Documentação trabalho de estruturas de dados 01

Integrantes:

Luan Campos Kuhlmann

Lucas Rodrigues de Lima Sousa

Link do projeto no github:

https://github.com/LuanKuhlmann/trabalho-1-estrutura-de-dados

Processo:

No processo de construção do código fonte para o trabalho, visando uma quantidade não muito grande de dados a serem analisados, ultilizamos o bubble sort pela facilidade e simplicidade da estrutura.

Utilizamos um arquivo .js para a construção do código backend e um html para fácil utilização dos recursos construídos, assim como um aquivo css para estilizar e facilitar o entendimento do usuário.

Para manter um código limpo, visamos diminuir os processos repetidos e implementar funções que possibilitam a reutilização para fácil entendimento futuro, assim como comentários explicando o processo e o objetivo da função.

Codigo backend .js:

```
const alunos = [];

function subimite(e) {
  e.preventDefault();
  const aluno = {};

  const nomeInput = document.getElementById('nome');
  const raInput = document.getElementById('ra');
  const idadeInput = document.getElementById('idade');
  let sexoInput = null;
  const mediaInput = document.getElementById('media');

  // loop para confirmar a seleção no radio
  const radioButtons = document.querySelectorAll('input[name="gender"]');
  for (const radioButton of radioButtons) {
    if (radioButton.checked) {
```

```
      sexoInput =
document.querySelector(`label[for="${radioButton.id}"]`).getAttribute('fo
r');
      break;
    }
  }

  // validar e armazenar
  try {
    const nome = nomeInput.value.trim();
    if (!nome.match(/^[a-zA-Z\s]*$/) || nome.length < 1) {
      throw new Error('Nome do aluno deve conter apenas letras.');
    }
    aluno.NOME = nome;

    const ra = raInput.value.trim();
    if (isNaN(ra) || ra.length !== 13) {
      throw new Error('RA deve conter apenas números e possuir 13
dígitos.');
    }
    aluno.RA = ra;

    const idade = parseInt(idadeInput.value, 10);
    if (isNaN(idade) || idade < 1 || idade > 100) {
      throw new Error('Idade informada precisa ser válida.');
    }
    aluno.IDADE = idade;

    if (sexoInput !== 'M' && sexoInput !== 'F' && sexoInput !== 'O') {
      throw new Error('Gênero do aluno deve ser "M" para masculino, "F"
para feminino ou "O" para outros.');
    }
    aluno.SEXO = sexoInput;

    const media = parseFloat(mediaInput.value);
    if (isNaN(media) || media < 0 || media > 10) {
      throw new Error('Média informada precisa ser válida e estar entre 0
e 10.');
    }
    aluno.MEDIA = media;

    aluno.RESULTADO = aluno.MEDIA >= 6 ? 'Aprovado' : 'Reprovado';

    // adicionar aluno e limpar o formulario
    alunos.push(aluno);

    show(alunos);

    nomeInput.value = '';
```

```
    raInput.value = '';
    idadeInput.value = '';
    // limpar a seleção de genero
    for (const radioButton of radioButtons) {
      radioButton.checked = false;
    }
    mediaInput.value = '';

    alert('Aluno cadastrado com sucesso!');
  } catch (error) {
    alert(error.message);
  }
}


// Ordena os alunos em ordem crescente
function ordemCrescente() {
  try {
    if (alunos.length === 0) {
      throw new Error('A lista está vazia.');
    }

    const alunosTemp = [...alunos];
    const n = alunosTemp.length;

    // Variavel para gerenciar o loop
    let swap;

    // Aplicando o sort
    do {
      swap = false;
      for (let i = 0; i < n - 1; i++) {

        // Checando os nomes e ordenando corretamente
        if (alunosTemp[i].NOME > alunosTemp[i + 1].NOME) {
          const temp = alunosTemp[i];
          alunosTemp[i] = alunosTemp[i + 1];
          alunosTemp[i + 1] = temp;
          swap = true;
        }
      }
    } while (swap);

    show(alunosTemp);

  } catch (error) {
    alert(error.message);
  }
}
```

```
// Função para orderar os alunos em ordem decrescente o RA
function ordemDecrescenteRA() {
  try {
    if (alunos.length === 0) {
      throw new Error('A lista está vazia.');
    };

    const alunosTemp = [...alunos];
    const n = alunos.length;

    // Variavel para gerenciar o loop
    let swap;

    // Aplicando o sort
    do {
      swap = false;
      for (let i = 0; i < n - 1; i++) {

        // Checando os RA e ordenando corretamente
        if (alunosTemp[i].RA < alunosTemp[i + 1].RA) {
          const temp = alunosTemp[i];
          alunosTemp[i] = alunosTemp[i + 1];
          alunosTemp[i + 1] = temp;
          swap = true;
        }
      }
    } while (swap);

    show(alunosTemp);

  } catch (error) {
    alert(error.message);
  }
}

// Função para ordenar os alunos aprovados em ordem crescente de nomes
function ordemCrescenteAprovados() {
  try {
    if (alunos.length === 0) {
      throw new Error('A lista está vazia.');
    }

    // Constroi uma array apenas com os alunos aprovados
    const alunosAprovados = alunos.filter((aluno) => aluno.RESULTADO ===
'Aprovado');
    if (alunosAprovados.length === 0) {
      throw new Error('Nenhum aluno aprovado na lista.');
    }
```

```
    // Variavel para gerenciar o loop
    let swap;

    // Aplicando o sort
    do {
      swap = false;
      for (let i = 0; i < alunosAprovados.length - 1; i++) {

        // Checando os nomes e ordenando corretamente
        if (alunosAprovados[i].NOME > alunosAprovados[i + 1].NOME) {
          const temp = alunosAprovados[i];
          alunosAprovados[i] = alunosAprovados[i + 1];
          alunosAprovados[i + 1] = temp;
          swap = true;
        }
      }
    } while (swap);

    show(alunosAprovados);

  } catch (error) {
    alert(error.message);
  }
}

// Função para mostrar os dados no formulario
function show(array) {
  const tbody = document.getElementById('listBody');
    if (tbody) {
    tbody.innerHTML = array.map(user => {
      return `<tr>
                <td>${user.NOME}</td>
                <td>${user.RA}</td>
                <td>${user.IDADE}</td>
                <td>${user.SEXO}</td>
                <td>${user.MEDIA}</td>
                <td>${user.RESULTADO}</td>
              </tr>`;
    }).join('');
  }
}

window.addEventListener('load', () => {
  document.getElementById('form').addEventListener('submit', subimite);
});
```

