ALGORITMOS E PROGRAMAÇÃO DE COMPUTADORES

Disciplina: 113476

Profa. Carla Denise Castanho

Universidade de Brasília – UnB
Instituto de Ciências Exatas – IE
Departamento de Ciência da Computação – CIC

11. REGISTROS

Registros

- Até o momento, vimos agrupamentos de dados de um **mesmo tipo**, vetores e matrizes.
- Frequentemente, no entanto, precisamos agrupar um conjunto de dados **relacionados**, de **tipos não necessariamente iguais**.
- Ex.: Nome, idade e sexo de uma pessoa; coordenadas de um ponto; *status* do personagem de um jogo; atributos de um produto; etc...

Registros

- Para isso, utilizamos **registros**.
- Um registro é **novo tipo**, definido pelo programador, formado por um ou mais campos, que podem ter **tipos diferentes**.
- Por esse motivo, registros também são chamados de **variáveis compostas heterogêneas**.

Registros

- Observe o exemplo...
- Podemos ver que a variável *Nome* é do tipo literal; que *Código*, *Level*, *Vida*, *Ataque* e *Defesa* são do tipo inteiro; que *Taxa de Captura* é do tipo real; e que *Paralisado* é do tipo booleano.
- Todas essas variáveis são **relacionadas**, e estão agrupadas em um **novo domínio**.
- Dizemos então que, da mesma maneira que existem os tipos inteiro, real, literal, etc, agora existe um novo tipo, *Dados do Monstro*, e também **podemos criar variáveis desse novo tipo**.

Dados do Monstro	
Código:	6
Nome:	Charizard
Taxa de Captura	5,6%
Level:	36
Vida:	56
Ataque:	15
Defesa:	14
Paralisado:	Falso

Declarando Registros

- Em pseudocódigo, temos duas formas de declarar registros. Podemos declarar diretamente no tipo da variável...

Exemplo - Declaração de Registro (primeira forma).

```
Algoritmo Batalha

Variáveis
    jogador, inimigo : registro (cod : inteiro, nome : literal,
        txcap : real, level, vida, atq, def : inteiro,
        paralis : booleano)

Início
    ...
Fim
```

Declarando Registros

- Ou uma seção separada, para que o tipo possa ser utilizado por várias variáveis...

Exemplo - Declaração de Registro (segunda forma).

```
Algoritmo Batalha
Definições
    TipoDadosMonstro : registro (cod : inteiro, nome : literal,
        txcap : real, level, vida, atq, def : inteiro,
        paralis : booleano)

Variáveis
    jogador, inimigo : TipoDadosMonstro

Início
    ...
Fim
```

variáveis

tipo

Declarando Registros

- Em C, temos a palavra chave *struct*, que pode ser usada de várias maneiras diferentes...

Declarando registros em C - Structs Anônimas

```c
#include <stdio.h>

int main () {
    /* declara as variaveis "jogador" e "inimigo", do tipo struct definido abaixo */
    struct {
        int cod;
        char nome[30];
        float txcap;
        int level, vida, atq, def;
        int paralis;
    } jogador, inimigo;
    ...

    return 0;
}
```

Estamos criando o novo tipo ao mesmo tempo em que declaramos as variáveis. Essa forma é equivalente à primeira forma em pseudocódigo.

Observe que não precisamos dar um nome para o novo tipo, por isso, dizemos que essa é uma *struct anônima*.

Declarando Registros

Declarando registros em C - Structs com Nomes

```c
#include <stdio.h>

/* declara uma nova struct, chamada DadosMonstro */
struct DadosMonstro {
    int cod;
    char nome[30];
    float txcap;
    int level, vida, atq, def;
    int paralis;
};

int main () {
    /* declara as variaveis "jogador" e "inimigo", do tipo struct DadosMonstro */
    struct DadosMonstro jogador, inimigo;

    ...

    return 0;
}
```

Nesse exemplo, primeiro nós declaramos uma nova struct, e demos um nome para ela: "DadosMonstro".

