Maria Eduarda Coelho da Silva - Exercicios sobre Threads

Exercício Teórico

1 - O que é uma thread e como ela difere de um processo?

Processos: Mais isolamento, porém mais lento e comunicação mais complexa.Usado em aplicações independentes e isoladas.

- **Programa em execução:** São programas independentes, cada um com seu próprio espaço de memória e recursos.
- **Isolamento:** Oferecem alto isolamento, uma falha em um não afeta outros.
- **Custoso:** Mais lentos para criar e alternar entre eles (troca de contexto).
- **Comunicação:** Comunicação entre processos é mais complexa (usa mecanismos como pipes, sockets, etc.).
- **Exemplo:** Vários aplicativos abertos ao mesmo tempo (navegador, editor de texto, etc.).

Threads: Mais rápidos, mais leves, comunicação mais fácil, porém menos isolamento. Usado em tarefas dentro da mesma aplicação que precisam trabalhar juntas.

- **Unidades de execução:** São partes de um processo, compartilham a memória e recursos do processo.
- **Compartilhamento:** Compartilham a mesma memória, comunicação é mais fácil e rápida.
- **Leve:** Mais rápidas para criar e alternar entre elas (troca de contexto).
- **Paralelismo:** Permitem executar tarefas em paralelo (em sistemas multi-core).
- **Risco:** Uma thread com problema pode afetar outras threads do mesmo processo.
- **Exemplo:** Um navegador que baixa arquivos enquanto você navega, um editor de texto que salva automaticamente em segundo plano.

2 - Quais são as principais vantagens do uso de threads?

Melhora o desempenho: Permite executar tarefas em paralelo (se você tiver vários núcleos de processador), acelerando a execução de programas.

Melhor uso dos recursos: Threads compartilham a mesma memória e recursos do processo, economizando memória e facilitando a comunicação entre as tarefas.

Respostas mais rápidas: Permite que um programa continue respondendo mesmo quando está executando tarefas demoradas em segundo plano.

Criação e troca rápidas: Threads são mais rápidas para criar e alternar entre elas do que processos.

3 - Quais os tipos de Thread´s?

1. Threads de Kernel:
Gerenciadas pelo Sistema Operacional;
Criação e troca mais custosas;
Usam todos os núcleos do processador.

2. Threads de Usuário:
Gerenciadas pelo programa;
Criação e troca mais leves;
O sistema não as vê individualmente.

Exercício Prático

1 - Criar um programa simples que utiliza threads para realizar duas tarefas simultaneamente: contar de 1 a 10 e imprimir letras de 'A' a 'J'.

```
async function countNumbersThread() {
    for (let i = 1; i <= 10; i++) {
        console.log("NÚMERO [THREAD 1]:", i);
            await new Promise(resolve => setTimeout(resolve, 10)); //
Simula um pequeno atraso
    }
}

async function printLettersThread() {
    for (let i = 65; i <= 74; i++) {
        console.log("LETRA [THREAD 2]:", String.fromCharCode(i));
            await new Promise(resolve => setTimeout(resolve, 10)); //
Simula um pequeno atraso
    }
}

async function main() {
    console.log("Iniciando tarefas...")
    // Executa as duas funções de forma concorrente usando Promise.all
    await Promise.all([countNumbersThread(), printLettersThread()]);
    console.log("Tarefas concluídas")
}

main();
```

2 - Modifique o Programa inserindo mais duas a sua escolha.

```
async function countNumbersThread1() {
```

```
    for (let i = 1; i <= 10; i++) {
        console.log("NÚMERO [THREAD 1]:", i);
        await new Promise(resolve => setTimeout(resolve, 10));
    }
}

async function countNumbersThread2() {
    for (let i = 90; i <= 100; i++) {
        console.log("NÚMERO [THREAD 2]:", i);
        await new Promise(resolve => setTimeout(resolve, 10));
    }
}



async function printLettersThread1() {
    for (let i = 65; i <= 74; i++) {
        console.log("LETRA [THREAD 3]:", String.fromCharCode(i));
        await new Promise(resolve => setTimeout(resolve, 10));
    }
}

async function printLettersThread2() {
    for (let i = 75; i <= 84; i++) {
        console.log("LETRA [THREAD 4]:", String.fromCharCode(i));
        await new Promise(resolve => setTimeout(resolve, 10));
    }
}

async function main() {
    console.log("Iniciando tarefas...")
      await  Promise.all([countNumbersThread1(), printLettersThread1(),
countNumbersThread2(), printLettersThread2()]);
    console.log("Tarefas concluídas")
}

main();
```