

```
public static int compute(String word) {  
    char[] letters = {'a', 'e', 'i', 'o', 'u'};  
    int result = 0;  
  
    for (int i = 0; i < word.length(); i++) {  
        for (int j = 0; j < letters.length; j++) {  
            if (word.charAt(i) == letters[j]) {  
                result++;  
            }  
        }  
    }  
  
    return result;  
}
```

The diagram illustrates the control flow of the provided Java code. Nodes, represented by colored circles, are placed at various points in the code: at the start of the method, before the 'letters' array declaration, before the 'result' variable declaration, before the first 'for' loop, at the start of the inner 'for' loop, before the 'if' statement, after the 'result++' statement, and at the end of the method. Edges, represented by lines, show the flow of execution: a purple line connects the start node to the node before the first 'for' loop; a yellow line connects the node before the first 'for' loop to the node at the start of the inner 'for' loop; a red line connects the node before the inner 'for' loop to the node before the 'if' statement; an orange line connects the node before the 'if' statement to the node after 'result++'; a purple line connects the node after 'result++' to the node at the end of the inner 'for' loop; a red line connects the node at the end of the inner 'for' loop to the node at the end of the first 'for' loop; and a yellow line connects the node at the end of the first 'for' loop to the node before the 'return' statement.