

Trabalho 02 – Relações Genéricas entre Componentes, Funções Internas, Comunicação Filho → Pai

Para cada exercício abaixo, crie uma pasta, em src, diferente no mesmo projeto (questao01, questao02 e questao03). Cada pasta deve conter os componentes que devem ser implementados em cada questão. No arquivo “index.js”, você pode ir chamando a questão que quiser para poder testar, sempre deixando comentado (`/* */`) o código de chamada das outras questões. **TESTE** todas as questões.

1 -) Usando o que foi visto em sala de aula, sobre Relações Genéricas entre Componentes (aula01-slide 30), implemente o que se pede:

- Crie os arquivos `Curso.jsx`, `Turma.jsx` e `Estudante.jsx`;
- Em `Curso.jsx`, você deverá imprimir em uma tag `<h2>` o nome do curso passado via props. Logo abaixo da tag `<h2>`, você deverá incluir um `{props.children}` para receber componentes do tipo `<Turma>`. Sendo assim, genericamente, o componente `<Curso>` poderá receber quantas `<Turma>` quiser (veja slide 31).
- Em `Turma.jsx`, você deverá imprimir em uma tag `<h2>` o nome da turma passado via props. Você deverá também chamar os `<Estudante>` logo abaixo da tag `<h2>` fazendo uso do `React.Children` e passando como parâmetro para cada filho o nome da Turma (veja slide 36).
- Em `Estudante.jsx`, você deverá imprimir, simplesmente:
`Estudante {props.nome} do Curso {props.curso}`.
- Um exemplo de chamada destes componentes em `index.js` (desconsidere o resto do código):

```
<Curso curso='Sistemas de Informação'>
  <Turma turma='Fundamentos de Programação'>
    <Estudante nome='Fulano de Tal'>
    <Estudante nome='Beltrano de Tal'>
  </Turma>
  <Turma turma='Introdução a Banco de Dados'>
    <Estudante nome='Fulano de Tal'>
    <Estudante nome='Beltrano de Tal'>
    <Estudante nome='Sicrano de Tal'>
  </Turma>
</Curso>
```

2 -) Implemente a mesma versão do código da Questão 1, só que usando apenas componentes `HARDCODED`, ou seja, o que foi visto no slide 27. Você não deverá usar nem `{props.children}` e nem o `{React.Children}`.

3 -) Usando o que foi visto em sala de aula sobre a comunicação entre componentes Filho → Pai e Funções Internas, faça o que se pede:

- Crie os arquivos `IMCFilho.jsx` e `IMCPai.jsx`;
- Em `IMCFilho.jsx`, implemente:
 - Uma função “`calculaIMC`” que recebe como parâmetro o peso (kg) e altura (m) e retorna o IMC. O IMC é calculado como $\text{peso}/(\text{altura}*\text{altura})$. Veja slide 40.
 - No **return** de `IMCFilho`, retorne uma `<div>` com um botão interno que chame a função “`notificarIMCPai`”, passada como parâmetro via props (veja slide 44).
 - “`notificarIMCPai`” deverá receber como parâmetro o resultado do IMC calculado pelo função “`calculaIMC`”.

- “calculaIMC”, implementado em IMCFilho.jsx, irá receber altura e peso de IMCPai, via props.
- Em IMCPai.jsx, implemente:
 - Assim como no slide 45, passe para o filho, via props, a função “notificarIMCPai”. Essa função deve apenas exibir um alerta com a mensagem do IMC calculado pelo filho.
 - Ainda, passe para IMCFilho, via props, a altura e o peso, para que ele possa calcular o IMC em sua função interna.

ATENÇÃO: ENVIAR APENAS A PASTA src DO SEU PROJETO. O TRABALHO PODERÁ SER FEITO EM DUPLAS. NESTE CASO, ENVIE O NOME DA DUPLA EM UM COMENTÁRIO DENTRO DE index.js. PROJETOS ENVIADOS COM OUTRAS PASTAS ALÉM DA src SERÃO DESCONSIDERADOS.

OS TRABALHOS DEVEM SER ENVIADOS PELO SIPPA (TR2). VERIFIQUEM SE O ARQUIVO FOI ENVIADO CORRETAMENTE. NÃO HAVERÁ ABERTURA POSTERIOR DO SIPPA.

OS TRABALHOS DEVEM SER ENVIADOS ATÉ AS 14:00 DE 18/03.