Em seguida, nós utilizamos a struct "DadosMonstro" como se fosse qualquer outro tipo para declarar duas variáveis. Observe que é necessário colocar a palavra chave "struct" na declaração.

Declarando Registros

- Para não termos que colocar a palavra chave *struct* na declaração de variáveis, podemos utilizar a palavra chave *typedef*.
- Essa é uma forma na Linguagem C de **associar um identificador a um tipo**. Por exemplo:

Definindo novos tipos em C - *typedef*

```c
#include <stdio.h>

/* a partir da declaracao abaixo, "inteiro" eh um novo tipo, sinonimo de int */
typedef int inteiro;

int main () {
    /* declara dois inteiros */
    inteiro a, b;
    ...
    return 0;
}
```

A notação:

`typedef <tipo> <identificador>;`

Define que "identificador" passa a ser sinônimo de "tipo".

- Observe o exemplo no próximo slide para *structs*...

Declarando Registros

Declarando registros em C - Utilizando *typedef*

```c
#include <stdio.h>

/* declara um novo tipo, chamado DadosMonstro, que é uma struct */
typedef struct {
    int cod;
    char nome[30];
    float txcap;
    int level, vida, atq, def;
    int paralis;
} DadosMonstro;

int main () {
    /* declara as variaveis "jogador" e "inimigo", do tipo DadosMonstro */
    DadosMonstro jogador, inimigo;

    ...

    return 0;
}
```

Observe a diferença. Aqui, nós estamos dizendo que o identificador "DadosMonstro" é um tipo, e esse tipo é uma struct.

Em seguida, estamos declarando duas variáveis do tipo "DadosMonstro". Observe que não devemos utilizar a palavra chave "struct" na declaração.

Acessando Campos do Registro

- Para acessar uma das variáveis internas do registro, i.e., um de seus campos, utilizamos a notação:

```
<nome da variável>.<nome do campo>
```

- Por exemplo:

```
inimigo.defesa = 100;
```

Acessando Campos do Registro

Exemplo - Acessando Campos do Registro

```
Algoritmo Batalha
Definições
    TipoDadosMonstro : registro (cod : inteiro, nome : literal,
        txcap : real, level, vida, atq, def : inteiro,
        paralis : booleano)
Variáveis
    jogador, inimigo : TipoDadosMonstro

Início
    inimigo.cod ← 6
    inimigo.nome ← "Charizard"
    inimigo.txcap ← 0.056
    inimigo.level ← 36
    ...

Fim
```

Acessando Campos do Registro

Exemplo - Acessando Campos do Registro

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

typedef struct {
    int cod;
    char nome[30];
    float txcap;
    int level, vida, atq, def;
    int paralis;
} DadosMonstro;

int main () {
    DadosMonstro jogador, inimigo;

    inimigo.cod = 6;
    strcpy(inimigo.nome, "Charizard");
    inimigo.txcap = 0.056;
    inimigo.level = 36;

    ...

    return 0;
}
```

Registros

- Vimos que **um registro é um tipo**, como outro qualquer, com a vantagem de **agrupar** dados de vários tipos diferentes.
- Portanto, podemos:
  - criar **vetores**, **matrizes** e **ponteiros** de registros;
  - utilizar registros em **parâmetros** e no **retorno de funções**;
  - criar **campos de registro** que também são registros;
- Vamos ver alguns exemplos...

Vetores de Registros

Exemplo - Vetores de Registro

```
Algoritmo FolhaPagamento
Definições
    Funcionario : registro (matricula : inteiro, nome, sexo, cargo : literal,
        salario : real, nro_filhos : inteiro)
Variáveis
    funcs : vetor [100] de Funcionario

Início
    Para i ← 0 até 99 faça
        Leia (funcs[i].matricula)
        Leia (funcs[i].nome)
        Leia (funcs[i].sexo)
        Leia (funcs[i].cargo)
        Leia (funcs[i].salario)
        Leia (funcs[i].nro_filhos)
    Fim-Para
    ...
Fim
```

Vetores de Registros

Exemplo - Vetores de Registro

