1. As partes em **negrito** devem constar no relatório
2. Faça *download* do arquivo para a prática
3. Descompacte o arquivo
4. Observe o programa joinEx.py. Para entender o que ele faz, procure na documentação API da biblioteca de threads para Python3 os tipos e as funções que você não conhece:
    1. A classe Thread
    2. thread.start()
    3. thread.join()
5. **Dê uma descrição do que faz cada um desses tipos ou funções.**
6. **Rode o programa. O que ele faz? Explique cada linha do resultado da saída do programa.**
7. **Por que as threads do programa não terminam na ordem em que são criadas?**
8. **O que acontece se você retirar as linhas 21 e 22 do arquivo joinEx.py. O programa precisa dessas linhas para terminar com o resultado esperado?**
9. **O que a função sys.exit() faz nesse caso?**
10. Observe o programa simpleMutexEx.py. Para entender o que ele faz, procure na documentação API da biblioteca de threads para Python3 os tipos e as funções que você não conhece:
    1. classe Lock
    2. Lock.acquire()
    3. Lock.release()
11. **Dê uma descrição do que faz cada um desses tipos ou funções.**
12. **Rode o programa. O que ele faz?**
13. **Retire as diretivas de sincronização "acquire()" e "release()". O que acontece com o programa? Você saberia explicar esse comportamento?**
14. **Desenvolva um programa em Python que modela o problema do produtor-consumidor usando threads e as diretivas de sincronização de Python. Seu programa deve ter as seguintes características:**
    1. O tamanho do buffer deve estar definido em uma constante declarada logo após os imports. Eu quero poder mudar o tamanho do buffer e o seu programa deve continuar funcionando corretamente.
    2. Sempre que o produtor produzir um item X (que deve ser um inteiro), ele deve escrever na tela "Produced X"
    3. Sempre que o consumidor consumir um item X (que deve ser um inteiro), ele deve escrever na tela "Consumed X"
    4. Faça uma função de delay para podermos ver o que acontece no terminal. Caso contrário a tela vai rolar muito rápido e não teremos como ver.
    5. Se o buffer estiver vazio e for a vez do consumidor executar, ele deve dormir
    6. Se o buffer estiver cheio e for a vez do produtor executar, ele deve dormir
    7. Verifique na documentação as diretivas de wait() e notifyAll() da classe Condition
15. **Adicione seu código corrigido ao relatório.**