1. Explique a diferença entre endereços lógicos e endereços físicos e as razões que justificam seu uso.

R: Endereço lógico é o endereço a nível de programa que é gerado na compilação, ele
enxerga a memória como sendo unicamente para o programa. Através da realocação dinâmica
que consiste em utilizar um endereço base(endereço físico) e os endereços lógicos como
offset, obtém-se o endereço físico para cada endereço lógico. Sendo o endereço físico um
endereço que representa uma localização real e válida na memória.

2. Explique em que consiste a resolução de endereços nos seguintes momentos: codificação, compilação, ligação, carga e execução.

R: Codificação: programa escolhe a posição de cada variável e do código do programa (Sistemas embarcados em linguagem de máquina).

Compilaçao: compilados escolhe a posição das variáveis na memória, código fonte faz e parte do programa deve se conhecido no momento da compilação para evitar conflito em endereços na memória.

Ligação: compilador gera símbolos que representam as variáveis

Carga: define os objetos de variáveis e funções de carga do código em memória para lançamento de novo processo.

3.Como é organizado o espaço de memória de um processo?

R: Text: contém o código a ser executado pelo processo, gerado durante a compilação e a ligação das bibliotecas.

Data: dados estáticos usados pelos programas.

Heap: armazenam dados para alocação dinâmica, malloc, free.

Slock: mantém a pilha de execução do processo.

4. O que é uma MMU – Memory Management Unit?

R: É um hardware especial que faz a análise dos endereços lógicos emitidos pelo processador e determina os endereços físicos correspondentes na memória da máquina, permitindo então seu acesso pelo processador.

5. Seria possível e/ou viável implementar as conversões de endereços realizadas pela MMU em software, ao invés de usar um hardware dedicado? Por que?

R: Sim. Porque a proteção de memória entre processos é essencial para a segurança e estabilidade dos sistemas mais complexos, nos quais centenas ou milhares de processos podem estar na memória simultaneamente. Assim a cada troca de contexto entre processos, as regras de conversão da MMU devem ser ajustadas para somente permitir o acesso a área de memória definida para cada novo processo corrente.

6. Analise as seguintes afirmações relativas ao uso da memória RAM pelos processos:
I. Os endereços físicos gerados pelo processador são convertidos em endereços lógicos através da MMU - Memory Management Unit.
II. O acesso a endereços de memória inválidos é notificado ao processador através de interrupções geradas pela MMU.
III. A área de memória TEXT contém o código-fonte a ser compilado e executado pelo processo.
IV. A área de memória DATA é usada para armazenar todas as variáveis e constantes usadas pelo processo.
V. A área de memória HEAP é usada para as alocações dinâmicas de memória, sendo usada através de funções como malloc e free.
VI. A área de memória STACK contém as pilhas do programa principal e das demais threads do processo.
Indique a alternativa correta:
(a) As afirmações II e III estão corretas.
(b) As afirmações I e V estão corretas.
(c) Apenas a afirmação III está correta.
(d) As afirmações II e V estão corretas.
(e) As afirmações IV e VI estão corretas.
Justifique as afirmações julgadas erradas (Assim: XIV está errada porque ...):

R: Resposta certa: D

I. FALSO. A MMU, faz apenas a análise dos endereços lógicos emitidos pelo processador e determina os endereços físicos correspondentes na memória da máquina, permitindo então seu acesso pelo processador.
III. FALSO. Contém o código a ser executado pelo processo, gerado durante a compilação e a ligação das bibliotecas.
IV. FALSO. Porque está área da memória possui tamanho fixo, deve estar acessível para leituras e escritas, mas não para execução.
VI. FALSO. Contém somente a pilha do programa principal, como threads podem ser criadas e destruídas dinamicamente, a pilha de cada thread é mantida em uma área própria, geralmente alocada no heap.

