Laboratório de Algoritmos e Técnicas de Programação II

Aula 01 - Arquivo: definição, declaração e manipulação

Álvaro Magri Nogueira da Cruz

Agenda

Introdução I

Problema

- O armazenamento de dados em variáveis e *arrays* é **temporário**;
  - Encerrou o programa $\rightarrow$ dados **são perdidos**.

Solução

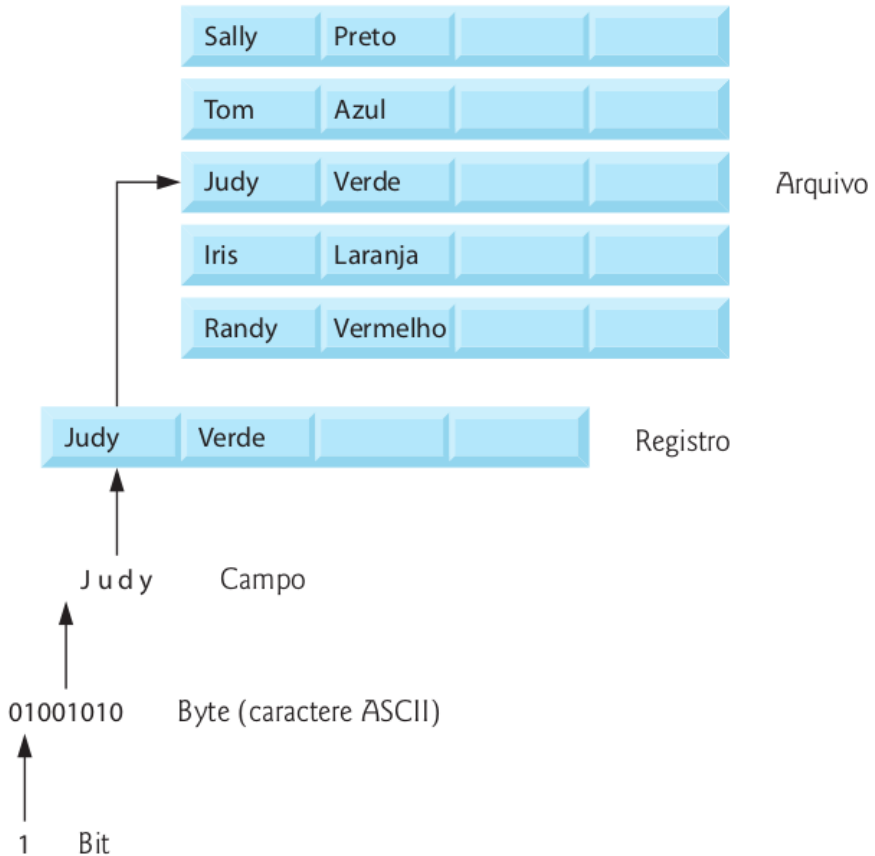
- **Arquivos** são utilizados na conservação **permanente de dados**;
  - São armazenados em dispositivos secundários (HDs, SSDs, CDs...);
  - Encerrou o programa $\rightarrow$ dados **armazenados**.

Registros I

O que é um registro?

- Em c é uma *struct*, composto por vários campos;
- E.g.: Um sistema de pagamento com os campos:
  1. Número de identificação (alfanumérico);
  2. Nome (alfabético);
  3. Endereço (alfanumérico);
  4. Salário (numérico).
- Assim, um registro é um grupo de **campos relacionados**;
- **Um arquivo é um grupo de registros relacionados**.

Registros II



Arquivos em C I

Observação

- Em C, cada arquivo é como uma sequência de bytes:

- Cada arquivo termina com um **marcador de fim de arquivo**;
- Quando um arquivo é aberto $\rightarrow$ um objeto é criado e um **stream** associado ao objeto;
- A biblioteca padrão é definida pela <stdio.h>

Arquivos em C II

Algumas funções importantes

1. **fgetc:** lê um caractere de um arquivo;
2. **getchar:** lê um caractere de um arquivo;
3. **fputc:** grava um caractere em um arquivo;
4. **putchar:** grava um caractere em um arquivo;
5. **fgets:** lê uma linha do arquivo;
6. **fputs:** escreve uma linha no arquivo;
7. **fscanf:** lê de um arquivo e não do teclado do computador;
8. **fprintf:** imprime em um arquivo e não na tela do computador;
9. **fread:** transfere um número especificado de bytes do local no arquivo especificado pelo ponteiro de posição do arquivo para uma área na memória que começa com o endereço especificado;
10. **fwrite:** transfere para um arquivo um número especificado de bytes a partir de um local especificado na memória.

