

**U.C. 21111**

**Sistemas Operativos**

**18 de setembro de 2017**

### **INSTRUÇÕES**

- Leia estas instruções na totalidade antes de iniciar a resolução do teste.
- O enunciado do teste é constituído por 2 grupos de questões, tem 3 páginas e termina com a palavra FIM.
- Se o seu exemplar não estiver completo ou nele se verificar qualquer outra deficiência, por favor dirija-se ao professor vigilante.
- O teste deve ser resolvido na sua totalidade em folhas de respostas.
- Nas respostas, tenha a preocupação de utilizar uma letra legível.
- Todas as respostas devem ser escritas unicamente com caneta azul ou preta.
- O teste é SEM CONSULTA. Todos os elementos necessários à resolução são fornecidos no enunciado.
- Não é permitido utilizar máquina de calcular.
- As cotações são indicadas por grupo e nas próprias questões.
- Nas questões de escrita de programas, a sua correção terá em conta critérios de proficiência e compreensibilidade do código (legibilidade, indentação, estrutura, comentários e explicação geral).
- O não cumprimento das instruções implica a anulação das respetivas questões.
- O tempo de realização do teste é de 150 minutos.

### Grupo I [12 valores]

- 1.1. [1.2] Comente a seguinte frase: "O sistema operativo é um computador virtual".
- 1.2. [1.2] Em que consiste a técnica de spooling ?
- 1.3. [1.2] Explique o conceito de pseudoparalelismo e relacione-o com o conceito de processo.
- 1.4. [1.2] Caracterize o conceito de processo multitarefa.
- 1.5. [1.2] Explique em que consiste o escalonamento de um processo e descreva quatro situações que possam levar o SO a efectuá-lo.
- 1.6. [1.2] Explique como pode evitar a ocorrência de situações de impasse (deadlocks) através de uma abordagem à condição de espera circular. Indique quais as limitações desta estratégia.
- 1.7. [1.2] O conceito de espaço de endereçamento virtual unidimensional permite atribuir um espaço próprio de endereçamento para cada processo ou tarefa (thread) ? Justifique.
- 1.8. [1.2] Considere um sistema com memória virtual mononível. O espaço de endereçamento virtual é de 20 bits e a dimensão da página virtual é de 2KB. Quantas entradas tem a Tabela de Páginas ?
- 1.9. [1.2] Explique o significado de directoria de trabalho (ou corrente) e relacione-o com o conceito de "pathname"relativa.
- 1.10. [1.2] Qual o nome do circuito de controlo que permite a comunicação directa entre um dispositivo de I/O e a memória RAM ?

### Grupo II [8 valores]

- 2.1. [3] Escreva um programa em linguagem C padrão que crie um processo filho que com recurso à função `execv()` execute o comando `ps -o pid,user`. O processo pai deve esperar que o processo filho termine.
- 2.2. [5] Escreva um programa multitarefa em linguagem C segundo a norma POSIX que transfira por blocos os elementos de um vector `int x[1000]` para um vector `int y[1000]`. Os vectores constituem variáveis globais ao programa e admite-se que foram devidamente inicializados.

A tarefa principal deve criar 10 subtarefas em que cada uma em paralelo (ou em pseudo-paralelismo) copia um bloco de 20 elementos do vector `x[]` para o vector `y[]`. O bloco origem em `x[]` tem início no elemento de menor índice ainda não transferido. O bloco destino em `y[]` tem início no elemento de menor índice ainda não preenchido. A tarefa principal deve esperar que todas as subtarefas terminem.

Nota 1: com a estratégia indicada, a ordem dos elementos em `y[]` será diferente (por blocos) da ordem em `x[]`, dependendo da ordem do escalonamento das tarefas.

Nota 2: planeie cuidadosamente como é dividido o trabalho entre as sub-tarefas e como é efectuada a sincronização e a comunicação da informação necessária à resolução do problema entre todas as tarefas.

## Formulário

```
#include <stdlib.h>
int system(char *string);

#include <sys/types.h>
#include <unistd.h>
pid_t fork(void);
pid_t getpid(void);
pid_t getppid(void);

#include <unistd.h>
unsigned int sleep(unsigned int seconds);
extern char **environ;
int execl(char *path, char *arg, ...);
int execlp(char *file, char *arg, ...);
int execle(char *path, char *arg, ..., char *envp[]);
int execv(char *path, char *argv[]);
int execvp(char *file, char *argv[]);
int execve(char *path, char *argv [], char *envp[]);

#include <sys/types.h>
#include <sys/wait.h>
pid_t wait(int *status);

#include <pthread.h>
int pthread_create(pthread_t *thread, pthread_attr_t *attr,
    void *(*start_routine)(void*), void *arg);
int pthread_attr_init(pthread_attr_t *attr);
int pthread_attr_setdetachstate(pthread_attr_t *attr, int detachstate);
#define PTHREAD_CREATE_DETACHED
#define PTHREAD_CREATE_JOINABLE
int pthread_join(pthread_t thread, void **value_ptr);
int pthread_mutex_init(pthread_mutex_t *mutex,
    pthread_mutexattr_t * attr);
int pthread_mutex_lock(pthread_mutex_t *mutex);
int pthread_mutex_unlock(pthread_mutex_t *mutex);
int pthread_mutex_destroy(pthread_mutex_t *mutex);
```

FIM