7. Explique as principais formas de alocação de memória.

R: Partições fixas - a forma mais simples de alocação de memória consiste em dividir a memória destinada aos processos em N partições fixas, de tamanhos iguais ou distintos. Nesse esquema, a tradução entre os endereços lógicos vistos pelos processos e os endereços físicos é feita através de um simples registrador de relocação, cujo valor é somado ao endereço lógico gerado pelo processador, a fim de obter o endereço físico correspondente.
Alocação contígua - nesse caso, a MMU deve ser projetada para trabalhar com dois

registradores próprios: um registrador base, que define o endereço inicial da partição ativa, e um registrador limite, que define o tamanho em bytes dessa partição. A tradução de endereços lógicos em físicos ocorre da seguinte forma: cada endereço lógico gerado pelo processo em execução é comparado ao valor do registrador limite; caso seja maior ou igual a este, uma interrupção é gerada pela MMU de volta para o processador, indicando um endereço inválido. Caso contrário, o endereço lógico é somado ao valor do registrador base, para a obtenção do endereço físico correspondente.
Alocação por segmentos - é uma extensão da alocação contígua, na qual o espaço de memória de um processo é fracionado em áreas, ou segmentos, que podem ser alocados separadamente na memória física. Ao estruturar a memória em segmentos, o espaço de memória de cada processo não é mais visto como uma sequência linear de endereços lógicos, mas como uma coleção de segmentos de tamanhos diversos e políticas de acesso distintas.
Alocação paginada - o espaço de endereçamento lógico dos processos é mantido linear e unidimensional (ao contrário da alocação por segmentos, que usa endereços bidimensionais). Internamente, e de forma transparente para os processos, o espaço de endereços lógicos é dividido em pequenos blocos de mesmo tamanho, denominados páginas.
Alocação segmentada paginada - os processos veem a memória estruturada em segmentos, o hardware da MMU converte os endereços lógicos na forma [segmento:offset] para endereços lógicos lineares (unidimensionais), usando as tabelas de descritores de segmentos. Em seguida, esse endereços lógicos lineares são convertidos nos endereços físicos correspondentes através do hardware de paginação (tabelas de páginas e TLB), visando obter o endereço físico correspondente.

8. Explique como é feita a translação entre endereços lógicos e físicos e o mecanismo de tratamento de falta de página em um sistema de memória virtual paginada.

R: A ideia central do mecanismo de memória virtual em sistemas com memória paginada consiste em retirar da memória principal as páginas menos usadas, salvando-as em uma área do disco rígido reservada para esse fim. O armazenamento externo das páginas pode ser feito em um disco exclusivo para esse fim (usual em servidores de maior porte), em uma partição do disco principal(usual no Linux e outros UNIX) ou em um arquivo reservado dentro do sistema de arquivos do disco principal da máquina, geralmente oculto (como no Windows NT e sucessores). Quando um processo tentar acessar uma página ausente, esta deve ser transferida de volta para a memória para permitir seu acesso, de forma transparente ao processo. Caso a página não exista, o processo tentou acessar um endereço inválido e deve ser abortado.

9. Por que os tamanhos de páginas e quadros são sempre potências de 2?

R: Pois a memória é dividida blocos de tamanho fixo chamados de frames-molduras de páginas e as páginas tem que ser construídas de forma a se encaixar melhor nessas molduras. RDM(mesma coisa, só um complemento): A memória física é dividida em blocos de tamanho fixo(frames-molduras de páginas) sempre em tamanhos de potência de 2 e entre 512 ~ 8192 bytes. Por isso a memória lógica também deve ser dividida em blocos(páginas) de mesmo tamanho que a da física.

10. Analise as seguintes afirmações relativas às técnicas de alocação de memória:
I. Na alocação em partições fixas, a MMU é composta basicamente de um registrador e um somador. **(verdadeiro)**
II. Na alocação contígua, a área de memória acessível a cada processo é definida por um registrador base e um registrador limite. **(verdadeiro)**
III. A técnica de alocação contígua é imune a problemas de fragmentação externa. **(falso, essa técnica é sim sujeita a fragmentação externa)**
IV. A alocação por segmentos resolve o problema da fragmentação externa. **(falso, ela não resolve o problema de fragmentação)**
V. Na alocação por segmentos, cada endereço de memória é composto de duas partes: segmento e deslocamento. **(verdadeiro)**
VI. A alocação por páginas resolve o problema da fragmentação externa. **(verdadeiro)**
Indique a alternativa correta:

(a) As afirmações II, III e VI estão corretas.
(b) As afirmações I, II, V e VI estão corretas. (verdadeiro)
(c) Apenas a afirmação V está correta.
(d) As afirmações IV e VI estão corretas.
(e) Todas as afirmações estão corretas.

