Georgia
3. Florida
Enter your selection now:
```

If a menu is lengthy, you may want to put the code to display it in a separate method as well.

After displaying the menu, use an input statement to read the user's choice from the keyboard.

Step 5: Write the Definition of the `void` Method with a Parameter

A programmer could include the decision structure that displays the nickname of the user's choice in the `main` method, but this code could also be placed in a separate method.

After the closing bracket of the `main` method, but before the closing bracket of the class, leave a blank line; then, write the header for the method to display the nickname.

Don't write another method's definition inside the curly braces for the `main` method! Each method's definition should be separate.

Java allows you to write the method definitions in any order; you can put `main` as the first method, the last method, or anywhere in the middle as desired.

This header will be similar to the header for `main`, but the name of the method should indicate its purpose, and the parentheses after the method name should contain a single parameter capable of receiving the value of the user's choice (a whole number).

If a method has multiple parameters, each parameter must include a data type and a name, even if some of the parameters share the same data type.

After the header for this new method, add another set of curly braces. Inside these curly braces, write a decision structure that checks the value of the parameter variable, displaying the corresponding state nickname for each choice or an error message if the parameter's value does not match any of the values in the menu.

The variable name used to refer to an argument received from another method should be the parameter name, which may or may not be the same as the argument name!

Step 6: Call the `void` Method

Back in the `main` method, proceed to a new line after the input statement. Call the `void` method you just defined by typing its name, an opening parenthesis, the name of the variable used to receive input, a closing parenthesis, and a semicolon.

Don't write the entire method definition here! A method call is a single line indicating when to execute the statements in another method and what arguments to send to its parameters.

Although a parameter in a method header must include a data type, an argument in a method call does not include a data type.

The number of arguments in the parentheses of a method call must match the number of parameters in the header of the method definition, and the data type of the argument in each position must be compatible with the parameter in the same position!

The value the user enters will be copied from the input variable in `main` to the parameter variable in the other method.

Step 7: Compile Your Code and Execute the Byte Code

To determine if there are any syntax errors in your source code, and to generate the byte code once your source code is free of syntax errors, select the item **Compile Java** from the **External Tools** submenu of the **Tools** menu. The results of the build process will appear in the **Tool Output** area below your code. If the Tool Output area indicates that there are syntax errors in your code, you may double-click the line that describes each syntax error to see the line that contains the error (or, in some cases, the line after the error).

Once you receive a message in the Tool Output area that the tool completed successfully, you're ready to test your program. In the **External Tools** submenu of the **Tools** menu, select **Run Java Application**. Make sure that the program displays the appropriate state nickname (or an error message) for the value you provide.

To test multiple values, you'll need to run the program multiple times. You could nest all of this logic inside a loop to allow for multiple nickname requests, but this is not required.

Step 8: Add Inline Comments and Submit Your Code

Make sure that your code has **three or more inline comments** near the statements that you believe warrant the additional description (especially those that deal with methods). These can be simple line comments at the end of a line of code that needs explanation, or block comments where each precedes a series of statements described in the comment.

Follow the instructions on the course website to attach and submit your source code.

Make sure you complete the submission process! If you attach your file without submitting the assignment, your file will not be received!