

Universidade de Brasília

Departamento de Ciência da Computação



Lista de Exercício 1 - OA

Autores:

Gabriel Bessa 16/0120811

Thiago Veras 16/0146682

Disciplina:

Organização de Arquivos

Turma:

A

Professor:

Oscar Gaidos

Brasília
30 de Março de 2018

1 Exercícios:

Exercício 1: Implemente um programa que leia e concatene UM OU MAIS arquivos textos e grave o arquivo concatenado em DISCO ou o exibe no VÍDEO. Faça esse programa o mais seguro possível, isto é, após operações de E/S, verifique se não houve erro. Se houve, sinalize para o usuário e contorne a situação. O programa deverá ter a opção de ler caracter a caracter ou leitura de bloco.

Explicação do Código: Nosso programa consiste em ler 'n' arquivos de uma só vez. Primeiramente ele checa se os arquivos podem ser abertos, uma vez que os mesmos precisam estar no mesmo diretório que o programa em si. Se forem abertos o nome de cada arquivo é salvo num vetor dinâmico de string. Quando o programa começa a rodar, ele vai na ordem que o usuário colocou os programas, lê a string de cada arquivo, salva num vetor dinâmico e joga no arquivo final, assim concatenando todas as versões numa versão final (final.txt).

Explicação das Funções:

Cabeçalho :

```
1 #include <stdio.h>    // Use function Printf()
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.

Menu :

```
1 void intro(){
2     printf(CYAN"\n");
3     printf("#####\n");
4     printf("#                               #\n");
5     printf("#           FILE CONCATENATOR           #\n");
6     printf("#                               #\n");
7     printf("#####\n");
8     printf("#                               #\n");
9     printf("#       Aluno: Thiago Veras Machado       #\n");
10    printf("#           M: 16/0146682               #\n");
11    printf("# Aluno: Gabriel Cunha Bessa Vieira #\n");
12    printf("#           M: 16/0120811               #\n");
13    printf("#                               #\n");
14    printf("#####\n");
15    printf("\n" RESET);
16    printf("How many files you want to read ? : ");
17 }
```

Função responsável por imprimir no terminal o nome e matrícula dos alunos responsáveis pelo trabalho.

Verificador de arquivos :

```
1 bool open_all(vector<string> files){
2     bool ans = true; // Return value if all files could be
3     opened
4     fstream arquivo;
5     for(auto file : files){ // Walk through files vector
6         to get file name
7         arquivo.open(file); // Open the file
8         cout << "Opening " << file << " : ";
9         if(!arquivo){ //Check if opened the file
10             cout << RED << "FAILED" << RESET << endl;
11             ans = false;
12         }
13         else{
14             cout << GREEN << "SUCCESS" << RESET << endl;
15             // Close file if was opened
16             arquivo.close();
17         }
18     }
19     return ans; // Return true if could open all files
20 }
```

Esta função é responsável por verificar se é possível abrir todos os arquivos fornecidos, primeiramente se inicializar a variável de retorno como verdadeira para poder retornar caso seja possível. se percorre todos as strings com o nome dos arquivos (linha 6) e tenta abrir o arquivo com aquele nome, se o arquivo não existir (linha 12) imprime na tela uma mensagem de falha e seta a variável de retorno como falsa, pois não conseguiu abrir o arquivo. Caso contrário, imprime uma mensagem em verde de sucesso e fecha o arquivo.

```
Opening teste1.txt : SUCCESS
Opening teste2.txt : SUCCESS
Concatenating all files to final.txt
Concatenation successful.
```

Figura 1: Testando 2 arquivos, ambos lidos com sucesso.

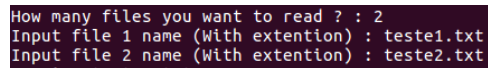
```
Opening teste1.txt : SUCCESS
Opening teste3.txt : FAILED
Opening teste2.txt : SUCCESS
Opening all files status : FAILED
System finished
```

Figura 2: Testando 3 arquivos, porém o arquivo teste3.txt não existe, logo falha em ler todos os arquivos.

Leitura do nome dos arquivos :

```
1 vector<string> read(int n){
2
3     // Vector of string to store all files names
4     vector<string> files;
5
6     // String to get the file by file name
7     string file;
8
9     for(int i = 1; i <= n; i++){
10         cout << "Input file " << i << " name (With extention
11             ) : ";
12         // Read file name
13         cin >> file;
14         // Push file to files vector
15         files.push_back(file);
16     }
17     return files;
}
```

A função `read` é responsável por simplesmente ler todos os 'n' nomes dos arquivos inseridos pelo usuário e retornar um vetor de string com todos os nomes.

A terminal window with a dark background showing the following text: "How many files you want to read ? : 2", "Input file 1 name (With extention) : teste1.txt", and "Input file 2 name (With extention) : teste2.txt".

