

```
public static String compute(String word1, String word2) {  
    String result = "";  
    if (word1.length() == word2.length()) {  
        for (int i = 0; i < word1.length(); i++) {  
            result = result + word1.charAt(i) + word2.charAt(i);  
        }  
    }  
    return result;  
}
```

The diagram illustrates a code graph for the provided Java code. Nodes are represented by colored circles, and edges represent the flow of execution or data dependencies. The graph is divided into two main clusters: a purple cluster on the left and a yellow/orange cluster on the right.

- Purple Cluster (Left):** This cluster contains nodes for the initial setup and the loop's control logic. It includes nodes for `public static`, `String`, `compute`, `String`, `word1`, `String`, `word2`, `{`, `String`, `result`, `=`, `"`, `"`, `;`, `if`, `(`, `word1`, `.`, `length`, `()`, `=`, `word2`, `.`, `length`, `()`, `)`, `{`, `for`, `(`, `int`, `i`, `=`, `0`, `;`, `i`, `<`, `word1`, `.`, `length`, `()`, `;`, `i`, `++`, `)`, `{`, `result`, `=`, `result`, `+`, `word1`, `.`, `charAt`, `(`, `i`, `)`, `+`, `word2`, `.`, `charAt`, `(`, `i`, `)`, `;`, `}`, `}`, `return`, `result`, `;`, and `}`.
- Yellow/Orange Cluster (Right):** This cluster contains nodes for the loop's body logic. It includes nodes for `String`, `result`, `=`, `"`, `"`, `;`, `if`, `(`, `word1`, `.`, `length`, `()`, `=`, `word2`, `.`, `length`, `()`, `)`, `{`, `for`, `(`, `int`, `i`, `=`, `0`, `;`, `i`, `<`, `word1`, `.`, `length`, `()`, `;`, `i`, `++`, `)`, `{`, `result`, `=`, `result`, `+`, `word1`, `.`, `charAt`, `(`, `i`, `)`, `+`, `word2`, `.`, `charAt`, `(`, `i`, `)`, `;`, `}`, `}`, `return`, `result`, `;`, and `}`.

Edges connect the nodes in a way that represents the flow of execution. For example, the edge from `if` to `for` indicates that the loop is only executed if the condition is true. The edge from `result` to `result` in the loop body indicates that the result is updated in each iteration.