



IPL
escola superior
de tecnologia e gestão
instituto politécnico
de leiria

Licenciatura em
Engenharia Informática
UC de Programação Avançada
2º ano – Engenharia Informática
Regime diurno/pós-laboral
Ano letivo 2024/2025 - 1º Semestre

Teste Prático – Enunciado A

2024.11.09 / 09h30

Prova com consulta

Duração: 90 minutos

Nome completo: _____

N.º de estudante: _____

Regime: [] Diurno

[] Pós-laboral

IMPORTANTE

É expressamente proibido o recurso à Internet durante a prova. Qualquer utilização não autorizada da Internet leva à anulação da prova e ao reportar da situação às autoridades competentes. O mesmo sucede com outros tipos de tentativa de fraude.

• **Preparação da máquina virtual:**

- 1) Crie, no ambiente de trabalho, a pasta **EI_PA**
- 2) Seguidamente, copie a máquina virtual da UC para a pasta EI_PA do ambiente de trabalho.
 - a) Caso tenha a máquina virtual da UC numa PEN pode copiá-la para o ambiente de trabalho.
 - b) Caso contrário, o ficheiro 7z da máquina virtual encontra-se na pasta C:\VM, pelo que pode copiar o ficheiro 7Z (não mover!) para a pasta EI_PA do ambiente de trabalho e descompactá-lo.

• **Antes de iniciar a prova:**

- Execute os seguintes comandos:

```
cd; mkdir -p ~/ProvaP/R_NUMERO/
```

(em que **R** deve ser substituído pela letra **D** se for do regime diurno e **N** se for aluno do regime pós-laboral e **NUMERO** deve ser substituído pelo seu número ESTG);

- Para garantir que o seu diretório de trabalho seja o correto, faça:

```
cd ~/ProvaP/R_NUMERO/
```

• **Após ter terminado a prova:**

- Deverá proceder à criação de um arquivo TAR, fazendo uso do seguinte comando:

```
cd ~/ProvaP/R_NUMERO/; tar cvf ProvaP_YYYYMMDD_R_NUMERO.tar *
```

(em que YYYYMMDD corresponde à data corrente, e.g., 20241109, e R_NUMERO obedece ao formato acima indicado);

- Verifique que o arquivo “.tar” que criou não está vazio, através da execução de:

```
tar tvf ProvaP_YYYYMMDD_R_NUMERO.tar
```

- Entregue o arquivo “.tar” através da plataforma moodle, no espaço reservado para o efeito. Em caso de dúvidas, pergunte ao professor;
- Informe o professor para este validar a receção dos seus ficheiros.

Pergunta [20 valores]

(Escreva as suas respostas a esta pergunta no diretório "**~/ProvaP/R_NUMERO/Pergunta**". Deve indicar o seu nome completo e número de estudante IPLeiria no ficheiro **README.txt** a ser criado no diretório)

NOTA 1: não é permitida a chamada a comandos externos através da função *system* ou de outra com funcionalidade similar (e.g., */bin/sh*).

NOTA 2: a solução deve ser implementada com recurso aos ficheiros/makefile do arquivo EmptyProject-Template_v3.6.zip

NOTA 3: código entregue que **não compile** através do utilitário *make* e do respetivo *makefile* na máquina virtual da UC leva à atribuição da classificação de **0 (zero) valores** à resposta.

Recorrendo à linguagem C, à norma Pthreads e ao utilitário *gengetopt*, elabore o programa *WriteByThreads* que visa escrever uma string na saída padrão (*stdout*). Cada letra da string deve ser escrita pela thread correspondente a essa letra do alfabeto.

Concretamente, o programa deve receber uma string de entrada através da linha de comandos utilizando o parâmetro **-s/--str <palavra>**, e deverá criar 26 threads, uma para cada letra do alfabeto (e.g., uma thread para a letra a/A, outra thread para a letra b/B, etc.). Assim, cada thread será responsável por escrever na saída padrão somente a letra que lhe corresponde, independentemente de estar em minúscula ou maiúscula.

Por exemplo, se a string de entrada for "Paz", a thread responsável pela letra 'P' deve escrever, na saída padrão, o primeiro caractere ('P'), a thread responsável pela letra 'a' deve, após a thread 'P', escrever o segundo caractere ('a'), com a thread 'z' a terminar a escrita com a letra 'z'. A aplicação deve garantir que as letras da palavra são escritas na ordem correta, sincronizando as threads para o efeito. Para além da escrita do caractere que lhe corresponde, a thread deve ainda indicar o seu ID, acessível através da função *pthread_self*. Após a escrita da string completa, a aplicação deve terminar.

A aplicação só deve aceitar strings que contenham os caracteres do alfabeto ('a' a 'z') sejam elas maiúsculas ou minúsculas. Caso a string fornecida através da linha de comandos tenha algum caractere que não pertença ao conjunto acima indicado (por exemplo, à, é, Ç, ã, etc.), a aplicação deve terminar com uma apropriada mensagem de erro. O parâmetro da linha de comando deve ser obrigatoriamente tratado através do utilitário *gengetopt*.

A aplicação deve ainda emitir informação apropriada no canal de saída padrão seguindo o modelo apresentado no Exemplo 1.

Exemplo 1

```
./WriteByThreads -s "Estudara"
```

```
E [thread 'e'] (ID=140533435631168)
s [thread 's'] (ID=140533250901568)
t [thread 't'] (ID=140533242508864)
u [thread 'u'] (ID=140533234116160)
d [thread 'd'] (ID=140533444023872)
a [thread 'a'] (ID=140533469201984)
r [thread 'r'] (ID=140533259294272)
a [thread 'a'] (ID=140533469201984)
```

Exemplo 2

```
./WriteByThreads -s "avançada"
```

```
ERROR: non-regular string 'avançada' (first failed char is char #5)
```

Exemplo 3

```
./WriteByThreads
```

```
./WriteByThreads: '--str' ('-s') option required
```