```
How many files you want to read ? : 2
Input file 1 name (With extention) : teste1.txt
Input file 2 name (With extention) : teste2.txt
```

Figura 3: Lendo dois arquivos txt.

Concatenador de arquivos :

```
1 // Concatenate all files
2 void concatenate(vector<string> files){
3     // String to read line by line in the file
4     string line;
5     // Vector to store all lines concatenated
6     vector<string> all;
7     fstream arquivo;
8     // Walk through all files
9     for(auto file : files){
10         // Open the file
11         arquivo.open(file);
12
13         // Read all lines and store in vector all
14         while(getline(arquivo,line))all.push_back(line);
15
16         // Close file
17         arquivo.close();
18     }
19
20     // Open the final file that have all files
21     // concatenated
22     arquivo.open("final.txt");
23
24     // Check if file exist
25     if(!arquivo){
26         // If the file don't exist create a file called
27         // final.txt
28         cout << RED << "File final.txt don't exist ...
29         Creating a new one..." << RESET << endl;
30
31         // Clear all fstream content
32         arquivo.clear();
```

```

31
32 // Create a file
33 arquivo.open("final.txt", ios::out);
34
35 cout << GREEN << "File final.txt created successfully
36 " << RESET << endl;
37 }
38 else{
39 // If already exist, close the final.txt file
40 arquivo.close();
41
42 // Reopen file with ios:app mode that all
43 // operations will be performed at the end of the
44 // file
45 arquivo.open("final.txt", ios::app);
46
47 cout << GREEN << "Concatenating all files to final
48 .txt" << RESET << endl;
49 }
50
51 // Walk through all concatenated files and write at
52 // the end of final.txt file
53 for(auto line : all) arquivo << line;
54
55 // Close file
56 arquivo.close();
57
58 // Success message
59 cout << GREEN << "Concatenation successful.\n" <<
60 RESET;
61 }

```

A função concatenate é a função mais importante de todo o programa, ela é responsável por concatenar todos os dados de todos os arquivos. Primeiramente se percorre o vetor com o nome de todos os arquivos fornecidos pela função *read()* (linha 9), depois itera por todas as linhas do arquivos e vai salvando cada linha no vetor dinâmico *all* (linha 14), depois fecha o arquivo. Para concatenar todos os arquivos, primeiramente o programa teste se o arquivo *final.txt* (arquivo que irá ficar o resultado final da concatenação) já existe (linha 24), se o arquivo não existir, imprime na tela uma mensagem de erro ao abrir o arquivo, porém o próprio programa já cria um novo arquivo (linha 33), caso o arquivo exista, o sistema o fecha e reabre com o método *app*, que possibilita a escrita no final do arquivo.

```

Opening teste1.txt : SUCCESS
Opening teste2.txt : SUCCESS
File final.txt don't exist ... Creating a new one...
File final.txt created successfully
Concatenation successful.

```

Figura 4: Dois arquivos sendo lidos, porém o arquivo final.txt não existe, com isso o sistema o cria.

Main :

```

1  int main(){
2
3      // Vector to store all files name
4      vector<string> files;
5
6      // Read amount of files number
7      int n;
8
9      // Display intro message
10     intro();
11
12     // Read amount of files number
13     cin >> n;
14
15     // Read n files names
16     files = read(n);
17
18     // If can open all files, concatenate all
19     if(open_all(files))
20         concatenate(files);
21     else
22         cout << "Opening all files status : " << RED << "
                FAILED" << RESET << endl << "System finished" <<
                endl;
23 }

```

Parte principal de todo programa, responsável por operar todo o sistema, imprimindo a introdução na tela, lendo o nome dos arquivos e chamand as funções de abrir e concatenar os arquivos.

2 Outputs:

```
#####  
#                                     #  
#      FILE CONCATENATOR             #  
#                                     #  
#####  
#                                     #  
#      Aluno: Thiago Veras Machado   #  
#              M: 16/0146682         #  
# Aluno: Gabriel Cunha Bessa Vieira #  
#              M: 16/0120811         #  
#                                     #  
#####  
How many files you want to read ? : █
```

Figura 5: Introdução do programa.

```
1  Curso :  
2  - Organização de Arquivos  
3  Professor :  
4  - Oscar Gaidos  
5
```

Figura 6: Conteúdo de teste1.txt.

```
1  Alunos :  
2  - Thiago Veras  
3  - Gabriel Bessa  
4
```

Figura 7: Conteúdo de teste2.txt.


```
#####
#                                     #
#       FILE CONCATENATOR           #
#                                     #
#####
#                                     #
#       Aluno: Thiago Veras Machado #
#               M: 16/0146682        #
# Aluno: Gabriel Cunha Bessa Vieira #
#               M: 16/0120811        #
#                                     #
#####

How many files you want to read ? : 2
Input file 1 name (With extention) : teste1.txt
Input file 2 name (With extention) : teste2.txt
Opening teste1.txt : SUCCESS
Opening teste2.txt : SUCCESS
Concatenating all files to final.txt
Concatenation successful.
```

Figura 8: Todo o sistema funcionando.

```
1  Curso :
2  - Organização de Arquivos
3  Professor :
4  - Oscar Gaidos
5  Alunos :
6  - Thiago Veras
7  - Gabriel Bessa
8
```

Figura 9: Conteúdo de final.txt.

Exercício 2:

Pesquise e compare as diferenças entre um arquivo texto e um arquivo binário, do ponto de vista físico.

Arquivos físicos são uma coleção de bytes que são armazenados em um determinado dispositivo. No nível mais baixo de complexidade, tanto o arquivo texto e o arquivo binário são similares, pois ambos contém seus dados armazenados como uma série de bits 0's e 1's, mas suas estruturas internas são bem diferentes. Nos arquivos textos esses bits representam caracteres, já no arquivo binário, os bits representam dados personalizados, que se tentar abrir num arquivo texto, aparece uma sequência de símbolos.