

PRIMEIRA AVALIAÇÃO PARCIAL - D

DESENVOLVIMENTO DE SOFTWARE PARA WEB PROJETO DE INTERFACES WEB

2023.1 - PROF. JEFFERSON DE CARVALHO SILVA

UFC - QUIXADÁ

Questão 01 (2.5pts)	Questão 02 (2.5pts)	Questão 03 (2.5pts)	Questão 04 (2.5pts)

Obs.: Para cada questão de implementação, use pelo menos 3 comentários para explicar o que está ocorrendo no código. **Código não comentado não será corrigido!**
Evite comentários óbvios.

Questão 01 - Em um mesmo arquivo chamado **Questao01.jsx**, crie dois componentes JSX:

- O componente **Questao01X** (crie usando função arrow)
- O componente **Questao01Y** (crie usando função arrow)

O componente **Questao01X** deve chamar, internamente em seu JSX, o componente **Questao01Y**. Além disso, o componente **Questao01X** irá passar, via props, para o componente **Questao01Y**, o seguinte objeto:

```
const correntistas= [  
  { nome: "Sicrano", aplic: {pp: 856.4, rf: 50.4, cc: 34.0} },  
  { nome: "Beltrano", aplic: {pp: 0.0, rf: 700.67, cc: 800} },  
  { nome: "Fulano", aplic: {pp: 5000.00, rf: 0.0, cc: 500} }  
]
```

onde aplic = aplicações, pp = poupança e cc = conta corrente.

O componente **Questao01Y**, ao receber o objeto **correntistas**, deverá calcular a média das aplicações $\rightarrow (pp+rf+cc)/3$. As médias serão armazenadas em um vetor de três posições que deve ser retornado ao componente **Questao01X** (comunicação filho \rightarrow pai). Cada posição do vetor de médias corresponderá a um correntista. Por exemplo, a média armazenada na posição 0 do vetor corresponde a média aritmética das aplicações do correntista "Sicrano" e assim por diante.

De posse do vetor de médias, o componente **Questao01X** deve imprimir, em seu JSX, **apenas** o nome do correntista com a maior média de aplicações

Dicas:

- Em **Questao01X**, cria uma função para ser repassada a **Questao01Y** a qual inicializa uma variável de estado em **Questao01X** chamada de “medias”, com a médias calculadas por **Questao01Y** (comunicação filho → pai).
- Em **Questao01X**, crie uma função que retorne um JSX para renderizar o correntista de maior média .

Questão 02 - Crie um arquivo chamado **Questao02.jsx** que deverá, em seu JSX, renderizar a imagem de um Pokemon de sua preferência. Por exemplo, para renderizar o Pikachu, use o caminho:

<https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/25.png>

Este caminho irá mostrar o Pikachu **de frente**.

Insira um botão, em Questao02.jsx, que ao ser pressionado, deverá mudar a imagem frontal para imagem de costas. Eis o seu caminho de costas:

<https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/back/25.png>

Ao pressionar o botão novamente, a imagem volta a ficar de frente e assim por diante, alternando entre frente e costas.

Você deve criar uma variável de estado “**flag**”, booleana, para ficar alternando entre **true** e **false**. No entanto, o mais adequado seria o useRef (opcional, pois foi matéria extra).

Questão 03 - Seja o seguinte código a ser executado no console do navegador:

```
fetch("https://restcountries.com/v3.1/region/europe?fields=capital,population")
  .then(
    (response) => {
      return response.json()
    }
  )
  .then(
    (data) => {
      console.table(data.map(
        (country) => country.capital[0] + "-" + country.population
      ))
    }
  )
  .catch(error => console.log(error))
```

Ele retorna a saída, impressa em console.table:

[

```
{ "capital": ["Dublin"], "population": 4994724 },  
{ "capital": ["Nicosia"], "population": 1207361 },  
  ...  
{ "capital": ["Madrid"], "population": 47351567 }  
]
```

Ou seja, um vetor com vários objetos contendo duas propriedades: **capital**, do tipo vetor de string e **population**, do tipo numérico.

Crie um componente chamado **Questao03.jsx** que imprima na tela as capitais do países com a população (**population**) maior que a média das populações. Use **async-await**.

OBSERVAÇÃO - NO CASO DA INTERNET DO CAMPUS ESTAR OFFLINE:

Caso a internet não esteja disponível, você deverá então criar uma **Promise**, em **Questao03**, cujo **resolve** deve retornar o seguinte Array:

```
[  
  { "capital": ["Dublin"], "population": 4994724 },  
  { "capital": ["Nicosia"], "population": 1207361 },  
  { "capital": ["Madrid"], "population": 47351567 }  
]
```

Então, no **useEffect** de **Questao03**, você irá usar a **Promise** criada com o **async-await**. O resto do exercício será o mesmo.

Questão 04 - Crie um arquivo **Questao04.txt** e:

Explique o que são os hooks e a sua importância para o desenvolvimento React. Escolha dois hooks e os explique com exemplos (não há necessidade de executar o código)

ATENÇÃO

Crie um último arquivo chamado **PONTOS.TXT**. Neste arquivo diga quantos pontos você acha que deve ganhar em cada questão, por exemplo:

- 1 - 2,0
- 2 - 0,0
- 3 - 1,5
- 4 - 2,5

Pense da seguinte forma: cada funcionalidade não implementada (ou não respondida), **retire 0,5 pts** da questão.