11/19 숙제 답안

strict mode 꺼주세요.. why? https://stackoverflow.com/questions/72489140/react-18-strict-mode-causing-component-to-render-twice

> React StrictMode calls all Effects twice to make sure their cleanup/unmount handlers work as intended. You may need to change your effects accordingly, even if they have an empty dependency list and would normally not unmount before the site is closed.
>
> Note, this only happens in Strict + development mode. In a production build, effects will only be called once and when their dependencies change.

홈웍>

- 1초씩 카운팅하다가 10초 뒤에 꺼지는 가능을 가진 컴포넌트를 만들어주세요
- 오늘 수업에서 다뤘던 setInterval, clearInterval, setTimeout 함수를 활용해서 만들어주세요
- (++추가) 커스텀 useToggle 훅과 토글 버튼을 만들어서 클릭할 때마다 컴포넌트가 display/undisplay 되는 기능을 추가해주세요 (단, 돔에서 언마운트 될 때는 클린업 함수로 clearInterval을 사용해줍니다)

```
// 힌트

const interval = setInterval(() =>{console.log('hi')}, 1000);
setTimeout(()=>{clearInterval(interval)},5000)
```

답안> (주석을 잘 읽어주세요)

```
import "./styles.css";

import { useEffect, useState } from "react";

// 타이머 컴포넌트 생성
const Timer = ({ count }) => {
  const [counter, setCounter] = useState(0);

  useEffect(() => {
    let interval = setInterval(() => {
      setCounter((prev) => prev + 1);
```

```
    }, 1000); // let으로 선언한 이유는 뒤에...
    const handleClearInterval = () => {
      // 반복되는 코드라서 함수로 만들어줬어요
      if (interval) {
        clearInterval(interval);
        console.log("interval이 철회되었습니다");
        interval = false;
        // interval값을 프린트해보면 유니크한 interval키 값이 담기는데
        // 자바스크립트 내부에서 주는 키라서 clearInterval 작업을 할 때 참조만 해요
        // 그래서 얘는 clearInterval을 해줘도 삭제되지 않아요
        // 여기서 무효화 작업이 setTimeout이나 클린업 함수에서 반복되지 않도록 interval값을 false로 업데이트해
        // 값이 있는 경우에만 이 clearInterval이 실행되게 해주었답니다
      }
    };
    setTimeout(() => {
      handleClearInterval();
    }, 1000 * count);
    return () => {
      handleClearInterval();
    };
  }, []); // 컴포넌트가 맨 처음 돔에 렌더링 될 때 딱 한번 실행시키겠다
  return <h1>{counter}</h1>;
};

// 커스텀 useToggle 훅 만들기
const useToggle = (initialState = false) => {
  const [toggle, setToggle] = useState(initialState);
  const handleToggle = () => {
    setToggle((toggle) => !toggle);
  };
  return [toggle, handleToggle];
};

export default function App() {
  const count = 10; // 전체 타이머 시간을 우연하게 바꿀 수 있도록 빼줬음
  const [toggle, handleToggle] = useToggle(false);
  // 실습에서는 message를 useEffect로 toggle값에 따라 바뀌게 상태로 선언해줬는데 다시 보니
  // boolean 상태 값에 대해서 이렇게 단순히 텍스트만 바뀌는 경우는 아래처럼 one line if statement로 쓰는게 좋을 것 같습니다
  return (
    <div className="App">
      {!toggle ? <Timer count={count} /> : null}
      <button onClick={handleToggle}>
        {toggle ? "Toggled" : "Click to toggle"}
      </button>
    </div>
  );
}
```