

```
1 console.clear();
2
3 /*
4
5 -----
6
7 💡 MEMOIZATION FUNCTION
8
9 -----
10
11 memoize(fn) takes a function and returns a new optimized version of it. It stores results in a cache so that next time the
12 same input comes, the answer is returned instantly without recalculating.
13
14 -----
15 */
16
17 const memoize = (fn) => {
18
19     // Cache object to store previous results
20
21     const cache = {};
22
23     return function (num) {
24
25         // If the result already exists → return from cache
26
27         if (cache[num]) {
28
29             console.log("📦 Returned from Cache:", num);
30
31             return cache[num];
32         }
33
34         // If result does NOT exist → compute it
35
36         console.log("⚡ Calculated Fresh:", num);
37
```

```
38     const result = fn(num); // Call the original heavy function
39
40     // Store the computed result inside cache
41
42     cache[num] = result;
43
44     // Return the computed result
45
46     return result;
47 };
48 };
49
50 /*
51
52 -----
53
54 ⌚ A VERY SLOW FUNCTION (simulating heavy calculations)
55
56 -----
57
58 slowSquare(num) calculates the square of a number BUT includes
59
60 a time-consuming loop to simulate a heavy CPU task like:
61
62 - Complex mathematical calculation
63
64 - Image processing
65
66 - Large dataset manipulation
67
68 - API data processing
69
70 -----
71
72 */
73
74 const slowSquare = (num) => {
75
```

```
76     // Fake heavy processing
77
78     for (let i = 0; i < 1000000000; i++) {}
79
80     return num * num;
81 };
82
83 // Create a memoized (optimized) version of slowSquare
84
85 const memoizedSquare = memoize(slowSquare);
86
87 /*
88
89 -----
90
91 🚧 TESTING THE MEMOIZED FUNCTION
92
93 -----
94
95 First call    → Slow (actual calculation happens)
96
97 Second call   → Instant (result returned from cache)
98
99 -----
100
101 */
102
103 console.log(memoizedSquare(5)); // Fresh compute
104
105 console.log(memoizedSquare(5)); // From cache
106
107 console.log(memoizedSquare(10)); // Fresh compute
108
109 console.log(memoizedSquare(10)); // From cache
```