**Universidade Federal de Rondônia**

**Departamento Acadêmico de Ciência da Computação**

**Estrutura de Dados II**

**Lista 2 – Arquivos**

**Pode ser feito em duplas**

**Baseada na lista de exercícios da Profa. Maria Cristina, e livro do Folk.**

1. Explique a diferença entre arquivo lógico e arquivo físico.
2. Descreva as operações fundamentais que podem ser realizadas em um arquivo. Descreva as funções que executam estas operações na linguagem de programação que você usa, e como elas são utilizadas. Por que existem vários modos de abrir um arquivo?
3. Quais as funções de um gerenciador de arquivos?
4. Explique porque os arquivos abertos devem ser fechados.
5. Descreva o que acontece quando um arquivo já existente é aberto por um aplicativo (p. ex., um programa) para escrita. E se o mesmo ocorrer com um arquivo não existente?
6. Os sistemas de arquivos permitem definir atributos para controlar o acesso a um arquivo. O que acontece quando um programa tenta abrir um arquivo que tem proteção para leitura? E quando o arquivo tem proteção para escrita?
7. Muitos sistemas diferenciam os arquivos binários dos arquivos de texto. Qual a diferença entre eles?
8. No que consiste a operação de posicionamento (*seeking*) em um arquivo? Qual a sua utilidade? Exemplifique uma situação em que esta operação precisa ser utilizada.
9. Faça um programa que leia os últimos 10 caracteres de um arquivo qualquer e imprima-os na tela.
10. Faça um programa que leia o conteúdo de um arquivo e o escreve na tela.
11. Faça um programa para copiar o conteúdo de um arquivo para outro.
12. Faça um programa que lê um vetor numérico pelo teclado e o escreve em um arquivo.
13. Discuta as diferenças e semelhanças entre a memória principal (RAM) e a memória secundária (os arquivos).
14. Como são organizados fisicamente os discos? De que forma os discos armazenam os arquivos? Por que o tamanho real de um arquivo em disco é sempre maior do que o seu tamanho nominal?
15. Quais parâmetros são considerados para calcular o tempo de leitura de um arquivo mantido em disco?
16. Explique o que é um cilindro, e a razão para a organização de arquivos em cilindros.
17. Explique o que é um *cluster* e o que é um *extent*.
18. O que é a fragmentação de um arquivo no disco? Quais os tipos de fragmentação do arquivo, por que elas ocorrem e quais seus efeitos?
19. Discuta as vantagens e as desvantagens de organizar arquivos em blocos de tamanho definido pelo usuário, ao invés de em setores de tamanho fixo.
20. Por que os discos são considerados o gargalo de um sistema computacional? Explique como este problema pode ser minimizado.
21. Como os arquivos são organizados em uma fita magnética? Por que as fitas organizam os dados em blocos?
22. Quais as vantagens e desvantagens de fitas sobre discos com dispositivos de memória secundária?
23. No que consiste um sistema de armazenamento terciário?
24. Quais os parâmetros são considerados para calcular o tempo de leitura de um arquivo em fita? Procure estas informações para um dispositivo de fita comercial e calcule quanto tempo tal dispositivo levaria para ler um arquivo de 1MB.
25. O que são buffers de E/S (ou I/O)? Quais os passos executados para ler um byte do disco de forma que ele possa ser utilizado por um programa?
26. As aplicações usualmente armazenam as informações em arquivos organizando-as em campos e registros. Explique as diferentes maneiras pelas quais um campo pode ser armazenado em um arquivo para posterior recuperação.
27. Explique as diferentes estratégias que podem ser utilizadas para separar um registro de outro. Discuta as vantagens e desvantagens de cada uma delas.
28. Explique o que é fragmentação de campos e registros. Quando e por que ela ocorre?
29. Se a separação entre registros e campos é feita por delimitadores, quais as restrições para a escolha desses delimitadores? Descreva uma situação que exemplifique sua resposta.
30. Crie um programa para escrever registros de tamanho variável em um arquivo e outro capaz de recuperá-los. Faça o mesmo para registros de tamanho fixo. Os registros devem ter pelo menos 3 campos.
31. O que é gravado no arquivo quando uma struct do C é escrita em um arquivo? Como são armazenados campos que não são strings?
32. Como um registro é identificado para acesso aleatório? Qual operação permite localizar um registro no arquivo em C?
33. Explique como é possível melhorar o desempenho de um acesso sequencial a todo o conteúdo de um arquivo. Tal solução também garante um melhor desempenho de uma sequência arbitrária de acessos aleatórios? Discuta.
34. Quantas leituras são necessárias, em média, para encontrar um registro em um arquivo com N registros usando a busca sequencial? Quantas leituras são necessárias para identificar que um registro não está no arquivo?
35. Quais as vantagens e as desvantagens de utilizar arquivos organizados em registros de tamanho fixo?
36. O que é RRN? Como é possível fazer acessos aleatórios em arquivos a registros de tamanho variável?
37. É vantajoso manter um arquivo separado para armazenar apenas as chaves e os byte *offsets*, ou RRNs, dos registros no arquivo de dados? Como isto afeta a inserção de um novo registro?
38. Como um registro pode ser eliminado de um arquivo?
39. O que são modelos abstratos de dados, para que são utilizados e quais as suas vantagens?
40. Por que é interessante utilizar cabeçalhos nos arquivos?
41. Em princípio, não é possível fazer busca binária em um arquivo de dados com registros de tamanho variável. Por que a indexação do arquivo torna a busca binária possível? Com um arquivo com registros de tamanho fixo é possível fazer busca binária. Isto significa que indexação não é necessária para arquivos de registros de tamanho fixo?
42. Considerando um cadastro de CDs, por que título não é usado como chave primária no arquivo de dados? Se título fosse usado como chave secundária, que problemas deveriam ser considerados na escolha de uma forma canônica para os títulos?
43. Qual o propósito em deixar um indicador de desatualizado no cabeçalho de um índice? Em um ambiente de multiprogramação, este indicador poderia ser encontrado “setado” por um programa porque outro programa está em processo de reindexação. Como o primeiro programa deveria responder a esta situação?
44. Quando um registro é atualizado num arquivo, as chaves primárias e secundárias do índice podem ser alteradas ou não, dependendo do arquivo ter registros de tamanho fixo ou variável e dependendo do tipo de alteração que foi feita no registro de dados. Faça uma lista das situações diferentes que podem ocorrer devido a atualizações e explique como cada uma pode afetar os índices.
45. Discuta o problema que ocorre quando você inclui o registro abaixo em seu arquivo, assumindo que o índice por compositor usado é o mostrado na Figura 6.9. Como você poderia resolver o problema sem grandes alterações na estrutura do índice de chave secundária?

