Universidade Positivo

Algoritmos de Programação

Registros / Estruturas

Prof.ª Mariane Cassenote

mariane.cassenote@up.edu.br

Conteúdo

- Registros / Estruturas

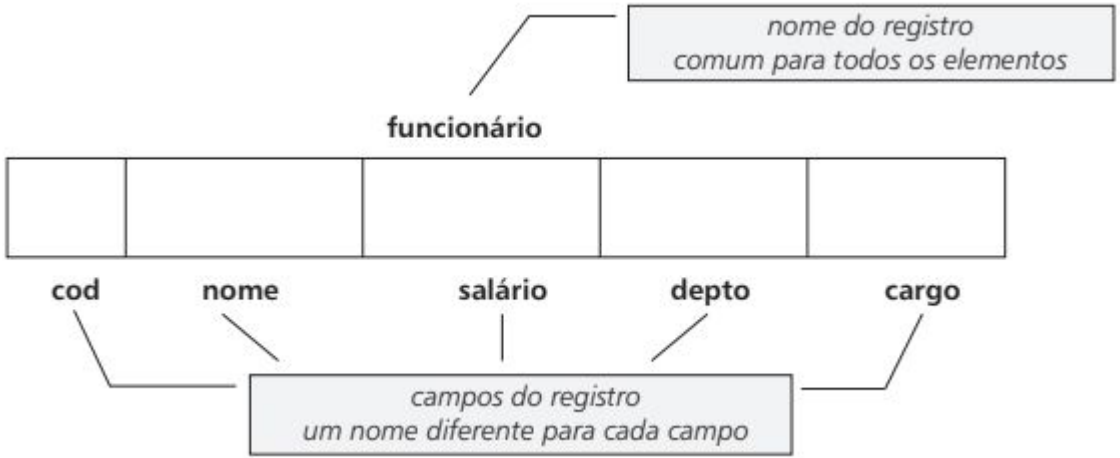
Ao final do componente curricular será possível aplicar o conceito de registros em problemas algorítmicos.

O que são registros?

- Também chamados de **estruturas (*structs*) heterogêneas**
- Estruturas referenciadas por um nome que fornecem uma maneira conveniente de agrupar variáveis que se correlacionam
- As variáveis agrupadas podem ser de tipos diferentes
- Podem agrupar variáveis **simples** (int, float, double, char) e **compostas** (vetores, matrizes, structs)

O que são registros?

- Os dados que integram um registro são chamados **campos**



EDELWEISS, Nina. Algoritmos e programação com exemplos em Pascal e C. Porto Alegre Bookman 2014 1 recurso on-line (Livros didáticos UFRGS 23).

Registros em C

Registros em C

Sempre criar acima da `main()`

- **Criação de tipo de registro**

```
struct <nome_do_registro> {
     <tipo_campo1> <nome_campo1>;
     <tipo_campo2> <nome_campo2>;
     …
     <tipo_campoN> <nome_campoN>;
};
```

```
struct funcionario {
     int cod;
     char nome[50];
     float salario;
     int depto;
     char cargo;
};
```

- **Acesso a um registro**

Sempre usar . (ponto) para acesso

```
<identificador_registro>.<nome_campo>
```

EDELWEISS, Nina. Algoritmos e programação com exemplos em Pascal e C. Porto Alegre Bookman 2014 1 recurso on-line (Livros didáticos UFRGS 23).

