BCC 201 - Introdução à Programação I

Arquivos de Texto

Guillermo Cámara-Chávez
UFOP

Arquivos I

- Podem armazenar grande quantidade de informação
- Dados são persistentes (gravados em disco)
- Acessso aos dados pode ser não seqüencial (acesso direto a registros em um banco de dados)
- Acesso à informação pode ser concorrente (mais de um programa ao mesmo tempo)

Nomes e extensões I

`arq.txt`	arquivo texto simples
`arq.c`	código fonte em C
`arq.pdf`	*portable document format*
`arq.html`	arquivo para páginas WWW

Tipos de arquivos I

- Arquivo texto - armazena caracteres que podem ser mostrados diretamente na tela ou modificados por um editor de textos simples. Por exemplo:
  `*** Arquivo texto. ***`
- Arquivo binário - seqüência de bits sujeita às convenções dos programas que o gerou (exemplos arquivos executáveis, arquivos compactados, arquivos de registros).

Diretório I

- também chamada de pasta
- contém arquivos e/ou outros diretórios

Caminhos absolutos ou relativos I

- Caminho absoluto
  - Descrição de caminho desde o diretório raiz:

```
/bin/emacs
/home/usr/arq.txt
```

- Caminho relativo
  - Descrição de um caminho desde o diretório corrente:

```
arq.txt
mc102/lab.c
```

Abrindo um arquivo para leitura I

- A chamada abaixo tenta abrir o arquivo `teste.txt`

```
if (fopen("teste.txt", "r") == NULL)
    perror("Erro ao abrir o arquivo \n");
else
    printf("Arquivo aberto para leitura. \n");
```

- Em caso de erro:
  - a função retorna NULL
  - a função `perror` exibe uma mensagem explicita

Lendo dados de um arquivo I

```
char c;
FILE *f = fopen("teste.txt", "r");
while(fscanf(f, "%c", &c) != EOF)
    printf("%c", c);
fclose(f);
```

- `fopen` retorna um stream pointer
- `fscanf` é semelhante à função `scanf`
- `fclose` fecha o arquivo

Escrevendo dados de um arquivo I

```
FILE *fr = fopen("teste.txt", "r");
FILE *fw = fopen("saida.txt", "w");
while(fscanf(fr, "%c", &c) != EOF)
    fprintf(fw, "%c", c);
fclose(fr);
fclose(fw);
```

- `fprintf` é semelhante à função `printf`

fopen I

```
FILE* fopen(const char *caminho, char *modo);
```

modo	operações	ponto no arquivo
r	leitura	início
r+	leitura e escrita	início
w	escrita	início
w+	leitura e escrita	início
a	escrita	final
a+	leitura	início
	escrita	final

Lendo um vetor de um arquivo I

Lêr um vetor de números inteiros que foi salvo dentro um arquivo de texto. O arquivo possui todos os elemento do arquivo assim como a dimensão do vetor. O formato do arquivo é como segue:

```
dimensao_vetor elem_1 elem_2 elem_3 ... elem_n
```

Lendo um vetor de um arquivo II

```
#include<stdio.h>
int main(){
    FILE *fr;
    int i, n, *v;
    fr = fopen("vetor.txt", "r");
    fscanf(fr, "%d", &n);
    v = (int*) calloc (n, sizeof(int));
    for(i = 0; i < n; i++)
        fscanf(fr, "%d", &v[i]);
    for(i = 0; i < n; i++)
        printf("%d ", v[i]);
    fclose(fr);
    free(v);
    return 0;
}
```

Escrevendo um vetor em um arquivo I

Escrever um vetor de inteiro em um arquivo seguindo o formato da questão anterior

Escrevendo um vetor em um arquivo II

```c
#include <stdio.h>
#include <stdlib.h>

int* Inserir(int n);
void EscreverVetor(FILE* pfile, int n, int* v);

int main()
{
    FILE *fw;
    int n, *v;
    fw = fopen("vetor.txt", "w");
    printf("Inserir dimensao do vetor \n");
    scanf("%d", &n);

    v = Inserir(n); // Inserir dados no vetor
    EscreverVetor(fw, n, v);// escrita no arquivo

    fclose(fw);
    free(v);
    return 0;
}
```

Escrevendo um vetor em um arquivo III

```
int* Inserir(int n)
{
    int i, *vet = NULL;
    vet = (int*) calloc (n, sizeof(int));
    printf("Inserir %d numeros \n", n);
    for (i = 0; i < n; i++)
    {
        scanf("%d", &vet[i]);
        /* scanf("%d", vet+i); */
    }
    return vet;
}
```

Escrevendo um vetor em um arquivo IV

```
void EscreverVetor(FILE* pfile, int n, int* v)
{
    int i;
    fprintf(pfile, "%d ", n); // dimensao
    for(i = 0; i < n; i++)
        fprintf(pfile, "%d ", v[i]);
}
```

Escrevendo uma matriz em um arquivo I

```c
#include <stdio.h>
#include <stdlib.h>
int** Inserir(int lin, int col);
void EscreverMatriz(FILE* pfile, int lin, int col, int** m);
int main()
{
    FILE *fw;
    int lin, col, **m = NULL;
    fw = fopen("matriz.txt", "w");
    printf("Inserir dimensao da matriz \n");
    scanf("%d %d", &lin, &col);

    m = Inserir(lin, col); // Inserir dados no vetor
    EscreverMatriz(fw, lin, col, m); // escrita no arquivo

    fclose(fw);
    if (m[0] != NULL) free(m[0]);
    if (m != NULL) free(m);
    return 0;
}
```

