ISEL – DEETC

Licenciatura em Engenharia Informática e de Computadores

Sistemas Operativos

2ª Época – SV 2009/2010 (14/07/2010) Duração: 2:30 H

Justifique todas as suas respostas.

I

1. **[1,5 Valores]** Uma forma de limitar o acesso a um recurso, num ambiente de múltiplas tarefas, consiste na utilização do mecanismo de sincronismo *spinlock*. Discuta as vantagens/desvantagens na sua adopção no contexto de arquitecturas monoprocessador e multiprocessador.

2. **[1,5 Valores]** Nos sistemas operativos actuais existem, geralmente, os conceitos de tarefa e processo. Assim, o programador tem a possibilidade de modular as suas aplicações com base em múltiplos processos ou em múltiplas tarefas. Aponte as principais vantagens/desvantagens pela adopção de cada uma das alternativas.

3. **[1,5 Valores]** Um programador afirmou que no contexto de uma aplicação com interface gráfica, desenvolvida com recurso à API Win 32, deve-se evitar que a tarefa associada ao tratamento de eventos realize operações de Entrada/Saída (Input/Output). Comente, justificando convenientemente a sua resposta, a afirmação do programador.

II

1. **[1,5 Valores]** Compare os mecanismos de sincronismo, da `Win32 API`, `Event` e `Waitable Timer`.

2. **[2 Valores]** No ambiente Windows a utilização da memória virtual pode ser realizada através das funções `VirtualAlloc` ou `HeapAlloc`. Descreva estas duas funções e indique as situações em que a sua utilização é mais adequada.

3. **[1,5 Valores]** Descreva, pormenorizadamente, como é tratada uma excepção estruturada (`SEH`) na situação em que todos os filtros de excepção devolvem `EXCEPTION_CONTINUE_SEARCH`.

III

1. **[2 Valores]** Considere uma arquitectura com suporte à gestão de memória virtual através de paginação com uma estrutura de quatro níveis. Sabendo que a dimensão das páginas é de 4KB, cada tabela de suporte à paginação dos 4 níveis ocupa uma página e que a dimensão das entradas das tabelas é de 8 Bytes indique:
   - O esboço da organização deste sistema de gestão de memória;
   - A dimensão do espaço de endereçamento virtual;
   - O número de páginas existentes no espaço de endereçamento virtual;
   - A dimensão, mínima e máxima, ocupada pelas estruturas de gestão de memória virtuais associadas a cada processo.

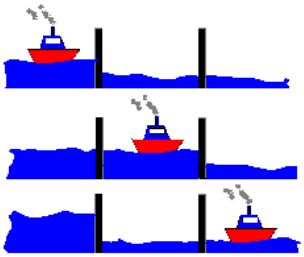
2. **[1,5 Valores]** Indique, justificando convenientemente as suas afirmações, as implicações na adopção de páginas de maior dimensão, por exemplo 8KB, na gestão de memória virtual baseada em paginação.

3. **[1,5 Valores]** Na gestão de memória virtual com paginação que tipos de fragmentação ocorrem na espaço de endereçamento virtual e no espaço de endereçamento físico? Considere as situações em que um programa requisita várias zonas de memória (páginas) com a função `VirtualAlloc` e que em algumas situações não utiliza, completamente, uma página.

IV

1. **[2,5 Valores]** A federação Portuguesa de futebol decidiu estabelecer uma seriação das preferências dos Portugueses pelos jogadores da selecção nacional presentes no mundial. Para o efeito realizou um conjunto de 15000 entrevistas em que cada entrevistado escolhe o seu jogador preferido através da indicação de um número entre 0 e 23, correspondendo 0 – Não respondeu e 1 a 23 – Voto no respectivo jogador. O resultado de cada entrevista encontra-se armazenado no *array* `int entrevistas[15000]`. Realize o programa para calcular o resultado final que deve criar 15 tarefas ficando cada uma delas responsável pelo processamento de 1000 entrevistas. O resultado final deve ficar armazenado no *array* `int resultadoFinal[23]`. Nota: considere na sua solução uma alternativa que permita tirar o maior partido possível da concorrência.