11. Considerando a tabela de segmentos abaixo (com valores em decimal), calcule os
endereços físicos correspondentes aos endereços lógicos 0:45, 1:100, 2:90, 3:1.900 e
4:200.

Segmento	0	1	2	3	4
Base	44	200	0	2.000	1.200
Limite	810	200	1.000	1.000	410

R: Considerando a arquitetura como sendo de alocação contínua temos que:
segmento : E.L = E.F, sendo E.F = base + E.L se E.L < limite

0:45 = 44 + 45 = **89**
1:10 = 200 + 100 = **300**

2:90 = 0 + 90 = **90**
3:1900 = **IRQ (endereço inválido)**
4:200 = 1200 + 200 = **1400**

12. Considerando a tabela de páginas abaixo, com páginas de 500 bytes1 , informe os endereços físicos correspondentes aos endereços lógicos 414, 741, 1.995, 4.000 e 6.633, indicados em decimal.

13. Considere um sistema com endereços físicos e lógicos de 32 bits, que usa tabelas de páginas com três níveis. Cada nível de tabela de páginas usa 7 bits do endereço lógico, sendo os restantes usados para o offset. Cada entrada das tabelas de páginas ocupa 32 bits. Calcule, indicando seu raciocínio:

(a) O tamanho das páginas e quadros, em bytes.

R: 2^11 Bytes, ou 2 KBytes.

(b) O tamanho máximo de memória que um processo pode ter, em bytes e páginas.

R: 2^21 páginas com 2^11 bytes.

(c) O espaço ocupado pela tabela de páginas para um processo com apenas uma
página de código, uma página de dados e uma página de pilha. As páginas de código e de dados se encontram no inicio do espaço de endereçamento lógico, enquanto a pilha se encontra no final do mesmo.

R: 2^11 x 3 x 2^7 bytes.

(d) Idem, caso todas as páginas do processo estejam mapeadas na memória.

R: 2^11 x (2^7 x 2^7 x 2^7 + 2^7) bytes.

14. Analise as seguintes afirmações relativas à alocação paginada:...

R: (d) As afirmações I, III e VI estão corretas.

II. É falsa pois o objetivo das tabelas multiníveis é reduzir o espaço usado pelas tabelas na memória e não reduzir a velocidade de acesso.

IV. Não é verdadeira pois ele apenas armazena os pares (página, quadro) dos endereços mais recentes e não por nível de frequência.

V. É incorreta porque o bit é modificado pela própria MMU e não pelo núcleo.

15. Explique o que é TLB, qual a sua finalidade e como é seu funcionamento.

R: O cache de tabela de páginas na MMU, que visa reduzir o tempo de conversão de endereços lógicos para físicos. Ele armazena pares (página, quadro) obtidos em consultas recentes às tabelas de páginas do processo ativo.

16. Por que é necessário limpar o cache TLB após cada troca de contexto entre processos? Por que isso não é necessário nas trocas de contexto entre threads?

R: Porque a cada troca de contexto, a tabela de páginas é substituída, motivo que faz com que o cache TBL deve ser esvaziado, pois seu conteúdo não é mais

válido. Isso permite concluir que trocas de contexto muito frequentes prejudicam a eficiência de acesso à memória, tornando o sistema mais lento. Essa necessidade de trocas de contexto não existe nas threads pelo motivo de que só ocorre o gerenciamento e percepção de apenas um fluxo de execução dentro de cada processo.

17. Um sistema de memória virtual paginada possui tabelas de página com três níveis e tempo de acesso à memória RAM de 100 ns. O sistema usa um cache TLB de 64 entradas, com taxa estimada de acerto de 98%, custo de acerto de 10 ns e penalidade de erro de 50 ns. Qual o tempo médio estimado de acesso à memória pelo processador? Apresente e explique seu raciocínio.

