

## Sistemas Operativos

(ano letivo 2019-20)

“

**E-fólio B |** Instruções para a realização do E-fólio



Este enunciado constitui o elemento de avaliação designado por “e-fólio B” no âmbito da avaliação contínua e tem a cotação total de 5 valores. A sua resolução deve ser entregue até às 23h55 do dia 18 de maio pelos alunos que escolheram a modalidade de avaliação contínua.

A resolução deve ser entregue através de um único ficheiro compactado .zip, que:

- (i) contém os ficheiros .c que constituem o código dos programas, prontos a serem compilados;
- (ii) contém um ficheiro de nome relatorio.pdf com um relatório simples e sucinto com informações solicitadas e/ou complementares de modo a permitir uma fácil compreensão do trabalho realizado. É desnecessário incluir uma listagem integral do código.
- (iii) O nome do ficheiro .zip a entregar deve seguir a seguinte convenção para o seu nome,

“NúmeroAluno-PrimeiroNome-Apelido-21111-efB.zip”

Por exemplo, um aluno com número 327555 e nome Paulo ... Costa, deverá dar o seguinte nome ao ficheiro, “327555-Paulo-Costa-21111-efB.zip”

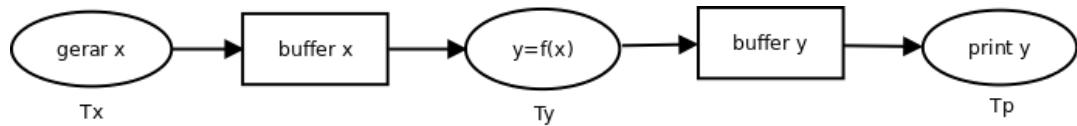
O ficheiro deve ser única e exclusivamente entregue através do recurso “E-fólio B” disponibilizado na plataforma (Nota: apenas é visível para os alunos inscritos em avaliação contínua), não sendo aceites trabalhos enviados por outras vias, como por exemplo por e-mail.

Esta é uma prova de avaliação **individual** e não “um trabalho de grupo”. A sua resolução deve provir unicamente do conhecimento adquirido e trabalho original desenvolvido pelo próprio aluno. Os alunos deverão saber distinguir claramente entre discutir os conteúdos abordados na unidade curricular (permitido) e discutir a resolução específica do e-fólio (não permitido).

No caso de dúvidas de interpretação do enunciado, utilize o fórum de avaliação para pedidos de esclarecimento.

# I

- 1. [5]** Escreva um programa multitarefa em linguagem C padrão e segundo a norma POSIX, de nome `mTEX.c`, constituído pela tarefa principal, por 1 tarefa designada `Tx`, por `nt` tarefas designadas `Ty` e 1 tarefa designada `Tp`, num total de `nt+3` tarefas. O objetivo do programa é calcular o valor da função exponencial  $y=f(x)=e^x$  para `n` valores reais de `x`. A estratégia a utilizar é do tipo produtor / consumidor como mostra a figura seguinte,



- A tarefa `Tx` cada vez que acede ao buffer `x` enche o buffer independentemente de ele se encontrar completamente vazio ou não, até no total ter gerado `n` valores.
  - As `nt` tarefas `Ty` lêm (removem) de cada vez  $k \leq kmax$  valores do buffer `x`, calculam os respectivos valores  $y=f(x)$  e escrevem (inserem) os  $k$  valores no buffer `y`, após o que repetem a operação.
  - A tarefa `Tp` cada vez que acede ao buffer `y` esvazia o buffer independentemente de ele se encontrar completamente cheio ou não, e imprime os valores com o formato "%. $5f$ \n".
  - O programa `mTEX` recebe obrigatoriamente 4 argumentos na linha de comandos,
- ```
>> ./mTEX nt n dimbuf kmax
```
- `nt` é o nº de tarefas `Ty`, com  $nt \geq 1$ .
  - `n` é o nº de valores gerados pela tarefa `Tx`, com  $n \geq nt$ .
  - `dimbuf` é a dimensão dos buffers `x` e `y`, com  $1 \leq dimbuf \leq n$ .
  - `kmax` é o nº de valores máximo retirado do buffer `x` pela tarefa `Tx`, com  $1 \leq kmax \leq dimbuf$ .
  - O programa `mTEX` deve testar se o número de argumentos dado na linha de comandos é correto e se os seus valores são válidos. Em caso de erro o programa deve emitir uma mensagem e terminar.
  - No início do programa, a tarefa principal (main) deve imprimir uma mensagem do tipo "Cálculo de NNN valores de  $e^x$  com NNN tarefas, dimbuf=NNN e kmax=NNN".
  - Os valores reais de `x` gerados pela tarefa `Tx` devem ter distribuição uniforme no intervalo  $[-1, 1]$  sendo gerados recorrendo à função `rand()` inicializada previamente com a semente 223.
  - Quando criadas, as `nt` tarefas `Ty` devem receber respetivamente como argumento o seu número de ordem, entre 0 e  $nt-1$ . Quando terminam, devem indicar à tarefa principal quantos valores de `y` calcularam.