2. **[3 Valores]** Realize uma aplicação em ambiente Windows que permita efectuar a simulação do Canal do Panamá. Como sabe o canal liga dois oceanos distintos, através de um sistema composto por 3 "divisões" e 2 comportas, que no seu conjunto permitem anular o desnível dos dois oceanos. Na sua simulação admita que cada barco é implementado por uma tarefa, e lembre-se que deve minimizar ao máximo o enchimento e o vazamento das divisões. Assuma que podem existir barcos a efectuar a travessia nos dois sentidos e que todos os barcos que esperam para passar o podem fazer de uma só vez.



Nuno Oliveira e Carlos Gonçalves

ANEXO

```
BOOL WINAPI CreateProcess(
  __in          LPCTSTR lpApplicationName,
  __in_out      LPTSTR lpCommandLine,
  __in          LPSECURITY_ATTRIBUTES
                                lpProcessAttributes,
  __in          LPSECURITY_ATTRIBUTES
                                lpThreadAttributes,
  __in          BOOL bInheritHandles,
  __in          DWORD dwCreationFlags,
  __in          LPVOID lpEnvironment,
  __in          LPCTSTR lpCurrentDirectory,
  __in          LPSTARTUPINFO lpStartupInfo,
  __out         LPPROCESS_INFORMATION
                                lpProcessInformation
);

VOID WINAPI ExitProcess(
  __in          UINT uExitCode
);

BOOL WINAPI GetExitCodeProcess(
  __in          HANDLE hProcess,
  __out         LPDWORD lpExitCode
);

HANDLE WINAPI CreateThread(
  __in    LPSECURITY_ATTRIBUTES lpThreadAttributes,
  __in    SIZE_T dwStackSize,
  __in    LPTHREAD_START_ROUTINE lpStartAddress,
  __in    LPVOID lpParameter,
  __in    DWORD dwCreationFlags,
  __out   LPDWORD lpThreadId
);

VOID WINAPI ExitThread(
  __in          DWORD dwExitCode
);

BOOL WINAPI GetExitCodeThread(
  __in          HANDLE hThread,
  __out         LPDWORD lpExitCode
);

DWORD WINAPI WaitForSingleObject(
  __in          HANDLE hHandle,
  __in          DWORD dwMilliseconds
);

DWORD WINAPI WaitForMultipleObjects(
  __in          DWORD nCount,
  __in          const HANDLE* lpHandles,
  __in          BOOL bWaitAll,
  __in          DWORD dwMilliseconds
);

void WINAPI InitializeCriticalSection(
  __out        LPCRITICAL_SECTION lpCriticalSection
);

void WINAPI EnterCriticalSection(
  __in_out     LPCRITICAL_SECTION lpCriticalSection
);

void WINAPI LeaveCriticalSection(
  __in_out     LPCRITICAL_SECTION lpCriticalSection
);

void WINAPI DeleteCriticalSection(
  __in_out     LPCRITICAL_SECTION lpCriticalSection
);

HANDLE WINAPI CreateMutex(
  __in     LPSECURITY_ATTRIBUTES lpMutexAttributes,
  __in     BOOL bInitialOwner,
  __in     LPCTSTR lpName
);

BOOL WINAPI ReleaseMutex(
  __in          HANDLE hMutex
);

HANDLE WINAPI CreateSemaphore(
  __in LPSECURITY_ATTRIBUTES lpSemaphoreAttributes,
  __in LONG lInitialCount,
  __in LONG lMaximumCount,
  __in LPCTSTR lpName
);

BOOL WINAPI ReleaseSemaphore(
  __in          HANDLE hSemaphore,
  __in          LONG lReleaseCount,
  __out         LPLONG lpPreviousCount
);

HANDLE WINAPI CreateEvent(
  __in     LPSECURITY_ATTRIBUTES lpEventAttributes,
  __in     BOOL bManualReset,
  __in     BOOL bInitialState,
  __in     LPCTSTR lpName
);

BOOL WINAPI SetEvent(
  __in          HANDLE hEvent
);

BOOL WINAPI ResetEvent(
  __in          HANDLE hEvent
);

LONG __cdecl InterlockedIncrement(
  __in_out      LONG volatile* Addend
);


LONG __cdecl InterlockedDecrement(
  __in_out      LONG volatile* Addend
);

LONG __cdecl InterlockedExchange(
  __in_out      LONG volatile* Target,
  __in          LONG Value
);
```