**Relatório Técnico – Problema do Produtor-Consumidor**

**1. Introdução**

O problema do Produtor-Consumidor é um clássico da computação concorrente. Ele consiste em coordenar processos que compartilham um buffer limitado:  
- O Produtor insere itens no buffer.  
- O Consumidor retira itens do buffer.  
- É necessário garantir sincronização para evitar condições de corrida e respeitar os limites do buffer.  
  
Neste trabalho, implementamos a solução em C++ com threads, mutex e variáveis de condição, além de uma versão sequencial para comparação.

**2. Objetivos**

- Implementar o problema do Produtor-Consumidor em duas abordagens:  
 1. Sequencial – execução alternada do produtor e consumidor.  
 2. Paralela – uso de threads para execução concorrente.  
- Comparar desempenho entre as abordagens (tempo total de execução).  
- Demonstrar o uso de mecanismos de sincronização.

**3. Metodologia**

Linguagem: C++17  
Compilador: g++ (MSYS2 MinGW-w64, versão 15.2.0)  
Execução:  
 ./compare.exe seq  
 ./compare.exe par  
 ./compare.exe both  
Estruturas utilizadas:  
- std::mutex para exclusão mútua no acesso ao buffer.  
- std::condition\_variable para controlar espera quando buffer está cheio ou vazio.  
- std::deque como buffer circular simples.

**4. Resultados**

Saída – Execução Sequencial (inserir print do terminal mostrando os logs do modo seq).

Saída – Execução Paralela (inserir print do terminal mostrando os logs do modo par).

Saída – Comparação (both) (inserir print do terminal mostrando os tempos finais).



**5. Discussão**

- Sequencial: O consumidor só começa após o produtor terminar um lote, resultando em tempo maior.  
- Paralelo: O produtor e o consumidor trabalham de forma simultânea, aproveitando a sobreposição de execução.  
- Sincronização: O uso de mutex e condition\_variable garante que o produtor não insira além da capacidade e que o consumidor não tente retirar itens de um buffer vazio.

**6. Conclusão**

O experimento comprovou que a execução paralela é mais eficiente que a sequencial, pois as operações de produção e consumo acontecem em sobreposição. A implementação demonstrou na prática conceitos de concorrência, sincronização e uso de threads, fundamentais para disciplinas de Sistemas Operacionais e Programação Concorrente.





