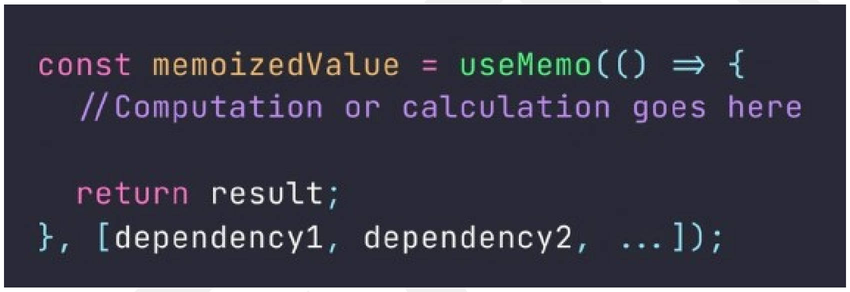
useMemo() React Hooks

- In React, the useMemo hook is used to **memoize the result** of a computation, so that the computation is only executed when its **dependencies change**.
- This can be useful to **optimize the performance** of expensive calculations or **computations** in functional components.
- **Syntax :**



```
const memoizedValue = useMemo(() => {
  //Computation or calculation goes here

  return result;
}, [dependency1, dependency2, ...]);
```

- **() =>... :** This is an inline function that contains the computation or calculation you want to memoize.
- **Dependency1, dependency2, ...:** An array of dependencies. If any of these dependencies change, the memoized value will be recalculated.

- **Example :**

```
import React, { useMemo, useState } from 'react'

function Usememo() {

     const [count,setcount]=useState(0);
     const [item,setitem]=useState(0);

     const memorized = useMemo(() => {
          console.log("CNT = "+(count*10));
          return count*count;
     },[count])

return (
    <>
          <center>
               <h1>Count : {count}</h1>
               <h1>Square : {memorized}</h1>
               <br />
               <button onClick={()=>{setcount(count+1)}}>Count Update</button>
               <button onClick={()=>{setitem(item+1)}}>Item Update</button>
          </center>
    </>
    )
}

export default Usememo
```