LON 1259 Fidelio Beethoven Maazel



1. O que é uma lista invertida? Quando ela é útil? Como ela é mantida em memória secundária? Esquematize o conteúdo de um índice secundário organizado como lista invertida para um arquivo de dados hipotético.
2. Como são alteradas as estruturas mostradas na figura 6.11 pela inclusão do registro abaixo?

LON 1259 Fidelio Beethoven Maazel



1. Suponha que você tenha um arquivo de dados de CD’s, muito grande, com um índice pela chave primária e índices pelas chaves secundárias organizados por compositor, artista e título. Suponha que uma lista invertida é usada para as chaves secundárias. Explique, passo a passo, como um programa poderia responder às seguintes solicitações:
   1. Liste todas as gravações de Bach ou Beethoven;
   2. Liste todas as gravações de Perleman de peças de Mozart ou Joplin.
2. O método e temporização do *binding* afeta dois importantes atributos de um sistema de arquivos: velocidade e flexibilidade. Discuta a relevância desses atributos, e o efeito do *binding* sobre o tempo em cada um dos atributos acima, para um sistema de informação de um hospital projetado para prover informação sobre os pacientes pelas chaves Nome do paciente, Código do paciente, Localização, Medicação, Médico ou Médicos, e Doença.
3. Suponha que você tenha 8MB de memória RAM para ordenar um arquivo de 800.000 registros.
   1. Quanto tempo durará a ordenação do arquivo usando o algoritmo *merge sort*?
   2. Quanto tempo durará a ordenação do arquivo usando o algoritmo *key sort*?
   3. Por que o algoritmo *key sort* não funcionará se houver 1MB de RAM disponível para a fase de ordenação?
4. Quantos posicionamentos (*seeks*) são necessários para uma intercalação em um único passo, se um posicionamento toma 50ms, em média, e o tamanho do *buffer* interno disponível é 500k? E se for 100k?
5. Em nossos cálculos envolvendo merge, assumimos que só uma busca e um atraso rotacional são necessários para um único acesso sequencial. Se isso não ocorrer, mais tempo será necessário para a realização de I/O. Por exemplo, o arquivo de 80 MB usado no exemplo em sala, para a fase de leitura do processo de geração de corridas, a leitura de cada corrida pode requerer muitos acessos. Assuma que o tamanho do *extent* para nosso drive hipotético é de 20.000 bytes (aproximadamente uma trilha), e que todos arquivos armazenados em blocos do tamanho da trilha devem ser acessados separadamente (uma procura e um atraso rotacional por bloco).
   1. Quantas buscas o passo 1 requer agora?
   2. Quanto tempo os passos 1, 2 e 3 levam?
   3. Qual é o impacto de um aumento no tamanho do arquivo por um fator 10 terá no tempo total do merge sort?
6. Suponha que um arquivo de dados com 6.000 registros, mantido em disco, deve ser ordenado em um computador cuja memória interna acomoda no máximo 600 registros por vez, usando o procedimento de intercalação. Considere que serão geradas 10 corridas de 600 registros cada, e que será realizada uma intercalação em 10 vias. Essa mesma área de memória interna é usada como buffer de entrada para leitura de dados do disco, e o sistema conta com um buffer de saída adicional que acomoda 200 registros.
   1. Durante a intercalação, quantos registros serão lidos de cada corrida cada vez que ela é acessada? Justifique.
   2. Quantos *seeks* serão realizados para ler dados durante o processo de intercalação (excluindo a fase de geração de corridas)? Quantos serão realizados para escrever dados? Justifique.