

Prova Escrita (sem consulta) – Enunciado B

2017-11-04

Duração: 50 min

Nome Completo: _____ N.º aluno: _____

Todas as perguntas devem ser resolvidas no enunciado!

[20 valores]

- Considere que todas as perguntas deste grupo são independentes.
- **Todas as perguntas respondidas incorretamente ou de forma ambígua descontam 25% da cotação da pergunta.**
- Selecione a resposta mais completa para cada uma das seguintes questões.
- Prova sem consulta.
- É expressamente proibido o uso de telemóveis ou de qualquer outro dispositivo eletrónico.

Tabela de respostas

Escreva, de forma legível, no retângulo reservado para o efeito, a letra da opção que considera a resposta certa. Caso não pretenda responder à pergunta, escreva “X” no meio do retângulo.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Estudante																				
Professor																				

1. [1 valor] *Threads* de processos distintos...

- partilham o segmento de *heap*
- partilham o segmento de pilha
- partilham o segmento de dados
- nenhuma das anteriores

2. [1 valor] Considere o código abaixo listado, tendo em atenção que `g_global` é uma variável global...

```
int g_global = 0x000A;
void func(int add){
    g_global += add;
}
```

- o código é thread-safe
- quando executado em ambiente concorrente, o código apresenta uma falha do tipo *bohrbug*
- a variável `g_global` é iniciada com um valor equivalente a zero
- nenhuma das anteriores

3. [1 valor] A existência do diretório /proc/997 num

sistema operativo Linux significa...

- a) que existem 997 processos ativos no sistema
- b) que existe o processo com PID 997
- c) que existe o processo com o PPID 997
- d) nenhuma das anteriores

4. [1 valor] A função `execlp` (protótipo: `int execlp(const char *file, const char *arg, ...);`)...

- a) possibilita a substituição da imagem do processo chamante pela imagem do ficheiro especificado no parâmetro `file`
- b) recorre ao caminho definido na variável do ambiente `PATH` para tentar localizar o ficheiro indicado pelo parâmetro `file`
- c) a lista de argumentos a serem passados para a execução do parâmetro `file` é terminada por `NULL`
- d) todas as anteriores

5. [1 valor] Considere o seguinte código fonte que compila sem avisos nem erros num sistema Lunbuntu 16.04

```
#include <stdio.h>
#include <unistd.h>
int main(void) {
    fork();
    for(int i=0;i<4;i++){
        if( fork() == 0 ){
            printf("Processo PID=%d\n", getpid());
        }
    }
    return 0;
}
```

Caso a execução decorra de forma normal (sem erros), quantas linhas são escritas para a saída padrão?

- a) 15
- b) 16
- c) 31
- d) Nenhuma das anteriores

6. [1 valor] A função `on_exit` tem o seguinte protótipo:

`int on_exit(void (*ptr)(int , void *), void *arg);`. O parâmetro `ptr` corresponde...

- a) a um ponteiro para função que devolve `void*` e recebe um inteiro e um ponteiro para `void*`
- b) a um ponteiro para zona de memória não tipada, isto é, do tipo `void*`
- c) a um ponteiro para função que não devolve nada (`void`) e que recebe como parâmetro um inteiro e um ponteiro para zona de memória não tipada
- d) nenhuma das anteriores

7. [1 valor] A função `mkstemp` (protótipo: `int mkstemp(char *template);`)...

- a) cria um ficheiro temporário de forma segura, devolvendo um descritor para o ficheiro aberto
- b) cria um diretório temporário cujo nome é iniciado pela string indicada através do parâmetro `template`
- c) cria uma *thread* temporária passando à *thread* a zona de memória apontada pelo parâmetro `template`
- d) nenhuma das anteriores

8. [1 valor] Na norma pthread, uma variável de condição...

- a) pode ser manipulada através das funções pthread_cond_wait, pthread_cond_timedwait, pthread_cond_signal e pthread_cond_broadcast
- b) é empregue para garantir a exclusão mútua no acesso a uma variável partilhada
- c) apenas deve ser acedida em exclusão mútua quando se faz uso das funções pthread_cond_wait e pthread_cond_timedwait
- d) nenhuma das anteriores

9. [1 valor] Considere o código em linguagem C da função func:

```
double func(double radius){  
    double b = 3.14 * radius * 2.0;  
    return b;  
}
```

A função func é...

