

The diagram illustrates the execution of a recursive function `add` and its calls within a `sum` function. It shows six call stacks, each representing a different point in the execution. The stacks are connected by arrows indicating the flow of control and return values.

- Stack 1:** `add()` (green header), `...`, `sum()` (red header), `...`, `c (3)` (blue header), `...`, `f (2)` (red header), `t (4)` (red header), `s` (red header), `f (2)` (red header), `t (3)` (red header), `s (2)` (red header), `l (2)` (blue header), `r (3)` (blue header). Return value: 5.
- Stack 2:** `add()` (green header), `...`, `sum()` (red header), `...`, `c (3)` (blue header), `...`, `f (2)` (red header), `t (4)` (red header), `s` (red header), `f (2)` (red header), `t (3)` (red header), `s (2)` (red header). Return value: 5.
- Stack 3:** `add()` (green header), `...`, `sum()` (red header), `...`, `c (3)` (blue header), `...`, `f (2)` (red header), `t (4)` (red header), `s (5)` (red header). Return value: 5.
- Stack 4:** `add()` (green header), `...`, `sum()` (red header), `...`, `c (3)` (blue header), `...`, `f (2)` (red header), `t (4)` (red header), `s (5)` (red header), `l (5)` (blue header), `r (4)` (blue header). Return value: 9.
- Stack 5:** `add()` (green header), `...`, `sum()` (red header), `...`, `c (3)` (blue header), `...`, `f (2)` (red header), `t (4)` (red header), `s (5)` (red header). Return value: 9.
- Stack 6:** `add()` (green header), `...`, `sum()` (red header), `...`, `c (3)` (blue header), `...`. Return value: 9.

A **Stack-Frame** bracket points to the `add(5,4)` stack, showing its local variables `l (5)` and `r (4)`.