Codigo frontend .html:

```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link rel="stylesheet" type="text/css" href="./TAREFA.css">
    <script src="TAREFA.js"></script>
</head>
<body>
    <form class="register" id="form">
        <div class="label">
            <div>Nome</div>
            <div>
                <input type="Text" class="input" id="nome" />
            </div>
        </div>

        <div class="label">
            <div>RA</div>
            <div>
                <input class="input" id="ra" />
            </div>
        </div>

        <div class="label">
            <div>Idade</div>
            <div>
                <input class="input" id="idade" />
            </div>
        </div>

        <div class="label">
            <div>Media</div>
            <div>
                <input class="input" id="media" />
            </div>
        </div>
        <div class="label">
            <div class="gender-input">
                <input type="radio" id="M" name="gender">
                <label for="M">Masculino</label>
            </div>
            <div class="gender-input">
                <input type="radio" id="F" name="gender">
                <label for="F">Feminino</label>
            </div>
            <div class="gender-input">
                <input type="radio" id="O" name="gender">
                <label for="O">Outros</label>
```

```
            </div>
        </div>
        <button class="buttonRes">Registrar</button>
    </form>
    <div class="list" id="list">
        <table>
            <thead>
                <tr>
                    <div class="container">
                        <button class="button"
onclick="ordemCrescente()">Nome Crescente</button>
                        <button class="button"
onclick="ordemDecrescenteRA()">RA Decrescente</button>
                        <button class="button"
onclick="ordemCrescenteAprovados()">Aprovados</button>
                    </div>
                    <th>Nome</th>
                    <th>RA</th>
                    <th>Idade</th>
                    <th>Genêro</th>
                    <th>Média</th>
                    <th>Situação</th>
                </tr>
            </thead>
            <tbody id="listBody"></tbody>
        </table>
    </div>
</body>
</html>
```

Codigo para estilização do frontend .css:

```
body {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 14px;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #e0e0e0;
    display: block;
    justify-content: center;
    align-items: center;
    margin-left: 5px;
    gap: 10px;
}

.register {
    display: flex;
```

```
    flex-wrap: wrap;
    justify-content: space-between;
    padding: 1rem 0;
    margin-top: 1%;
    display: flex;
    width: calc(100% - 10px); /* 30% da largura da tela menos 10px de
distância */
    height: calc(30% - 10px); /* 100% da altura da tela menos 10px de
distância */
    height: 30vh; /* 30% da altura da tela */
    background-color:  #f4f3f3; /* Cor de fundo da primeira div */
    border-radius: 10px;
}

.list {
    margin-top: 1%;
    width: calc(100% - 10px); /* 70% da largura da tela menos 10px de
distância */
    height: calc(70% - 10px); /* 100% da altura da tela menos 10px de
distância */
    background-color: #f4f3f3; /* Cor de fundo da segunda div */
    border-radius: 10px;
    float: center;
    border-collapse: collapse;
}

.list th,
.list td{
    font-size: 14px;
    width: 20%;
    height: 20%;
    padding: 10px ;
    border: solid 1px #ccc;
    float: center;
}

.buttonRes{
    margin-top: 5%;
    width: 10%;
    height: 30%;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    background-color: #07a;
    color: #ffff;
}

.button{
    border: none;
```

```
    border-radius: 5px;
    cursor: pointer;
    background-color: #07a;
    color: #ffff;
    margin: 3px;
    padding: 15px;
}

.button:hover{
    opacity: 0.8;
}


.container{
    display: grid;
    grid-auto-flow: column; /* Coloca os botões em uma linha */
    justify-content: center; /* Centraliza os botões horizontalmente */
    gap: 30%;
}


.label{
    font-size: 14px;
    border-radius: 4px;
    margin: 3px;
    padding: 15px;
}

.input{
    border: solid 1px #ccc;
    border-radius: 4px;
    padding: 5px;
}
```