Universidade de Brasília

Departamento de Ciência da Computação

Lista de Exercício 2 - OA

Autores:

Gabriel Bessa 16/0120811

Thiago Veras 16/0146682

Disciplina:

Organização de Arquivos

Turma:

A

Professor:

Oscar Gaidos

Brasília

11 de Abril de 2018

Exercícios:

1- Fator de blocagem:

- Descubra qual o tamanho da cluster do seu HD
- Faça um programa que gere cinco arquivos contendo os mesmos dados, cada um com um fator de blocagem correspondente aos valores abaixo:
512 bytes, 1/4, 1/2, 3/4 e uma cluster.
- O tamanho do registro e da cluser devem poder ser passados via linha de comando.

Resposta:

Tamanho da cluster no computador: 4096 Bytes.

Figura 1: Número de clusters

Figura 2: Tamanho dos clusters

Explicação do Código:

```
1  #include <stdio.h>    // Use function Prinf()
2  #include <iostream>   // Use cin and cout functions
3  #include <fstream>    // Use open() function
4  #include <regex>      // Use vector class
5  #include <string>     // Use String to read files name
6
7  // Use collors on printf
8  #define RED      "\x1b[31m"
9  #define GREEN    "\x1b[32m"
10 #define CYAN     "\x1b[36m"
11 #define RESET    "\x1b[0m"
12
13 // Used to omit ::std syntax
14 using namespace std;
```

Começo do código com todas as bibliotecas para uso de funções e defines para cores no terminal.

```
int main() {
    // Open file
    fstream file ("teste.txt", ios::in | ios::binary);

    // Fail opening file
    if(!file) return cout << RED << "File could not be
       opened\n" << RESET,0;

    // Block factors
    vector<int> factors = {512, 4096/4, 4096/2,
       3*4096/4, 4096};

    // Messages for each blocks
    vector<char*> message[5];

    // File length:
    file.seekg (0, ios::end);

    // Length = file length
    int length = file.tellg(), block, i = 0;

    // Iterate through all block factors
    for(int buf : factors){

        // returns to begin of file
        file.seekg(0);

        //reset block size
        block = 0;

        // Iterate while block size < file length
        while(block < length){

            // Var to read message from file
            char* curr_blk = (char*) malloc(buf);

            // Set all block to 0. May not fill it all
               block and null spaces needed to be 0
            memset(curr_blk, 0, buf);

            // To not read more than block length
            if(block + buf <= length)
```

```
40                  file.read(curr_blk, buf);
41              else
42                  file.read(curr_blk, length-block);
43
44              // adds message to that block factor
45              message[i].push_back(curr_blk);
46
47              // shift block size if file need more than 1
                    block
48              block += buf;
49          }
50          i++;
51      }
52
53      // OK message
54      cout << GREEN << "Blocks completed\n" << RESET;
55      for(int i = 0; i < 5; i++) printf("%dK : %d block(s)
           \n", factors[i], (int) message[i].size());
56
57      // Output file
58      fstream out; i = 0;
59
60      // Iterate through all factors to write a new file
61      for(auto buf : factors){
62
63          // Open factor size file name
64          out.open("out_" + to_string(buf) + ".txt", ios::
               out);
65
66          // Writed message from blocks to another file
67          for(auto blk : message[i++]) out.write(blk, buf)
               ;
68
69          // Close file
70          out.close();
71      }
72
73  }
```

Na linha 21 se lê os blocos de tamanho pré definido. Na linha 30, se lê o bloco (variável block) enquanto seu tamanho for menor que o tamanho do arquivo (variável length), depois na linha 39 verifica se é possivel ler um bloco sem exceder o limite do arquivo, caso contrário lê o que pode até o final

e salva na variavel curr_blk, passando para o vetor message. Para finalizar a leitura, incrementa-se a variável block com o tamanho do buffer lido (variável buff), realizando os deslocamento da leitura do arquivo.

2- Em uma fita de 2400 pés, densidade 30000 bpi, gap de tamanho 0,3 polegadas, ache o espaço necessário para armazenar 10000 registros de tamanho de 150 bytes.

Resposta:

$$S = n(b + g) \tag{1}$$

- S = Espaço desejado;
- n = Número de de blocos(registro por bloco);
- b = Tamanho físico do bloco de dados;
  b é calculado por:
  $$b = t.b/b.p.i \tag{2}$$
- g = Tamanho do gap;
- tb = Tamanho dos bytes;
- b.p.i = Bytes per int;

O resultado encontrado foi de 3050 polegadas

3- Segundo a visão do projetista de arquivo, determine:
a. O fator do bloco, com perdas mínimas, para armazenar registros de 128 bytes, em setores de 512 bytes, cujo bloco não pode ser superior ao cluster de 1536 bytes. Cada Página é igual a 4 blocos.
b. Faça a mesma coisa com registros de 125 bytes.
c. Qual é a fragmentação interna no caso 3a e no caso 3b?

Resposta:

a. O fator de blocagem para armazenar em setores de 512 bytes, com perdas mínimas seria 12. Como o fator de blocagem é o número de registros dentro de um bloco, serão 3 blocos por 4*128 = 512, que é o tamanho máximo de um setor.

b. Nesse caso o fator de blocagem também seria 12, pois como o registro tem 125 bytes, não tem como adicionar nenhum extra em nenhum bloco evitando perdas. Pois 4*125 = 500, se fosse pra adicionar iria para 625, estourando o limite máximo de armazenamento de um setor.

c. Não houve fragmentação interna no caso do primeiro item, pois todos os 3 setores juntos(512*3) ocuparam a cluster de 1536 bytes. Já no segundo item houve fragmentação de 36 bytes(1536-1500), já que ficaram faltando 36 bytes para serem ocupados pelos registros de 125 bytes.

4- Qual a vantagem de um arquivo em disco armazenado
a. Numa única extensão
b. Várias extensões distintas?

Resposta:

a. Quando o arquivo é armazenado em uma única extensão, apenas um seeking(busca) é necessário, ou seja, ele pode ser processado no menor tempo possível de busca.

b. Com arquivos menores, eles podem causar fragmentação interna, pois o espaço alocado a fim de armazenar o arquivo, pode ser muito maior que o próprio arquivo.

5- Quais são as vantagens e desvantagens da organização de trilhas em setores com grande capacidade.

Resposta:

Por definição, as trilhas são círculos concêntricos, que começam no final do disco e vão se tornando menores conforme se aproximam do centro. E as trilhas são divididas em setores, onde ocorre o armazenamento de dados. Sendo assim, podemos concluir que em um setor com grande quantidade de armazenamento temos vantagens e desvantagens, das quais:

Vantagens:

- Armazenamento de mais bytes por setor de disco.
- Mais velocidade de leitura dos arquivos.
- Menos setores ocupados dentro do arquivo

Desvantagens:

- Fragmentação interna, devido ao pequeno tamanho dos arquivos. (Enunciado 4-B)