useMemo

Is used to handle **values** rather than functions

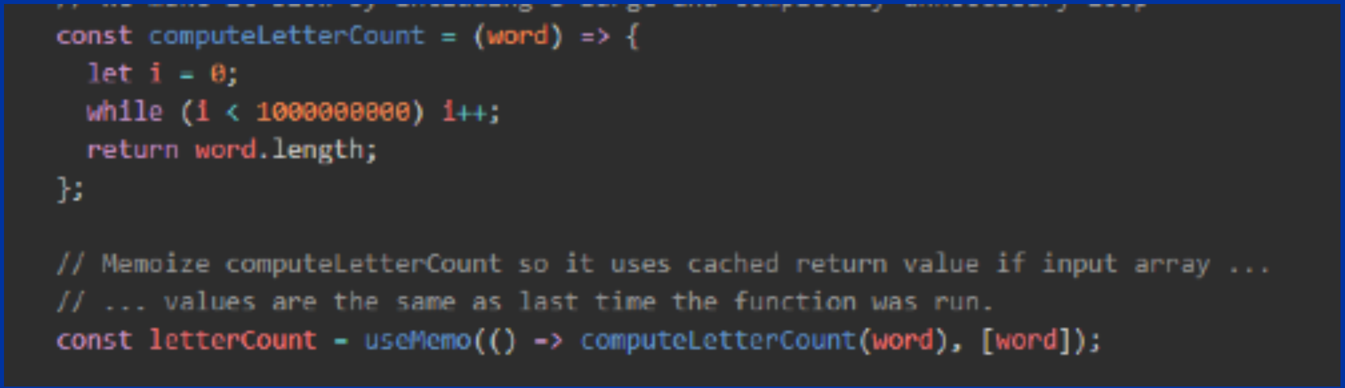
The useMemo is similar to useCallback hook as it accepts a function and a list of dependencies but it **returns the memoized value** returned by the passed function.

It recalculates the **value** only when one of its dependencies has change. It is useful to avoid expensive calculations on every render when the returned **value** is not going to change.

Eg a large functions that edit json data returned from an API call.

useMemo

useMemo allows you to memoize expensive functions so that you can avoid calling them on every render.



```
// [illegible]
const computeLetterCount = (word) => {
  let i = 0;
  while (i < 1000000000) i++;
  return word.length;
};

// Memoize computeLetterCount so it uses cached return value if input array ...
// ... values are the same as last time the function was run.
const letterCount = useMemo(() => computeLetterCount(word), [word]);
```

memo() is a higher order function
useMemo() is a hook to memoize expensive functions so they don't run again on re-render