

```
public static int compute(int value) {  
    if (value == 1) {  
        return 1;  
    }  
    return compute(value - 1) * value;  
}
```

The diagram illustrates the recursive call stack for the `compute` function. Nodes are represented by colored circles: purple for the initial call, orange for recursive calls, and yellow for the base case. Lines connect the nodes to show the sequence of calls and returns.

- Initial Call (Purple):** `compute(4)`. A node is placed above the `int` parameter.
- First Recursive Call (Orange):** `compute(3)`. A node is placed above the `value` parameter.
- Second Recursive Call (Orange):** `compute(2)`. A node is placed above the `value` parameter.
- Third Recursive Call (Orange):** `compute(1)`. A node is placed above the `value` parameter.
- Base Case (Yellow):** `return 1;`. A node is placed above the `1` value.

The flow of execution is as follows:

- The initial call `compute(4)` calls `compute(3)`.
- `compute(3)` calls `compute(2)`.
- `compute(2)` calls `compute(1)`.
- `compute(1)` returns `1` to `compute(2)`.
- `compute(2)` returns `2` to `compute(3)`.
- `compute(3)` returns `6` to `compute(4)`.
- `compute(4)` returns `24` to the caller.