UNIVERSIDADE FEDERAL DO PARANÁ
ENGENHARIA DE ELÉTRICA
TE355 -*Sistemas Operacionais Embarcados*
Segunda Prova

Nome: _____

Por favor, desligue seu celular. Não será permitido o uso de calculadoras. As questões podem ser respondidas a caneta ou lápis. Esta prova possui 8 questões, com um total de 120 pontos

1. Considere um sistema operacional que utiliza segmentação para a alocação de memória. Um processo possui a seguinte tabela de segmentos:

Segmento	Registrador Base	Tamanho
Código	CS=1000	2000
Dados	DS=4000	500
Pilha	SS=7000	300

10 (a) Explique como funciona a segmentação de memória e como ela é utilizada para possibilitar que o segmento seja alocado em qualquer posição na memória.

10 (b) Considerando o programa simples em linguagem C apresentado a seguir, explique como os segmentos de Código, Dados e Pilha são utilizados.

```
#include <stdio.h>

char x=1, y=2, z=3, w;

char soma(int a, int b, int c) {
    char r = a+b+c;
    return r;
}

void main() {
   w=soma(x, y, z);
}
```

a) A segmentação de memória permite que cada uma dessas áreas (código, dados, pilha) seja independente e possa ser alocada em qualquer posição na memória física, o que é possível porque o programa compilado contém referências para deslocamentos (*off-set*) em relação à base do segmento. Essa separação também facilita o uso eficiente da memória, já que os segmentos podem ser carregados ou movidos de forma independente. Por exemplo, três variáveis globais seriam criadas no segmento de código e acessadas como [0], [1], [2] (ou seja, valor apontado por DS+0, valor apontado por DS+1 e valor apontado por DS+2). O mesmo ocorre para os desvios condicionais ou incondicionais do programa, que estão no segmento de código e tambem com as variáveis locais e parâmetros colocados na pilha.

b) - SEGMENTO DE CÓDIGO (TEXT SEGMENT): Este segmento contém as instruções do programa, como as funções `main()` e `soma()`. Ele é armazenado em uma área de memória somente leitura para evitar modificações em tempo de execução.

No exemplo:

- As funções `main()` e `soma()` residem no segmento de código.

- SEGMENTO DE DADOS (DATA SEGMENT): Este segmento contém variáveis globais e variáveis estáticas. No exemplo:

- As variáveis globais `x`, `y`, `z` e `w` estão neste segmento:
  - `x=1`, `y=2`, `z=3` vão para a **área inicializada**.
  - `w` vai para a **área não inicializada** e é inicializada implicitamente com zero.

- SEGMENTO DA PILHA (STACK SEGMENT): Este segmento contém variáveis locais das funções, parâmetros e informações de controle de chamadas. A pilha cresce e diminui dinamicamente conforme as funções são chamadas e retornam.

No exemplo:

- A variável local `r` da função `soma()` e os parâmetros `a`, `b`, `c` estão na pilha.
- Cada vez que a função `soma()` é chamada, um novo quadro (frame) de pilha é criado para armazenar essas variáveis.

No caso específico da linguagem C, o valor de retorno é feito através de registradores (antigamente DX:AX, nos processadores atuais Intel de 64 bits no registrador RAX, já nos processadores ARM no registrador R0)

2. Considererando o conceito de bibliotecas estáticas ou dinâmicas, responda as seguintes perguntas:

10 (a) Quais são as vantagens de usar bibliotecas dinâmicas?

10 (b) Explique qual o benefício obtido com o uso de bibliotecas estáticas?

a) Uma biblioteca dinâmica é carregada na memória uma única vez, mesmo que vários programas a utilizem simultaneamente. O sistema operacional compartilha a mesma cópia da biblioteca entre os processos, **reduzindo o consumo de memória**. Vantagens adicionais incluem: atualizações podem ser feitas diretamente na biblioteca sem a necessidade de recompilar ou redistribuir os programas que dependem dela. Como as bibliotecas não estão embutidas no executável, os programas que dependem delas têm tempos de inicialização mais rápidos caso a biblioteca já esteja carregada em memória.