Criação de um arquivo de acesso sequencial I

- C **não impõe nenhuma estrutura** a um arquivo;
- Conceitos como "registro" não fazem parte da linguagem;
- O que isso quer dizer???
  - O registro para um arquivo é uma **abstração do programador**;
  - O programador deverá criar a **sua lógica** para criação e leitura do arquivo;
  - Isso que dizer que seu arquivo pode ter a extensão que você quiser!
    - ".goku", ".varzea", ".labprogii", ".etc"

Criação de um arquivo de acesso sequencial II

Criar um arquivo

1. Primeiramente declaramos um ponteiro do tipo **FILE**:
   - **FILE** *nomeDoPonteiro;
2. Depois devemos abrir este o arquivo (se ele não existe, será criado):
   - nomeDoPonteiro=**fopen**("nomeArquivo.extensão", "modoOperação");
   - Modo de operação pode ser: r (read), w (write) e a (append) → mais utilizados.

Gravar um arquivo

1. **fprintf**(nomeDoPonteiro, "%tipo", variável);
2. **fclose**(nomeDoPonteiro);

Criação de um arquivo de acesso sequencial III

Modo	Descrição
r	Abre um arquivo existente para leitura.
w	Cria um arquivo para gravação. Se o arquivo já existe, descarta o conteúdo atual.
a	Acréscimo; abre ou cria um arquivo para gravação no final do arquivo.
r+	Abre um arquivo existente para atualização (leitura e gravação).
w+	Cria um arquivo para atualização. Se o arquivo já existe, descarta o conteúdo atual.
a+	Acréscimo: abre ou cria um arquivo para atualização; a gravação é feita no final do arquivo.
rb	Abre um arquivo existente para leitura no modo binário.
wb	Cria um arquivo para gravação no modo binário. Se o arquivo já existe, descarta o conteúdo atual.
ab	Acréscimo; abre ou cria um arquivo para gravação no final do arquivo no modo binário.
rb+	Abre um arquivo existente para atualização (leitura e gravação) no modo binário.
wb+	Cria um arquivo para atualização no modo binário. Se o arquivo já existir, descarta o conteúdo atual.
ab+	Acréscimo: abre ou cria um arquivo para atualização no modo binário; a gravação é feita no final do arquivo.

Figura 1: Modos de operação

Criação de um arquivo de acesso sequencial IV

Exemplo de criação de um arquivo

```c
#include <stdio.h>
int main(){
    int conta; /* numero da conta */
    char nome[30]; /* nome da conta */
    double saldo; /* saldo da conta */
    FILE *arquivo; /* ponteiro de arquivo*/
    if((arquivo = fopen("meuFile.unesp", "w")) == NULL){
            printf("Arquivo nao pode ser aberto\n");
    }
    else{
        printf("Digite o numero de conta:\n");
        scanf("%d", &conta);
        printf("Digite o nome:\n");
        scanf("%s", &nome);
        printf("Digite o saldo.\n");
        scanf("%lf", &saldo);
        fprintf(arquivo, "%d %s %lf \n", conta, nome, saldo)
    ;
        fclose(arquivo); /* fclose fecha arquivo */
    } /* fim do else */
    return 0;
} /* fim do main */
```

Leitura de dados de um arquivo de acesso sequencial I

- Os dados são armazenados em arquivos de modo que possam ser recuperados para processamento quando necessário;
- Para fazer a leitura de uma linha de um arquivo utilizamos a função **fscanf(arquivo, "%tipoDaLeitura, &variavel");**

Leitura de dados de um arquivo de acesso sequencial II

Exemplo de leitura de um arquivo

```c
#include <stdio.h>
int main(){
    int conta; /* numero da conta */
    char nome[30]; /* nome da conta */
    double saldo; /* saldo da conta */
    FILE *arquivo; /* ponteiro de arquivo*/
    if((arquivo = fopen("meuFile.unesp", "r")) == NULL){
            printf("Arquivo nao pode ser aberto\n");
    }
    else{
        printf("Iniciando a leitura do arquivo...\n");
        while(!feof(arquivo)){
            fscanf(arquivo, "%d %s %lf", &conta, &nome, &
    saldo);
        }
        fclose(arquivo); /* fclose fecha arquivo */
        printf("Conta: %d\nNome: %s\nSaldo: %lf\n", conta,
    nome, saldo);
    } /* fim do else */
    return 0;
} /* fim do main */
```

