

```
public static String compute(String text) {  
    String result = "";  
    for (int i = 0; i < text.length(); i++) {  
        char c = text.charAt(i);  
        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z')) {  
            result = result + c;  
        }  
    }  
    return result;  
}
```

The diagram illustrates the control flow of the provided Java code. Nodes are represented by colored circles, and edges represent the flow of execution.

- Entry Node:** A black circle at the top left, representing the start of the `compute` method.
- Initialization:** A purple circle representing the assignment `String result = ""`.
- Loop Header:** A black circle representing the `for` loop's initialization and condition check.
- Loop Body:**
 - A purple circle representing the character access `char c = text.charAt(i)`.
 - A yellow circle representing the `if` statement's condition check.
 - An orange circle representing the string concatenation `result = result + c`.
- Exit/Return:** A red circle representing the `return result` statement.

The flow starts at the entry node, proceeds to the initialization of `result`, then enters the `for` loop. The loop body contains a character access and a conditional check. The `if` statement branches into two paths: one leading to the string concatenation and another leading to the loop's end. Both paths converge back to the loop's increment and condition check. The graph ends at the `return result` statement.