Registros em C

```c
#include <stdio.h>

// definição de estrutura sempre acima da main()

struct pessoa {
    char nome[50];
    int idade;
};

int main() {

    // criando uma estrutura p do tipo pessoa
    struct pessoa p;

    // preenchendo os dados da estrutura p

    printf("Digite o nome: ");
    scanf("%[^\n]s", p.nome);
    printf("Digite a idade: ");
    scanf("%d", &p.idade);

    // imprimindo os dados da estrutura p

    printf("%s tem %d anos \n", p.nome, p.idade);

    return 0;

}
```

```
> ./main
Digite o nome: Sirius
Digite a idade: 36
Sirius tem 36 anos
>
```

```c
#include <stdio.h>

// definição de estrutura sempre acima da main()

struct funcionario {
    int cod;
    char nome[50];
    float salario;
    int depto;
    char cargo;
};

int main() {

    // criando uma estrutura f do tipo funcionario

    struct funcionario f;

    // preenchendo os dados da estrutura f

    f.cod = 123;

    printf("Digite o nome: ");
    scanf("%[^\n]s", f.nome);

    printf("Digite o salario: ");
    scanf("%f", &f.salario);

    printf("Digite o cargo: ");
    scanf(" %c", &f.cargo);

    if (f.salario <= 2000.00) {
        f.depto = 1;
    } else {
        f.depto = 2;
    }

    // imprimindo os dados da estrutura f

    printf("\n --- Dados do funcionario --- \n");

    printf("Codigo: %d \n", f.cod);
    printf("Nome: %s", f.nome);
    printf("Salario: %.2f \n", f.salario);
    printf("Departamento: %d \n", f.depto);
    printf("Cargo: %c \n", f.cargo);

    return 0;

}
```

```c
#include <stdio.h>

// definição de estrutura sempre acima da main()

struct funcionario {
    int cod;
    char nome[50];
    float salario;
    int depto;
    char cargo;
};

int main() {

    // criando uma estrutura f do tipo funcionari

    struct funcionario f;

    // preenchendo os dados da estrutura f

    f.cod = 123;

    printf("Digite o nome: ");
    scanf("%[^\n]s", f.nome);
    printf("Digite o salario: ");
    scanf("%f", &f.salario);

    printf("Digite o cargo: ");
    scanf(" %c", &f.cargo);

    rio <= 2000.00) {
     = 1;

     = 2;

    os dados da estrutura f

     --- Dados do funcionario --- \n");

    digo: %d \n", f.cod);
    Nome: %s", f.nome);
    printf("Salario: %.2f \n", f.salario);
    printf("Departamento: %d \n", f.depto);
    printf("Cargo: %c \n", f.cargo);

    return 0;

}
```

```
> ./main
Digite o nome: Mariane
Digite o salario: 2500.00
Digite o cargo: B

 --- Dados do funcionario ---
Codigo: 123
Nome: Mariane
Salario: 2500.00
Departamento: 2
Cargo: B
>
```