b) A principal vantagem é a **independência do Sistema**: O executável gerado é autossuficiente, pois contém todo o código necessário para sua execução, incluindo as funções da biblioteca. Isso elimina a necessidade de que a biblioteca esteja instalada no sistema onde o programa será executado, tornando o software mais portável. Execução Mais Rápida: Como as bibliotecas já estão incorporadas no executável, não há necessidade de carregá-las em tempo de execução, caso a bibliotaca compartilhana não esteja carregada em memória. Compatibilidade Garantida: O desenvolvedor tem controle total sobre a versão da biblioteca utilizada no momento da compilação. Facilidade de Distribuição: Distribuir um único executável com todas as dependências incluídas simplifica a entrega de software, especialmente em sistemas onde o gerenciamento de dependências pode ser complexo. Previsibilidade e Confiabilidade: O programa usa exatamente a versão da biblioteca que foi vinculada em tempo de compilação, evitando surpresas causadas por alterações ou atualizações de bibliotecas externas.

10 3. A implementação de memória virtual é uma "parceria" entre o hardware e o sistema operacional. Descreva quais recursos de hardware devem ser implementados para permitir a implementação eficiente da memória virtual. O hardware deve implementar:

- Tradução de endereços lógicos para físicos usando tabelas de páginas.
- Bits de controle nas tabelas de páginas: bit de presença, bit de modificação, bit de acesso (ajuda nos algoritmos de substituição de páginas).
- Detecta e sinaliza page faults através de interrupções, o Sistema Operacional deve implementar a rotina de tratamento correspondente.

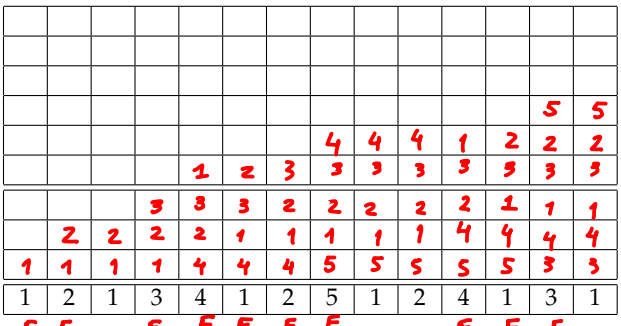
10 4. Considere o algoritmo de NPR (Newest Page Replacement, ou Substituição de Página Mais Nova). Este algoritmo substitui sempre a página mais recentemente carregada na memória, independentemente de ela ter sido acessada ou modificada. Se o acesso às páginas de memória fosse totalmente randômico, descreva se existe vantagem em usar o NPR ou o LRU (Least Recently Used) para reduzir o número de falhas de página.

Se o acesso às páginas de memória for totalmente randômico, não há vantagem significativa em usar o algoritmo NPR (Newest Page Replacement) ou o LRU (Least Recently Used) em termos de redução do número de falhas de página. Em cenários com acessos totalmente aleatórios ambos os algoritmos se aproximam de uma estratégia quase aleatória de substituição. O desempenho será principalmente determinado pela quantidade de quadros disponíveis na memória, não pela escolha do algoritmo. O NPR, mesmo sendo mal projetado para padrões reais, não é pior que o LRU nesse caso específico, mas também não apresenta vantagens. Portanto, a escolha do algoritmo não faria diferença em sistemas onde os padrões de acesso às páginas são completamente randômicos.

5. Um sistema de memória virtual utiliza um esquema de substituição de páginas com o algoritmo LRU (Least Recently Used). Considere a sequência de acessos às páginas: 1, 2, 1, 3, 4, 1, 2, 5, 1, 2, 4, 1, 3, 1 em um sistema com apenas 3 quadros de página na memória física.(Least Recently Used)

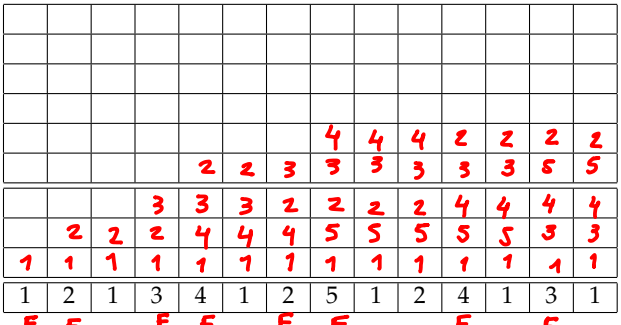
   Simule o comportamento do sistema, mostrando a evolução das páginas de memória a cada acesso.