- Especial atenção deve ser dada ao caso de uma tarefa tentar escrever num buffer cheio ou ler de um buffer vazio.
- O acesso aos buffers é feito em exclusão mútua e com espera ativa (pooling), ou seja, se a tarefa não puder escrever/ler porque o buffer está cheio/vazio, deve tentar novamente até conseguir. Antes de tentar novamente deve invocar a função `sched_yield()` que liberta o processador (ver man page) dando oportunidade às outras tarefas de serem escalonadas para execução.
- No caso das tarefas Tx e Tp, além do que foi referido no parágrafo anterior, após conseguirem aceder ao buffer devem também invocar a função `sched_yield()` para dar oportunidade às outras tarefas de serem escalonadas para execução e acederem ao buffer.
- Conceba uma estratégia para que as tarefas Ty e Tp saibam quando devem terminar.
- Antes de terminar, a tarefa principal, que deve ser a última a terminar, deve imprimir um relatório com o número de valores de y que cada tarefa calculou .
- Pondere quais as funções da biblioteca `pthread` que vai utilizar no programa e consulte as respetivas man pages para se informar dos detalhes de funcionamento de cada uma. Pondere também cuidadosamente quais os recursos e as estruturas de dados manipuladas pelas tarefas e que requeiram exclusão mútua no seu acesso para o bom funcionamento do programa.
- Indique no relatório as variáveis, mutexes e os troços de código correspondentes a regiões críticas do programa e justifique a sua existência/necessidade.

- O programa deve estar identificado com um cabeçalho similar ao seguinte,

```
/*
** UC: 21111 - Sistemas Operativos
** e-fólio B 2019-20 (mtex.c)
**
** Aluno: 327555 - Paulo Costa
*/
```

## **Critérios de correção:**

- Programa desenvolvido difere significativamente das especificações e instruções do enunciado => 0 valores.
- Programa não compila ou produz avisos (warnnings) com `gcc -Wall` => 0 valores.
- Código do programa não está correta e uniformemente indentado de modo a permitir a sua leitura fácil => 0 valores
- Programa não está comentado => 0 valores. Os comentários no programa elucidam questões relevantes do código locais ao comentário.
- Funcionalidade do programa de acordo com o pedido, estrutura, nível de simplicidade e qualidade do código (até 65%)
- Relatório. Explique o como e porquê relativamente às opções e soluções técnicas que tomou para a estrutura e funcionamento do programa (até 35%)

**Nota ética:** Nunca é de mais referir que o código a apresentar como solução para este e-fólio deve ser 100% original do aluno. A probabilidade de duas pessoas que efetivamente não comunicaram entre si, apresentarem programas “quase iguais” é considerada nula. Isto é válido para qualquer par de alunos (cópia), assim como entre um aluno e qualquer outra pessoa, em particular através da Internet (cópia/plágio), onde existem inúmeras soluções e código para os mais variados problemas, em sites, fóruns, blogs, etc.

Cumpra estritamente as normas de realização individual, como se estivesse num exame com consulta, onde pode consultar a documentação mas não pode falar com ninguém.

FIM