



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 B

2024.11.09 / 11h00

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.

• **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*, escreva o programa **search** que permite verificar se um determinado valor inteiro consta de um vetor e, se sim, em que posição se encontra. O programa deve receber como parâmetros **i)** o tamanho do vetor; **ii)** o valor a procurar e **iii)** o número de *threads* a utilizar no processo de procura, sendo que o tamanho do vetor indicado deve ser um múltiplo do número de *threads*.

O programa deve começar por gerar o vetor de números aleatórios, sendo que esses valores devem ser únicos (i.e., o vetor não pode ter valores repetidos) e estar compreendidos no intervalo inteiro [0, Max], com Max a corresponder a três vezes o valor indicado para o tamanho do vetor (e.g., se tamanho = 10 então Max = 30). Seguidamente deve proceder à criação das *threads*, atribuindo a cada uma um identificador inteiro compreendido entre 1 e o número de *threads*.

Cada *thread* deve processar um bloco de valores consecutivos, sendo o tamanho de cada bloco igual ao quociente do número de elementos do vetor pelo número de *threads* (i.e., $\text{num_elms_vetor} / \text{num_threads}$). A *thread* que encontrar o valor a procurar deve escrever na saída padrão (stdout) a sua identificação, o valor e o índice do vetor onde o valor foi encontrado. Todas as *threads* devem terminar imediatamente após o valor ser encontrado. Caso o valor não seja encontrado, essa informação deve ser mostrada no final do processo de procura.

A aplicação **search** deve suportar os seguintes parâmetros da linha de comandos, devendo os mesmos serem implementados com recurso ao utilitário *gengetopt*:

- **-s / --size (inteiro):** Tamanho do vetor. Este parâmetro é opcional. Caso o utilizador não defina o seu valor, o parâmetro deve assumir o valor 10. A aplicação deve validar se este valor é maior ou igual a 10. Caso contrário, termina com uma mensagem no canal padrão de erro;
- **-t/--target (inteiro):** Valor a procurar. Este parâmetro é obrigatório.
- **-n/--numthreads (inteiro):** Número de *threads* a utilizar no processo de procura. Este parâmetro é obrigatório. A aplicação deve validar se este valor é maior ou igual a 2. Caso contrário, termina com uma mensagem no canal padrão de erro.

Considere os seguintes exemplos de execução.

Exemplo 1 \$./search -s 10 -t 1 -n 5 [3 2 9 12 5 8 1 27 4 6] Thread 4 found target '1' at index 6	Exemplo 2 \$./search -s 10 -t 7 -n 2 [15 3 27 9 6 4 5 2 11 8] Target '7' was not found.
Exemplo 3 \$./search -s 6 -t 1 -n 5 [ERROR]: vetor size (6) is not a multiple of the number of threads (5)	Exemplo 4 \$./search -s 10 -t 1 --numthreads 1 [ERROR]: number of threads should be larger than 1.

Exemplo de código C para gerar valores aleatórios. Sugere-se a consulta de man rand_r	
Função rand_r : gerar números aleatórios Fora das threads	Função rand_r : gerar números aleatórios Nas threads
unsigned int r_state = time(NULL) ^ getpid(); int value1 = rand_r(&r_state); int value2 = rand_r(&r_state);	unsigned int r_state = time(NULL) ^ getpid() ^ pthread_self(); int value1 = rand_r(&r_state); int value2 = rand_r(&r_state);