10 (a) Determine o número de falhas de página para o algoritomo FIFO (First In First Out).



1	2	1	3	4	1	2	5	1	2	4	1	3	1
F	F		F	F	F	F	F			F	F	F	

10 (b) Determine o número de falhas de página para o algoritomo LRU (Least Recently Used).,



1	2	1	3	4	1	2	5	1	2	4	1	3	1
F	F		F	F		F	F			F		F	

6. Em um sistema Unix, as chamadas ao sistema de Entrada/Saída (E/S) são fundamentais para a interação com dispositivos, arquivos e fluxos de dados. Considere as operações de E/S e a forma como o Unix lida com dispositivos de bloco e caracteres. Responda às perguntas a seguir:

10 (a) No Unix, como o sistema operacional lida com a abstração de arquivos e dispositivos através das chamadas ao sistema de E/S? Explique como as chamadas de sistema são usadas para interagir com dispositivos de E/S.

10 (b) Explique em que situações deve ser utilizado um Pipe Anônimo (anonymous pipe, simplex FIFO) em sistemas Unix.

A) No Unix, o sistema operacional fornece uma abstração uniforme para arquivos e dispositivos através de sua interface de sistema de entrada e saída (E/S). Arquivos e dispositivos (como discos, terminais, impressoras, etc.) são tratados de maneira similar, usando a mesma interface de chamadas de sistema. Isso simplifica o desenvolvimento de software e promove a consistência. Tudo é um arquivo: No Unix, praticamente tudo é tratado como um arquivo, incluindo dispositivos de hardware. Arquivos comuns (dados em disco) e dispositivos (teclado, mouse, etc.) são representados por descritores de arquivo (file descriptors). Cada dispositivo é associado a um nó de dispositivo especial no sistema de arquivos, localizado geralmente em /dev. Chamadas ao sistema de E/S podem ser resumidas como operaçoes de arquivos: OPEN, READ, WRITE, SEEK

B) Os pipes anônimos (ou Simplex FIFO) são uma das formas mais simples de comunicação entre processos (IPC - Interprocess Communication) em sistemas Unix. Eles permitem que dados sejam transmitidos de maneira unidirecional entre dois processos relacionados. O uso de um pipe anônimo é indicado em cenários específicos devido às suas características e limitações: Comunicação entre Processos Relacionados: Permite troca de dados entre processos pai e filho. Fluxo Unidirecional de Dados:

- Facilita a transmissão de informações de um processo para outro.
- Encadeamento de Processos em Linha de Comando: Usado para conectar comandos em pipelines no shell.

- Transmissão de Dados Temporários: Ideal para dados que não precisam de armazenamento permanente.
- Evitar Arquivos Temporários: Transfere dados diretamente em memória, sem usar disco.
- Simplicidade e Desempenho: Proporciona uma solução IPC eficiente e fácil de implementar.

10 7. Considere o método de criptografia Cifras de transposição. Suponha que a chave seja "SOE" e que foi recebida uma mensagem (criptografada):

"AFAOAZTB XENNIEIA EMILDDOO".

Determine qual a mensagem original.

Considere o alfabeto e índices a seguir:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	Z

S (3)	O (2)	E (1)
E	X	A
M	E	F
I	N	A
L	N	O
D	I	A
D	E	Z
O	I	T
O	A	B

EXEME FINAL NO DIA DEZOITO

10 8. Considere que você deseja armazenar senhas de usuário em um arquivo. Descreva como fazer isso de forma segura. Considere, como exemplo, que você decidiu usar o método de cifras polialfabéticas, e forneça um exemplo de como a senha de um usuário definida como "URUBU" seria armazenada.