Criação de um arquivo de acesso aleatório I

Conceito

- Quando utilizamos *fprintf(fPtr, "%d", number)*, gravamos o inteiro na posição que o ponteiro *fPtr* está indicando;
- Se quiséssemos gravar necessariamente uma determinada quantidade e bytes, como um inteiro podemos utilizar:
  - *fwrite(&number, sizeof(int), 1, fPtr);*

Diferenças entre fprintf e fwrite

- No exemplo que demos:
  - No primeiro, **poderia** imprimir um único dígito, ou até 11 para um inteiro de 4 bytes;
  - No segundo, ele **sempre** gravará 4 bytes a partir da variável *number* para o arquivo representado pelo ponteiro *fPtr*.

Criação de um arquivo de acesso aleatório II

Exemplo de aplicação do acesso aleatório

- Crie um sistema de processamento de crédito capaz de armazenar até 100 registros de tamanho fixo. Cada registro deve consistir em um número de conta que será usado como chave de registro, um sobrenome, um nome e um saldo. O programa resultante deverá ser capaz de atualizar uma conta, inserir um novo registro de conta, excluir uma conta e listar todos os registros de conta em um arquivo de texto formatado para impressão. Use um arquivo de acesso aleatório.

Criação de um arquivo de acesso aleatório III

```c
#include <stdio.h>
/*definicao da estrutura dadosCliente*/
struct dadosCliente{
    int conta; /* numero da conta*/
    char nome[10]; /*nome da conta*/
    char sobrenome[15]; /* sobrenome da conta*/
    double saldo; /*saldo da conta*/
}; /*fim da estrutura dadosCliente*/
int main(){
    FILE *arquivo; /* ponteiro do arquivo banco.unesp*/
    struct dadosCliente cliente = {0,"","", 0.0}; /*cria
    dadosCliente*/
    /*fopen abre o arquivo; sai se nao puder abrir arquivo*/
    if ((arquivo = fopen("banco.unesp", "wb")) == NULL){
            printf("Arquivo nao pode ser aberto.\n");
    }
    else{
        for(int i = 1; i<=100; i++){  /* envia 100 registros
     vazios para arquivo */
            fwrite(&cliente, sizeof(struct dadosCliente), 1,
     arquivo);
        }
        fclose(arquivo); /* fclose fecha o arquivo */
    } /* fim do else */
    return 0; /* indica conclusao bem-sucedida */
} /* fim do main */
```

Criação de um arquivo de acesso aleatório IV

- Gravar em um arquivo de acesso aleatório.

```c
#include <stdio.h>
/*definicao da estrutura dadosCliente*/
struct dadosCliente{
    int conta; /* numero da conta*/
    char nome[10]; /*nome da conta*/
    char sobrenome[15]; /* sobrenome da conta*/
    double saldo; /*saldo da conta*/
}; /*fim da estrutura dadosCliente*/
int main(){
    FILE *arquivo; /* ponteiro do arquivo banco.unesp*/
    struct dadosCliente cliente = {0,"","", 0.0}; /*cria
    dadosCliente*/
    /*fopen abre o arquivo; sai se nao puder abrir arquivo*/
    if ((arquivo = fopen("banco.unesp", "rb+")) == NULL){
            printf("Arquivo nao pode ser aberto.\n");
    }
    else{
        /*exige que usuario especifique numero de conta */
        printf("\nDigite numero de conta (1 a 100, 0 para
    encerrar entrada): ");
        scanf("%d", &cliente.conta);
        /*usuario entra informacoes, copiadas para o arquivo
    */
```

Criação de um arquivo de acesso aleatório V

```
        while (cliente.conta != 0) {
            /*usuario digita sobrenome, nome e saldo */
            printf("\nDigite o nome: ");
            scanf(" %s", &cliente.nome);
            printf("\nDigite o sobrenome: ");
            scanf(" %s", &cliente.sobrenome);
            printf("\nDigite o saldo: ");
            scanf("%lf", &cliente.saldo);
            /*busca posicao no arquivo para registro
    especificado pelo usuario */
            fseek(arquivo,(cliente.conta-1)*sizeof(struct
    dadosCliente), SEEK_SET);
            /*grava informacao especificada pelo usuario no
    arquivo */
            fwrite( &cliente, sizeof( struct dadosCliente ),
     1, arquivo);
            /*permite que usuario informe outro numero de
    conta */
            printf("\nDigite numero de conta (1 a 100, 0
    para encerrar entrada): ");
            scanf("%d", &cliente.conta);
        } /* fim do while */
        fclose(arquivo); /* fclose fecha o arquivo */
    } /* fim do else */
    return 0; /* indica conclusao bem-sucedida */
} /* fim do main */
```