R: O tempo médio de acesso à memória pode então ser determinado pela média ponderada entre o tempo de acesso com acerto de cache e o tempo de acesso no caso de erro:

TMédio = 98% x 10ns
+ 5% x (50ns + (3 x 100ns))
+ 100ns
TMédio = 127,3ns

18. Explique o que é fragmentação externa. Quais formas de alocação de memória estão
livres desse problema?

R: Fragmentação externa é um problema que somente afeta as estratégias de alocação que trabalham com blocos de tamanho variável, como alocação contígua e alocação segmentada. Somente a alocação paginada é imune à fragmentação externa, pois sempre trabalha com blocos de mesmo tamanho.

19. Explique o que é fragmentação interna. Quais formas de alocação de memória estão livres desse problema?

R: Fragmentação interna: É a perda de espaço dentro de uma área de tamanho fixo. Numa memória secundária, ela ocorre quando um arquivo ou fragmento de arquivo não ocupa completamente o espaço da unidade de alocação destinado a ele, causando desperdício de espaço.

Alocações livres da fragmentação externa:

- Alocação encadeada: Esta estratégia elimina a fragmentação externa, pois todos os blocos livres do disco são utilizados sem restrições, e permite que arquivos sejam criados sem a necessidade de definir seu tamanho final.
- Alocação indexada: Arquivos podem ser criados em qualquer local do disco, sem risco de fragmentação externa;

20. Em que consistem as estratégias de alocação first-fit, best-fit, worst-fit e next-fit?

- first-fit: Consiste em escolher a primeira área livre que satisfaça o pedido de alocação; tem como vantagem a rapidez, sobretudo se a lista

de áreas livres for muito longa;

- best-fit: Consiste em escolher a menor área possível que possa atender à solicitação de alocação. Dessa forma, as áreas livres são usadas de forma otimizada, mas eventuais resíduos (sobras) podem ser pequenos demais para ter alguma utilidade;
- worst-fit: Consiste em escolher sempre a maior área livre possível, de forma que os resíduos sejam grandes e possam ser usados em outras alocações;
- next-fit: Variante da anterior (first-fit) que consiste em percorrer a lista a partir da última área alocada ou liberada, para que o uso das áreas livres seja distribuído de forma mais homogênea no espaço de memória;

22. Considere um banco de memória com os seguintes “buracos” não-contíguos:

B1	B2	B3	B4	B5	B6
10MB		4MB	7MB	30MB	12MB 20MB

Nesse banco de memória devem ser alocadas áreas de 5MB, 10MB e 2MB, nesta ordem, usando os algoritmos de alocação First-fit, Best-fit ou Worst-fit. Indique a alternativa correta:

(a) Se usarmos Best-fit, o tamanho final do buraco B4 será de 6 Mbytes.
(b) Se usarmos Worst-fit, o tamanho final do buraco B4 será de 15 Mbytes. <- CORRETA
(c) Se usarmos First-fit, o tamanho final do buraco B4 será de 24 Mbytes.
(d) Se usarmos Best-fit, o tamanho final do buraco B5 será de 7 Mbytes.
(e) Se usarmos Worst-fit, o tamanho final do buraco B4 será de 9 Mbytes.

R:
A alternativa 'B' está correta pois o *Worst-fit* utiliza sempre o maior buraco para preencher a solicitação. Nesse caso, com o valor (seguindo a sequência) de 5MB adicionado, o B4 será 25MB de espaço (30 - 5), seguindo com a alocação de 10MB, o buraco B4 passaria a ser 15MB (25-10MB). A próxima inserção será de 2MB, mas será realizada, seguindo a propriedade do *Worst-fit*, em B6, pois é o maior buraco.