- a) reentrante
 - b) não reentrante
 - c) recursiva
 - d) nenhuma das anteriores
10. [1 valor] Considere que A, B, C e D são threads de um mesmo processo. Considere ainda que M1 e M2 são dois mutexes e que lock(M1) significa *operação de lock no mutex M1* e unlock(M1) representa *operação de unlock sobre o mutex M1*...

thread A	thread B	thread C	thread D
lock(M2)	lock(M2)	lock(M2)	lock(M2)
lock(M1)	lock(M1)	lock(M1)	lock(M1)
(...)	(...)	(...)	(...)
unlock(M1)	unlock(M1)	unlock(M1)	unlock(M1)
unlock(M2)	unlock(M2)	unlock(M2)	unlock(M2)

Considerando somente a sequência de operações descrita na tabela e que não ocorre nenhuma terminação inesperada de threads (i.e., não há “crashes”), poderá a sequência de operações originar um *deadlock*?

- a) Sim, dado que as threads operam os mutexes por ordem diferente
- b) Não, pois todas as threads estão a usar os mesmos mutexes pela mesma ordem
- c) Sim, porque todas as threads estão a libertar (unlock) os mutexes pela ordem inversa de que foram tomados (lock)
- d) Nenhuma das anteriores

11. [1 valor] Na linguagem Java, um método declarado com a palavra-chave `synchronized`...

- a) é um método que solicita a data corrente a um servidor NTP
- b) é executado em exclusão mútua
- c) requer o uso explícito das primitivas `lock/unlock` pelo programador(a)
- d) nenhuma das anteriores

12. [1 valor] Considere o seguinte código em linguagem C:

```
short A = 1 << 3;
```

Sabendo que `sizeof(A)` é 2, o valor, na base decimal, atribuído à variável A é...

- a) 1
- b) 8
- c) 16
- d) 4

13. [1 valor] Num sistema Linux em que `sizeof(short)` é igual a dois, considere a seguinte declaração de variável: `int short A`. Qual das seguintes atribuições está correta?

- a) `A=011;`
- b) `A=0xFFFF;`
- c) `A= 3242;`
- d) Todas as anteriores

14. [1 valor] O conjunto de bits `0100001100100001` quando interpretado como uma variável inteira sem sinal de 16 bits corresponde a ...

- a) 65537 (base 10)
- b) Nenhuma das outras
- c) `0x4431` (base hexadecimal)
- d) `041441` (base octal)

15. [1 valor] O sistema operativo Unix recorre a um inteiro de 32 bits com sinal para representar o número de segundos decorridos desde o EPOCH (1 de janeiro de 1970 00h00 GMT). Ocorrerá um transbordo quando o número de segundos decorridos desde o EPOCH...

- a) ultrapassar o valor $2^{32}-1$
- b) ultrapassar o valor $2^{31}-1$
- c) ultrapassar o valor 2^{32}
- d) ultrapassar o valor 2^{31}

16. [1 valor] Considere o seguinte código em linguagem C:

```
uint16_t A = 0xFFFF;
A = A & ~(1 << 2);
```

O que se pode dizer sobre o valor da variável A após a execução do código?

- a) A variável A fica com o valor 0xFFFB
- b) O 3º bit menos significativo de A fica a zero
- c) O bit mais significativo de A fica a um
- d) Todas as anteriores

17. [1 valor] Considere o seguinte código C:

```
uint8_t A = (1 << 2) | (1 << 4) | (1 << 6);
```

O que se pode dizer sobre o valor da variável A após a execução do código?

- a) Nenhuma das outras
- b) A variável A fica com o valor 0x84
- c) A variável A fica com o valor 0x54
- d) A variável A fica com o valor 0x34

18. [1 valor] Para se obter a listagem das chamadas efetuadas durante a execução de um processo às funções existentes em bibliotecas ...

- a) executa-se o comando ltrace
- b) executa-se o comando ps
- c) executa-se o comando strace
- d) nenhuma das anteriores

19. [1 valor] Caso se pretenda executar num programa em linguagem C, a linha de comando "ls /proc | wc -l", deve-se...

- a) Criar um processo filho através da função *fork* e nesse processo filho chamar uma função da família exec indicando como primeiro parâmetro a string "ls /proc | wc -l"
- b) Chamar, no programa em linguagem C, a função system da seguinte forma `system("ls /proc | wc -l");`
- c) Chamar diretamente a função `exec1p` (protótipo: `int exec1p(const char *file, const char *arg, ...);`) da seguinte forma `exec1p("ls /proc | wc -l", NULL)`
- d) Todas as anteriores

20. [1 valor] Considerando o protótipo da chamada ao sistema write:

```
ssize_t write(int fd, const void *buf, size_t count);
```

A seguinte chamada à função write: `write(2, "PA", strlen("PA"));...`

- a) Efetua a escrita da cadeia de caracteres "PA" para o canal de saída padrão
- b) Não é válida, pois deveria ser empregue `sizeof("PA")` e não `strlen("PA")`
- c) Efetua a escrita da cadeia de caracteres "PA" para o canal de erro padrão
- d) Nenhuma das anteriores