6. State로 상태관리하기

다중 선택	react 입문

State(상태)

- 어떤 사물이 현재 가지고 있는 모양을 정의
- 변화할 수 있는 동적인 값
- 하나의 컴포넌트에 여러 개의 State 만들기 가능

```
import "./App.css";
import { useState } from "react";

function App() {
     const [state, setState] = useState(0); // 이렇게 쓰는 것이 
  // const state = useState(초기값 또는 비우기);
  console.log(state); // [undefined, ƒ]

  return <></>;
}

export default App;
```

- console.log(state); // [undefined(초기값? 또는 현재값), ƒ]
  - 첫 번째 인수 : state의 현재 값
  - 두 번째 인수 : state를 변화시키는 함수(= 상태 변화 함수)

사용 예시

```
import "./App.css";
import { useState } from "react";
```

```jsx
function App() {
  const [count, setCount] = useState(0);
  const [light, setLight] = useState("OFF");

  return (
    <>
      <div>
      {/* ON/OFF 버튼 클릭 시 조건에 맞게 ON/OFF 변경 */}
        <h1>{light}</h1>
        <button
          onClick={() => {
            setLight(light === "ON" ? "OFF" : "ON");
          }}
        >
          {light === "ON" ? "끄기" : "켜기"}
        </button>
      </div>
      <div>
        {/* 버튼을 누를 때마다 h1 안의 숫자 +1 */}
        <h1>{count}</h1>
        <button
          onClick={() => {
            setCount(count + 1);
          }}
        >
          버튼
        </button>
      </div>
    </>
  );
}

export default App;
```

const, let 대신 state를 사용하는 이유

- state 값이 변경되었을 때만 리렌더링 됨(즉, 변수의 값을 많이 수정해도 렌더링이 다시 일어나지 않음)

> 💡 변화하는 가변적인 값을 화면에 렌더링 해야 한다면 `State` 이용!