UNIVERSIDADE DE FORTALEZA
CURSO DE ANÁLISE E DESENVOLVIMENTO DE SISTEMAS DISCIPLINA: PROGRAMAÇÃO FUNCIONAL

INTEGRANTES DA EQUIPE:
Sarah Maria Freitas Paz Aragão (Todas as funções) - 2213885

Requisitos

Requisitos Funcionais:

- Lista de idades pré definida: Está no estado ou state do React “Idades”
- Função de duplicar a idade: utiliza a função “mutiplicarIdade”
- Função de triplicar a idade utiliza a função a variável “multiplicarPor3”, que chama a função “criarMultiplicador”

Requisitos Não Funcionais:

- Organização do código com nomes claros e comentários para facilitar a manutenção do código

Casos de teste

ChatGPT utilizado, não consegui realizar os testes

Aqui estão alguns casos de teste simples para o componente React que você forneceu, usando **Jest** e **React Testing Library**:

1. Teste para renderizar a lista inicial de idades

Esse teste verifica se a lista de idades inicial é renderizada corretamente.

jsx Copiar código

```jsx
import { render, screen } from '@testing-library/react';
import App from './App';

test('Renderiza a lista inicial de idades', () => {
  render(<App />);
  const items = screen.getAllByRole('listitem');
  expect(items.length).toBe(5);
  expect(items[0].textContent).toBe('12 de Idade');
  expect(items[4].textContent).toBe('54 de Idade');
});
```

2. Teste para multiplicar idades por 2

Esse teste verifica se ao clicar no botão "Multiplicar seu tempo de vida por 2", as idades são multiplicadas corretamente.

jsx Copiar código

```jsx
import { render, screen, fireEvent } from '@testing-library/react';
import App from './App';

test('Multiplica as idades por 2 ao clicar no botão', () => {
  render(<App />);

  const button = screen.getByText(/Multiplicar seu tempo de vida por 2/i);
  fireEvent.click(button);

  const items = screen.getAllByRole('listitem');
  expect(items[0].textContent).toBe('24 de Idade');
  expect(items[4].textContent).toBe('108 de Idade');
});
```

Conceitos utilizados no código

- Função Lambda

```
const multiplicarIdade= () => {
  const novaIdade = idades.map((numero) => numero * 2); // Função lambda
  setIdades(novaIdade);
};
```

- List comprehesion

```
// Simulando uma "list comprehension" com .map para exibir as idades
const listaIdades = idades.map(function(numero) {
  return <li key={numero}>{numero} de Idade</li>;
});
```

- Closure e função de alta ordem

```
// Função de alta ordem para criar um multiplicador
function criarMultiplicador(fator: number) {
  return function(numero: number) {
    return numero * fator; // A função lembra do valor de fator (closure)
  };
}
```