Escrevendo uma matriz em um arquivo II

```
int** Inserir(int lin, int col)
{
    int i, j, **mat = NULL;
    mat = (int**) calloc (lin, sizeof(int*));
    mat[0] = (int*) calloc (lin*col, sizeof(int));
    for (i = 1; i < lin; i++)
        mat[i] = &mat[0][col*i];

    printf("Inserir %d numeros \n", lin*col);

    for (i = 0; i < lin; i++)
        for (j = 0; j < col; j++)
            scanf("%d", &mat[i][j]); // formato matriz
            /*scanf("%d", *(mat+i)+j) formato ponteiro*/
    return mat;
}
```

Escrevendo uma matriz em um arquivo III

```
void EscreverMatriz(FILE* pfile, int lin, int col,
                    int** mat)
{
    int i, j;
    fprintf(pfile, "%d %d\n", lin, col); // dimensao
    for(i = 0; i < lin; i++)
    {
        for(j = 0; j < col; j++)
            fprintf(pfile, "%d ", mat[i][j]);
        fprintf(pfile, "\n");
    }
}
```

Exercicios I

Conta o número de brancos em cada linha do Arquivo “test.txt”

```
a b c
top10 methods to count spaces

1 3
```

Saída:

```
a b c
Brancos: 2
top10 methods to count spaces
Brancos: 4
Brancos: 0
1 3
Brancos: 3
```

Exercicios II

```c
#include <stdio.h>
#include <stdlib.h>
int main(){
    FILE *fr;
    int cont = 0, fim;
    char car;
    fr = fopen("test.txt", "r");
    do{ fim = fscanf(fr, "%c", &car);
        cont = 0;
        while(car != '\n' && fim != EOF)
        {
            printf("%c", car);
            if (car == ' ') cont++;
            fim = fscanf(fr, "%c", &car);
        }
        printf("\n Brancos: %d \n", cont);
    }while(fim != EOF);
    return 0;
}
```

Exercicios III

Criar uma matriz de 10×10. Preencher a matriz com valores aleatórios, depois salvá-la em um arquivo. Cada número deverá ter 4 decimais.

Exercicios IV

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define LIN 10
#define COL 10

void Preenche(float M[LIN][COL]);
void Escreve(FILE* pfile, float M[LIN][COL]);

int main()
{
    FILE *fw;
    float num[LIN][COL];
    fw = fopen("random.txt", "w");
    srand(time(NULL));
    Preenche(num);
    Escreve(fw, num);
    fclose(fw);
    return 0;
}
```

Exercicios V

```
void Escreve(FILE* pfile, float M[LIN][COL])
{
    int i, j;
    for (i = 0; i < LIN; i++)
    {
        for (j = 0; j < COL; j++)
            fprintf(pfile, "%5.4f ", M[i][j]);
        fprintf(pfile, "\n");
    }
}

void Preenche(float M[LIN][COL])
{
    int i, j;
    for (i = 0; i < LIN; i++)
        for (j = 0; j < COL; j++)
            M[i][j] = (float)rand() / RAND_MAX;
}
```

Exercicios VI

Gere uma estrutura `Aluno` com os seguintes dados: nome, idade, nota1, nota2. Insira n alunos, depois calcular a média de notas para cada aluno. Finalmente, salvar em um arquivo os dados da estrutura e a média de notas. Repita o processo para cada aluno.

Exercicios VII

```
typedef struct Aluno{
    char nome[100];
    int idade;
    double nota1, nota2;
}ALU;

void Inserir(ALU* alunos, int n);
void Escreve(FILE *pfile, ALU* alunos, int n);
```

Exercicios VIII

```
int main()
{
    FILE *fw;
    ALU* alu = NULL;
    int n;
    printf("Inserir numero de alunos");
    scanf("%d%*c", &n);
    alu = (ALU*) calloc (n, sizeof(ALU));

    fw = fopen("aluno.txt", "w");
    Inserir(alu, n);
    Escreve(fw, alu, n);
    fclose(fw);
    free(alu);

    return 0;
}
```

Exercicios IX

```
void Escreve(FILE *pfile, ALU *alu, int n)
{
    int i, pos;
    for (i = 0; i < n; i++)
    {
        pos = strlen(alu[i].nome) - 1;
        alu[i].nome[pos] = '\0';
        fprintf(pfile, "%s\t%d\t%lf\t%lf\t%lf\n",
                alu[i].nome, alu[i].idade,
                alu[i].nota1, alu[i].nota2,
               (alu[i].nota1+alu[i].nota2)/2);
    }
}
```

Exercicios X

```
void Inserir(ALU* alu, int n)
{
    int i;
    for (i = 0; i < n; i++)
    {
        printf("Cadastro Aluno %d \n", i+1);
        printf("Inserir nome: ");
        fgets(alu[i].nome, 100, stdin);
        printf("Inserir idade: ");
        scanf("%d", &alu[i].idade);
        printf("Inserir duas notas: ");
        scanf("%lf %lf%*c", &alu[i].nota1, &alu[i].nota2);
    }
}
```

Exercicios Propostos I

1. Considere um arquivo que possui uma lista de pares de números reais (valores de x e y). Cada linha deve possuir um par de valores separados por tabulação. Elabore um programa para calcular o valor da função $z = x^2 + y^2$, para cada par de valores, e colocar a tabela x y z resultado em outro arquivo. Este outro arquivo deve ter, em cada linha, uma tripla de valores x y z separados por tabulação (caracter `'\t'`)
2. Elabora um programa para ler um arquivo de texto de, no máximo, 100 linhas e criar um arquivo com as linhas de texto em ordem inversa. Utilize para tanto um vetor de *strings* para armazenar temporariamente as linhas. DICA, utilize a função `fgets` para ler uma linha completa do arquivo.

FIM