23. Considerando um sistema de 32 bits com páginas de 4 KBytes e um TLB com 64 entradas, calcule quantos erros de cache TLB são gerados pela execução de cada um dos laços a seguir. Considere somente os acessos à matriz buffer (linhas 5 e 9), ignorando páginas de código, heap e stack. Indique seu raciocínio.

```
1-    unsigned char buffer[4096][4096] ;
2-
3-    for (int i=0; i<4096; i++) // laço 1
4-          for (int j=0; j<4096; j++)
```

```
5-                      buffer[i][j] = 0 ;
6-
7-      for (int j=0; j<4096; j++) // laço 2
8-              for (int i=0; i<4096; i++)
9-                      buffer[i][j] = 0 ;
```

R: O *laço 1*, já que percorre linha por linha da matriz, usa de forma mais eficiente o cache, gerando apenas um erro a cada linha acessada, resultando em 4096 erros de cache.
Já o *laço 2*, percorre coluna à coluna, gerando um erro à cada célula acessada na matriz. Nesse caso resulta em 16.777.216 erros no segundo laço.

25. O que é uma falta de página? Quais são suas causas possíveis e como o sistema operacional deve tratá-las?

R: Falta de página significa que em um processo de acesso, uma página não está presente. Então, a Unidade de Gerenciamento de Memória gera uma interrupção para verificar se a página existe. Se existir, gerencia o espaço, removendo outras páginas, se necessário, de modo que a página em questão seja ajustada na tabela de páginas e retorna ao processo de acesso.

26. Calcule o tempo médio efetivo de acesso à memória se o tempo de acesso à RAM é de 5 ns, o de acesso ao disco é de 5 ms e em média ocorre uma falta de página a cada 1.000.000 (106) de acessos à memória. Considere que a memória RAM sempre tem espaço livre para carregar novas páginas. Apresente e explique seu raciocínio

R: Acesso a ram = 5 ns
Acesso a disco = 5 ms
e com a falta de página a cada 10^6 e com espaço livre no disco o tempo médio é igual a 10 ns

28. Considere um sistema de memória com quatro quadros de RAM e oito páginas a alocar. Os quadros contêm inicialmente as páginas 7, 4 e 1, carregadas em memória nessa seqüência. Determine quantas faltas de página ocorrem na seqüência de acesso {0, 1, 7, 2, 3, 2, 7, 1, 0, 3}, para os algoritmos de escalonamento de memória FIFO, OPT e LRU.

R: FIFO: 5 faltas de página
OPT: 5 faltas de página
LRU: 4 faltas de página

29. Repita o exercício anterior considerando um sistema de memória com três quadros de RAM.

R:

FIFO: 6
OPT: 5

LRU: 7

30. Um computador tem 8 quadros de memória física; os parâmetros usados pelo mecanismo de memória virtual são indicados na tabela a seguir:

página	carga na memória	último acesso	bit R	bit M
p_0	14	58	1	1
p_1	97	97	1	0
p_2	124	142	1	1
p_3	47	90	0	1
p_4	29	36	1	0
p_5	103	110	0	0
p_6	131	136	1	1
p_7	72	89	0	0

Qual será a próxima página a ser substituída, considerando os algoritmos LRU, FIFO, segunda chance e NRU? Indique seu raciocínio.

R:

LRU: p2, pois é a página que foi acessada a mais tempo.
FIFO: p0, se considerarmos que a última página inserida foi a p7.
Segunda Chance: p3, se considerarmos que a última página inserida foi a p7, ele buscará a partir da p0 (a primeira inserida) a primeira página cujo bit de referência seja 0.
NRU: p5, se considerarmos que a última página inserida foi a p7, ele buscará a partir da p0 (a primeira inserida) a primeira página cujo bit de referência e o de edição seja 0.

31. Considere um sistema com 4 quadros de memória. Os seguintes valores são obtidos em dez leituras consecutivas dos bits de referência desses quadros: 0101, 0011, 1110, 1100, 1001, 1011, 1010, 0111, 0110 e 0111. Considerando o algoritmo de envelhecimento, determine o valor final do contador associado a cada página e indique que quadro será substituído.

R:
q1: 0001 1111
q2: 1110 0011
q3: 1111 1001
q4: 1010 1100