Criação de um arquivo de acesso aleatório VI

- Ler um arquivo de acesso aleatório.

```c
#include <stdio.h>
/*definicao da estrutura dadosCliente*/
struct dadosCliente{
    int conta; /* numero da conta*/
    char nome[10]; /*nome da conta*/
    char sobrenome[15]; /* sobrenome da conta*/
    double saldo; /*saldo da conta*/
}; /*fim da estrutura dadosCliente*/
int main(){
    FILE *arquivo; /* ponteiro do arquivo banco.unesp*/
    struct dadosCliente cliente = {0,"","", 0.0}; /*cria
    dadosCliente*/
    /*fopen abre o arquivo; sai se nao puder abrir arquivo*/
    if ((arquivo = fopen("banco.unesp", "rb+")) == NULL){
            printf("Arquivo nao pode ser aberto.\n");
    }
    else {
            printf( "%-6s%-16s%-11s%10s\n", "Conta", "Nome",
     "Sobrenome", "Saldo" );
            /* le todos os registros do arquivo (ate eof) */
            while ( !feof(arquivo) ) {
                fread( &cliente, sizeof( struct dadosCliente
     ), 1, arquivo);
```

Criação de um arquivo de acesso aleatório VII

```
                /* mostra registro */
                if ( cliente.conta != 0 ) {
                printf( "%-6d%-16s%-11s%10.2f\n", cliente.
    conta, cliente.nome, cliente.sobrenome, cliente.saldo);
                } /* fim do if */
             } /* fim do while */
             fclose(arquivo); /* fclose fecha o arquivo */
     } /* fim do else */
     return 0; /* indica conclusao bem-sucedida */
} /* fim do main */
```

- Agora conclua o programa!
- Faça com que:
  - Atualize contas (Dica: leia a conta que queira atualizar, atualize os valores na struct e depois grave novamente);
  - Exclua contas (Dica: substitua por uma conta em branco!);
  - Insira novas contas (Dica: verifique se a conta já existe. Senão, substitua a conta em branco pela a qual você quer criar).

Exercícios

DICA: fique à vontade em usar arquivo de acesso aleatório ou sequencial.

1. Escreva um programa que gere um vetor inteiro aleatório de n posições. Salve esse vetor em um arquivo. Depois abra o arquivo, leia-o e informe ao usuário qual o maior número, menor e a média dos números do vetor.
2. Escreva um programa para ser o diário de classe de um professor da UNESP. Ele deve permitir a entrada do número e matrícula do aluno (pode ser um número aleatório), sua presença em aula (0-100%) e sua média (0-10). O programa deve permitir cadastrar alunos, presença e média, assim como poder alterá-los. Imagine que o professor cadastre alguns alunos, depois feche o programa, para não ter que fazer tudo de novo, o programa deverá salvar e recuperar em arquivo as informações.

Referências I

1. **Deitel H. M., e Deitel P.J; "C: Como programar". 6.ed. Pearson Prentice Hall, 2011. 818p.**
2. Jean Paul Tremblay & Richard P. Bunt. "Ciência dos Computadores - Uma abordagem algorítmica". McGraw-Hill.
3. Jaime Evaristo. "Aprendendo a Programar / Programando em Turbo Pascal". Edufal - Editora da Univ. Federal de Alagoas. Maceió, 1996.
4. Harry Farrer et al. "Pascal Estruturado (da série "Programação Estruturada de Computadores")". Editora Guanabara Dois. Rio de Janeiro, 1985.
5. Stephen O'Brien. "Turbo Pascal 6 Completo e Total". Makron Books.

Referências II

6. Celes, W., Cerqueira, R., Rangel, J.L. "Introdução a Estrutura de Dados". Elsevier, 2004.
7. Feofiloff, P. "Algoritmos em Linguagem C". Elsevier, 2009. 208p.
8. Schildt, H. "C Completo e Total". 3ª ed. Pearson. 1996. 852p.