Registros em C

Escrevendo com typedef

Registros em C

```c
struct funcionario { // definição da struct funcionario
      int cod;
      char nome[50];
      float salario;
      int depto;
      char cargo;
};

int main() {
   struct funcionario f; // criação da estrutura f
   ...
   return 0;
}
```

```c
typedef struct { // definição da struct funcionario
   int cod;
   char nome[50];
   float salario;
   int depto;
   char cargo;
} funcionario;

int main() {
   funcionario f; // criação da estrutura f
   ...
   return 0;
}
```

```c
#include <stdio.h>

// definição de struct sempre acima da main()

typedef struct {
    int cod;
    char nome[50];
    float salario;
    int depto;
    char cargo;
} funcionario;

int main() {

    // criando uma estrutura f do tipo funcionario

    funcionario f;

    // preenchendo os dados da estrutura f

    f.cod = 123;

    printf("Digite o nome: ");
    scanf("%[^\n]s", f.nome);

    printf("Digite o salario: ");
    scanf("%f", &f.salario);

    printf("Digite o cargo: ");
    scanf(" %c", &f.cargo);

    if (f.salario <= 2000.00) {
        f.depto = 1;
    } else {
        f.depto = 2;
    }

    // imprimindo os dados da estrutura f

    printf("\n --- Dados do funcionario --- \n");

    printf("Codigo: %d \n", f.cod);
    printf("Nome: %s", f.nome);
    printf("Salario: %.2f \n", f.salario);
    printf("Departamento: %d \n", f.depto);
    printf("Cargo: %c \n", f.cargo);

    return 0;
}
```

```c
#include <stdio.h>

// definição de struct sempre acima da main()

typedef struct {
    int cod;
    char nome[50];
    float salario;
    int depto;
    char cargo;
} funcionario;

int main() {

    // criando uma estrutura f do tipo funcionario

    funcionario f;

    // preenchendo os dados da estrutura f

    f.cod = 123;

    printf("Digite o nome: ");
    scanf("%[^\n]s", f.nome);
    printf("Digite o salario: ");
    scanf("%f", &f.salario);

    printf("Digite o cargo: ");
    scanf(" %c", &f.cargo);

    rio <= 2000.00) {
     = 1;

     = 2;

    os dados da estrutura f

     --- Dados do funcionario --- \n");

    digo: %d \n", f.cod);
    Nome: %s", f.nome);
    printf("Salario: %.2f \n", f.salario);
    printf("Departamento: %d \n", f.depto);
    printf("Cargo: %c \n", f.cargo);

    return 0;
}
```

```
> ./main
Digite o nome: Mariane
Digite o salario: 2658.98
Digite o cargo: C

 --- Dados do funcionario ---
Codigo: 123
Nome: Mariane
Salario: 2658.98
Departamento: 2
Cargo: C
>
```

Registros em C

Vetores em Registros

```c
#include <stdio.h>

// definição de struct sempre acima da main()

typedef struct {
    int cod;
    char nome[50];
    float salario[3];
    int depto;
    char cargo;
} funcionario;

int main() {

    // criando uma estrutura f do tipo funcionario

    funcionario f;

    // preenchendo os dados da estrutura f

    f.cod = 123;

    printf("Digite o nome: ");
    scanf("%[^\n]s", f.nome);

    for(int i = 0; i < 3; i++) {
        printf("Digite o salario %d: ", i);
        scanf("%f", &f.salario[i]);
    }

    printf("Digite o departamento: ");
    scanf("%d", &f.depto);

    printf("Digite o cargo: ");
    scanf(" %c", &f.cargo);

    // imprimindo os dados da estrutura f

    printf("\n --- Dados do funcionario --- \n");

    printf("Codigo: %d \n", f.cod);
    printf("Nome: %s", f.nome);

    printf("3 ultimos salarios: \n");
    for(int i = 0; i < 3; i++)
        printf("%.2f ", f.salario[i]);

    printf("\nDepartamento: %d \n", f.depto);
    printf("Cargo: %c \n", f.cargo);

    return 0;
}
```

```c
#include <stdio.h>

// definição de struct sempre acima da main()

typedef struct {
    int cod;
    char nome[50];
    float salario[3];
    int depto;
    char cargo;
} funcionario;

int main() {

    // criando uma estrutura f do tipo funcionario

    funcionario f;

    // preenchendo os dados da estrutura f

    f.cod = 123;

    printf("Digite o nome: ");
    scanf("%[^\n]s", f.nome);

    for(int i = 0; i < 3; i++) {
        printf("Digite o salario %d: ", i);
        scanf("%f", &f.salario[i]);
    }

    te o departamento: ");
    &f.depto);

    te o cargo: ");
     &f.cargo);

    dados da estrutura f

    -- Dados do funcionario --- \n");

    go: %d \n", f.cod);
    : %s", f.nome);

    timos salarios: \n");
    0; i < 3; i++)
    printf("%.2f ", f.salario[i]);

    printf("\nDepartamento: %d \n", f.depto);
    printf("Cargo: %c \n", f.cargo);

    return 0;
}
```