```c
#include <stdio.h>

typedef struct {
    int matricula;
    char nome[30], sexo, cargo[100];
    float salario;
    int nro_filhos;
} Funcionario;

int main () {
    Funcionario funcs[100];
    int i;

    for (i = 0; i < 100; i++) {
        scanf("%d", &funcs[i].matricula);
        scanf("%s", funcs[i].nome);
        scanf("%c", &funcs[i].sexo);
        scanf("%s", funcs[i].cargo);
        scanf("%f", &funcs[i].salario);
        scanf("%d", &funcs[i].nro_filhos);
    }
    ...

    return 0;
}
```

Registros e Funções

Exemplo - Registros como parâmetros e como retorno de funções.

```c
#include <stdio.h>

typedef struct {
    float x, y;
} Ponto;

Ponto somaVetorial(Ponto a, Ponto b) {
    Ponto resultado;
    resultado.x = a.x + b.x;
    resultado.y = a.y + b.y;

    return resultado;
}

int main () {
    Ponto p1, p2, p3;
    p1.x = 0.1;
    p1.y = 0.2;
    p2.x = 6;
    p2.y = 5;
    p3 = somaVetorial(p1, p2);

    printf("Resultado: (%f, %f).\n", p3.x, p3.y);

    return 0;
}
```

Ponteiros para Registros

Exemplo - Ponteiros para Registros

```c
#include <stdio.h>

typedef struct {
    int dia, mes, ano;
} Data;

void incrementaMes(Data *data) {
    (*data).mes = (*data).mes + 1;
    if ((*data).mes == 13) {
        (*data).dia = 1;
        (*data).mes = 1;
        (*data).ano = (*data).ano + 1;
    }
}

int main () {
    Data data;
    data.dia = 31;
    data.mes = 12;
    data.ano = 2015;

    incrementaMes(&data);
    printf("Nova data: %d/%d/%d.\n", data.dia, data.mes, data.ano);

    return 0;
}
```

Ponteiros para Registros

- No caso de **ponteiros para registros**, a Linguagem C introduz o **operador "->"**, (chamado de operador "seta") para facilitar a vida do programador.

- O comando:

```
ptr_para_registro -> campo
```

- é equivalente a:

```
(*ptr_para_registro).campo
```

- Veja o mesmo exemplo anterior utilizando este operador...

Ponteiros para Registros

Exemplo - Operador seta

```c
#include <stdio.h>

typedef struct {
    int dia, mes, ano;
} Data;

void incrementaMes(Data *data) {
    data -> mes = data -> mes + 1;
    if (data -> mes == 13) {
        data -> dia = 1;
        data -> mes = 1;
        data -> ano = data -> ano + 1;
    }
}

int main () {
    Data data;
    data.dia = 31;
    data.mes = 12;
    data.ano = 2015;

    incrementaMes(&data);
    printf("Nova data: %d/%d/%d.\n", data.dia, data.mes, data.ano);

    return 0;
}
```

Registros como Campos de Registros

Exemplo - Registros de registros

```c
#include <stdio.h>

typedef struct {
    int dia, mes, ano;
} Data;

typedef struct {
    int matricula;
    char nome[30];
    Data dt_nascimento;
} Funcionario;

void le_data(Data *data) {
    scanf("%d%d%d", &data->dia, &data->mes, &data->ano);
}

int main () {
    Funcionario funcs[100];
    int i;

    for (i = 0; i < 100; i++) {
        scanf("%d", &funcs[i].matricula);
        scanf("%s", funcs[i].nome);
        le_data(&funcs[i].dt_nascimento);
    }
    ...

    return 0;
}
```

Registros – Exercício

- Temos um empresa que começou a ser informatizada, queremos cadastrar no máximo 100 funcionários com as seguintes informações de cada funcionário: Nome; Sexo; Salário; Matrícula; Endereço e Cargo.
- Queremos também saber o salario médio dos funcionários da empresa, o salario médio dos homens, e o salario médio das mulheres da empresa.
- Faça um algoritmo para o problema acima utilizando vetor de registros. Leia o número de funcionários *N*, que será menor ou igual a 100.