Vježba 1 — useState

Zadatak

Napravi komponentu **CounterWithLimit**.
Komponenta treba:

1. Prikazivati trenutni broj klikova (`count`).
2. Imati dva gumba: **+1** i **Reset**.
3. Ako broj klikova dosegne **10**, prikaži poruku:
   “Dostigli ste maksimalan broj klikova!” i onemogući gumb **+1**.

Vježba 2 — useEffect

Zadatak

Napravi komponentu **Clock**.
Komponenta treba:

1. Prikazivati trenutno vrijeme (sat:minute:sekunda).
2. Vrijeme se treba automatski ažurirati svake sekunde pomoću `setInterval`.
3. Kada se komponenta ukloni sa stranice (unmount), interval se mora očistiti.

Vježba 3 — useContext

Zadatak

Napravi jednostavan **ThemeSwitcher** koristeći `useContext`.
Komponenta treba:

1. Imati dva moda: **light** i **dark**.
2. Theme se treba spremiti u **context** i biti dostupan svim child komponentama.
3. Dodaj gumb koji mijenja theme.
4. Komponenta `Header` treba prikazivati trenutni theme koristeći `useContext`.

Vježba 4 — useRef

Zadatak

Napravi komponentu **FocusInput** koristeći `useRef`.
Komponenta treba:

1. Imati `<input>` polje u koje korisnik može pisati tekst.
2. Imati gumb **Fokusiraj input** koji automatski stavlja kursor u input polje.
3. State se **ne smije koristiti** za fokus — koristimo samo `useRef`.

Vježba 5 — useReducer

Zadatak

Napravi komponentu **TodoList** koristeći `useReducer`.
Komponenta treba:

1. Imati input polje za unos zadatka.
2. Gumb **Dodaj** koji dodaje novi zadatak u listu.
3. Svaki zadatak prikazati u listi s gumbom **Ukloni**.
4. State treba biti upravljan preko `useReducer`, a ne `useState`.

Vježba 6 — useCallback

Zadatak

Napravi komponentu **SearchList** koristeći `useCallback`.
Komponenta treba:

1. Imati niz imena (`["Ana", "Marko", "Ivana", "Petar", "Lana"]`).

2. Imati input polje za unos pojma za pretragu.
3. Funkcija za filtriranje imena (`filterNames`) treba biti spremljena pomoću `useCallback`.
4. Dodaj child komponentu **NameList** koja prima funkciju `filterNames` kao prop i prikazuje filtrirana imena.
5. U konzoli prati kada se child renderira.

Vježba 7 — useMemo

Zadatak

Napravi komponentu **ExpensiveCalculator** koristeći `useMemo`.
Komponenta treba:

1. Imati input za unos broja.
2. Funkcija treba računati **faktorijel** tog broja (npr. `5! = 5×4×3×2×1 = 120`).
3. Funkcija je “skupa” — neka sadrži umjetno usporenje (petlja).
4. Optimiziraj računanje pomoću `useMemo`, tako da se faktorijel ponovo izračuna samo ako se broj promijeni.
5. Dodaj još jedan state (`theme`) s gumbom za promjenu između `light` i `dark`. Promjena teme **ne smije** ponovo pokretati skupi izračun.

```
// Funkcija za faktorijel (s umjetnim usporenjem)
function factorial(n) {
  console.log("Računam faktorijel...");
  let result = 1;
  for (let i = 1; i <= n; i++) {
    // umjetno usporenje
    for (let j = 0; j < 1e6; j++) {}
    result *= i;
  }
  return result;
}
```

Vježba 8 — Custom Hooks

Zadatak

Napravi vlastiti hook **useLocalStorage**.
Komponenta treba:

1. Spremiti vrijednost (npr. korisničko ime) u `localStorage`.
2. Pročitati vrijednost iz `localStorage` prilikom učitavanja stranice.
3. Ažurirati i state i `localStorage` kada se promijeni input.