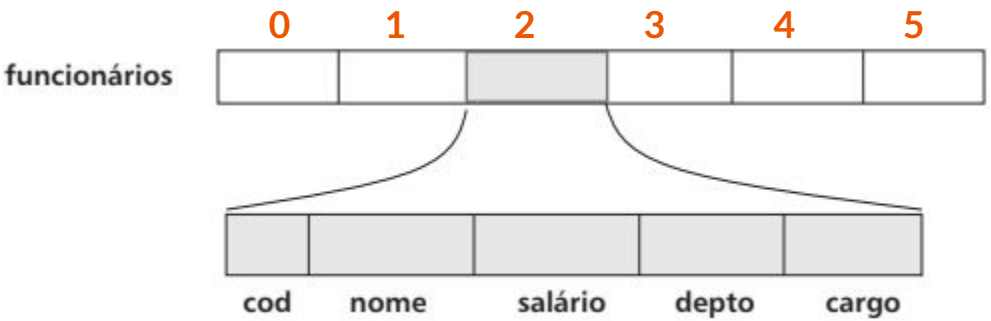
```
> ./main
Digite o nome: Mariane
Digite o salario 0: 1257.98
Digite o salario 1: 3659.78
Digite o salario 2: 2496.58
Digite o departamento: 12
Digite o cargo: A

 --- Dados do funcionario ---
Codigo: 123
Nome: Mariane
3 ultimos salarios:
1257.98 3659.78 2496.58
Departamento: 12
Cargo: A
>
```

Registros em C

Registros em Vetores

Vetor de registros

- Temos um vetor de 6 funcionários
- Cada elemento do vetor é um registro que contém as informações do funcionário



EDELWEISS, Nina. Algoritmos e programação com exemplos em Pascal e C. Porto Alegre Bookman 2014 1 recurso on-line (Livros didáticos UFRGS 23).

Em vetor de struct, ler string com scanf()

```c
#include <stdio.h>

// definição de struct sempre acima da main()

typedef struct {
    int cod;
    char nome[50];
    float salario;
    int depto;
    char cargo;
} funcionario;

int main() {

    // criando uma estrutura f do tipo funcionario
    funcionario f[3];

    // preenchendo os dados da estrutura f
    for (int i = 0; i < 3; i++) {

        printf("\n--- Cad. funcionario %d ---\n", i);

        printf("Digite o codigo: ");
        scanf("%d", &f[i].cod);

        // limpeza de buffer do teclado antes de ler cada string
        fflush(stdin);

        printf("Digite o nome: ");
        scanf("%[^\n]s", f[i].nome);

        printf("Digite o salario: ");
        scanf("%f", &f[i].salario);

        printf("Digite o departamento (inteiro): ");
        scanf("%d", &f[i].depto);

        printf("Digite o cargo (caractere): ");
        scanf(" %c", &f[i].cargo);

    }

    // imprimindo os dados da estrutura f

    for (int i = 0; i < 3; i++) {

        printf("\n --- Dados do funcionario --- \n");

        printf("Codigo: %d \n", f[i].cod);
        printf("Nome: %s \n", f[i].nome);
        printf("Salario: %.2f \n", f[i].salario);
        printf("Departamento: %d \n", f[i].depto);
        printf("Cargo: %c \n", f[i].cargo);

    }

    return 0;
```

