Nome:_____

Nº ESTG: _____

Aval. Periódica - EI/Prog. Avançada

Teste Prático – 2022.10.29 /B

Licenciatura em
Engenharia Informática
UC de Programação Avançada
2º ano – Engenharia Informática
Regime diurno/pós-laboral
Ano letivo 2022/2023 - 1ºSemestre

Teste Prático – Enunciado B

2022.10.29 / 11h00' | Prova com consulta | **Duração:** 80 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.

- **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., 20221029, 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.
NOTA 2: a solução deve ser implementada com recurso aos ficheiros do diretório **EmptyProject-Template.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, implemente o programa **seat_reservations** que simula a reserva de lugares para um espetáculo. Os lugares na mini-sala de espetáculos são representados num vetor de 16 posições. Cada célula do vetor tem o valor 0 se o lugar correspondente ainda não estiver reservado e o valor *x* se o lugar foi reservado pela *thread* identificada com o **id *x***. O identificador "id x" da *thread* corresponde à ordem de criação da thread, com a 1ª thread a ser criada a receber o identificador 1, a 2ª thread a ser criada a receber o identificador 2, e assim sucessivamente.

O programa deve criar **num_of_threads** *threads* e cada *thread* deve tentar fazer **num_of_reservations** reservas individuais, sendo cada reserva para um lugar específico selecionado aleatoriamente. Caso uma *thread* gere um valor correspondente a um lugar já reservado por outra *thread*, a reserva não é realizada. No total, devem ocorrer **num_of_threads X num_of_reservations** tentativas de reserva. O programa deve permitir que uma *thread* A escreva na posição ***i*** do vetor, e simultaneamente uma outra *thread* B escreva na posição ***j***, desde que *i* seja diferente de *j*, ou seja, nesse caso a *thread* A não deve esperar pela *thread* B e vice-versa.

Antes de terminar, o programa deve escrever no canal de saída padrão o estado de cada lugar (0 – "not reserved"; ***thread x*** se a reserva foi efetuada pela *thread* com o **id *x***).

A aplicação **seat_reservations** deverá aceitar o seguinte parâmetro da linha de comandos:

-T/--num_of_threads <int>: número de *threads* a criar. Caso não seja especificado, deve ser considerado o valor 10. O valor deve ser maior ou igual a 1. Parâmetro opcional.

-R/--num_of_reservations <int>: número de reservas a realizar por cada *thread*. Caso não seja especificado, deve ser considerado o valor 4. O valor deve ser maior ou igual a 1. Parâmetro opcional.

Nota: Poderá usar a função *rand_r* para gerar os números aleatórios, conforme ilustrado no exemplo seguinte.

```
void *thread_func(void *arg) {
      unsigned int rand_state = time(NULL) ^ getpid() ^ pthread_self();
      int value1 = rand_r(&rand_state);
      int value2 = rand_r(&rand_state);
      return NULL;
}
```

Considere os seguintes exemplos.

Exemplo nº 1

$./seat_reservations -T 10 -R 2

```
Reservations:
Seat  1: not reserved
Seat  2: thread 5
Seat  3: not reserved
Seat  4: thread 6
Seat  5: thread 10
Seat  6: not reserved
Seat  7: thread 1
Seat  8: not reserved
Seat  9: not reserved
Seat 10: thread 7
Seat 11: thread 5
Seat 12: thread 2
Seat 13: thread 3
Seat 14: not reserved
Seat 15: thread 1
Seat 16: not reserved
```

Nota: no exemplo, as *thread* 4, 8 e 9 não conseguiram realizar nenhuma reserva.

Exemplo nº 2

$./seat_reservations -T -1 -R 50

```
num_of_threads must be > 0.
```

Exemplo nº 3

$./seat_reservations -T 50 -R -1

```
num_of_reservations must be > 0.
```