Em vetor de struct, ler string com scanf()

```c
#include <stdio.h>

// definição de struct sempre acima da main()

typedef struct {
    int cod;
    char nome[50];
    float salario;
    int depto;
    char cargo;
} funcionario;

int main() {

    // criando uma estrutura f do tipo funcionario
    funcionario f[3];

    // preenchendo os dados da estrutura f
    for (int i = 0; i < 3; i++) {

        printf("\n--- Cad. funcionario %d -

        printf("Digite o codigo: ");
        scanf("%d", &f[i].cod);

        // limpeza de buffer do teclado antes de ler ca
        fflush(stdin);
```

```
> ./main

--- Cadastrando funcionario 0 ---
Digite o codigo: 001
Digite o nome: Chaves
Digite o salario: 6598.15
Digite o departamento (inteiro): 5
Digite o cargo (caractere): A

--- Cadastrando funcionario 1 ---
Digite o codigo: 002
Digite o nome: Chiquinha
Digite o salario: 5987.59
Digite o departamento (inteiro): 8
Digite o cargo (caractere): B

--- Cadastrando funcionario 2 ---
Digite o codigo: 003
Digite o nome: Quico
Digite o salario: 4987.26
Digite o departamento (inteiro): 3
Digite o cargo (caractere): C

 --- Dados do funcionario ---
Codigo: 1
Nome: Chaves
Salario: 6598.15
Departamento: 5
Cargo: A

 --- Dados do funcionario ---
Codigo: 2
Nome: Chiquinha
Salario: 5987.59
Departamento: 8
Cargo: B

 --- Dados do funcionario ---
Codigo: 3
Nome: Quico
Salario: 4987.26
Departamento: 3
Cargo: C
>
```

```c
tf("Digite o nome: ");
f("%[^\n]s", f[i].nome);

tf("Digite o salario: ");
f("%f", &f[i].salario);

tf("Digite o departamento (inteiro): ");
f("%d", &f[i].depto);

tf("Digite o cargo (caractere): ");
f(" %c", &f[i].cargo);


ndo os dados da estrutura f

t i = 0; i < 3; i++) {

tf("\n --- Dados do funcionario --- \n");

tf("Codigo: %d \n", f[i].cod);
tf("Nome: %s \n", f[i].nome);
tf("Salario: %.2f \n", f[i].salario);
tf("Departamento: %d \n", f[i].depto);
tf("Cargo: %c \n", f[i].cargo);
```

Para praticar

1. Crie um programa que cadastre em registros o nome, o peso e a altura de 5 esportistas. Apresente na tela os dados de cada esportista e seu IMC (deve haver um campo para o IMC no registro).

2. Crie um programa que armazene o resultado de 5 jogos realizados em uma rodada do campeonato brasileiro de futebol. Para cada jogo, será necessário armazenar as seguintes informações:

    a. Data do jogo;
    b. Time mandante;
    c. Time visitante;
    d. Gols do mandante;
    e. Gols do visitante.

Depois dos dados informados, o algoritmo deve escrever na tela os dados de cada jogo, indicando quem foi o vencedor ou se deu empate. Utilize um vetor de registros na sua implementação.

Para praticar

3. Suponha que um banco permita no máximo 5 transações enviadas por PIX em um dia. Construa um programa que receba do teclado o saldo disponível na conta do cliente. Em seguida, cadastre os dados de cada um dos 5 PIX enviados (chave e valor). A cada PIX cadastrado, apresente o saldo atual da conta do cliente. Caso o cliente não tenha dinheiro suficiente para enviar o PIX, exiba uma mensagem de erro e não faça a transação. Ao final, imprima os dados das transações e o saldo da conta do cliente.

Para praticar

4. Elaborar um programa que armazene o nome e a altura de 10 pessoas com o uso de registros. O programa deve usar um menu que execute as seguintes etapas:
    a. Cadastrar os 10 registros;
    b. Apresentar os registros (nome e altura) das pessoas com 1.50 m ou menores;
    c. Apresentar os registros (nome e altura) das pessoas com mais de 1.50 m;
    d. Apresentar os registros (nome e altura) das pessoas com mais de 1.5 m e menos de 1.80 m;
    e. Calcular a média das alturas e apresentar os registros (nome e altura) das pessoas que estão acima da média.

Adaptado de MANZANO, José Augusto Navarro Garcia; OLIVEIRA, Jayr Figueiredo de. Algoritmos: lógica para desenvolvimento de programação de computadores. 22. ed. São Paulo: Ed. Érica, 2009.

Universidade Positivo

Algoritmos de Programação

Registros / Estruturas

Prof.ª Mariane Cassenote

mariane.cassenote@up.edu.br