Algoritmos de Programação 2

Professor: José Eurípedes Ferreira de Jesus Filho
jeferreirajf@gmail.com

Universidade Federal de Jataí – UFJ

Dados mais complexos!

Introdução

- Neste momento do curso, sabemos como representar diversos tipos de dados no computador.

- Como ficaria a **representação de uma data**?

- E se fôssemos representar uma **ficha cadastral** com nome, email, cpf, senha, data de nascimento, etc?

Registros

- Podemos definir **tipos compostos**, que também podem ser chamados de registros.

- Os registros geralmente são formados por mais de um tipo de **dado primitivo** e por isso o nome “**tipo composto**”.

- Um registro também pode possuir vetores, matrizes e/ou outras estruturas dentro dele.

Registros

- Em um **bom uso de registro**, o conjunto de variáveis possuem alguma relação entre si.

- Um exemplo são os dados de uma pessoa.

NOME DA PESSOA			NACIONALIDADE
ENDEREÇO RESIDÊNCIA			
MUNICÍPIO	UF	CEP	TELEFONE
CPF	Nº REGISTRO NACIONAL DE ESTRANGEIRO – RNE		DATA VALIDADE – RNE
SEXO	ESTADO CIVIL	REGIME CASAMENTO	

Registros

- Antes, vamos aprender um comando super legal!

```
#include <stdio.h>

typedef int nivel;

int main(){
    nivel jogador1 = 10;

    printf(“%d”, jogador1);

    return(0);
}
```

Registros

- Agora vamos aprender a declarar um registro:

```
int main(){

    struct {
        int dia;
        int mes;
        int ano;
    } data;

    data.dia = 05;
    data.mes = 02;
    data.ano = 1988;

    return(0);
}
```

Registros

- Agora vamos aprender a criar um tipo baseado em um registro:

```
typedef struct s_data {
    int dia;
    int mes;
    int ano;
} Data;

int main(){

    Data data1;
    Data data2;

    return(0);
}
```

Registros

- Como ficaria misturando vetores?

```
typedef struct s_data {
    int dia;
    int mes;
    int ano;
} Data;

int main(){

    Data feriados[50];

    feriado[0].dia = 25; feriado[0].mes = 12; feriado[0].ano = 2022;

    return(0);
}
```

Registros

- Ou ainda:

```
typedef struct s_data {
    int dia;
    int mes;
    int ano;
} Data;

typedef struct s_agenda {
    char** descricao;
    Data* data;
} Agenda;

int main(){


    return(0);
}
```

Sumarizando

- Registros
  - ✓ Estruturas formadas por uma ou mais variáveis que podem ser de tipos diferentes.
  - ✓ As variáveis que formam uma estrutura possuem relação entre si.
- Registros em C
  - ✓ Utilizamos a palavra reservada **`struct`** para definirmos uma estrutura.
  - ✓ Se quisermos definir um novo tipo, devemos utilizar **`typedef`**.

Exercício

- Desenvolva o código C de um programa que consiga representar pontos 3D e figuras 3D no plano cartesiano tridimensional. Construa um triângulo 2D com essa representação.

Exercício

- Desenvolva uma função em C que recebe duas datas e calcula quantos dias de diferença existem entre elas. Para isso, crie o tipo data.

Exercício

- Uma conta bancária geralmente é descrita por um número de conta e uma agência. Além disso, uma conta possui seu saldo e está vinculada a uma pessoa. A pessoa por sua vez, possui seu nome, seu RG e seu CPF para identificá-la. Desenvolva um programa que lê do teclado os dados de uma pessoa e os dados de uma conta bancária. Imprima na tela o nome da pessoa, o número da conta e o seu saldo.

Exercício

- Crie uma estrutura chamada **`Colaborador`** com os campos para nome, idade e salário. Crie uma função que recebe um nome, uma idade e um salário e então retorna um **`Colaborador`**. Crie uma função que recebe um **`Colaborador`** por parâmetro e imprime os dados na tela.

Exercício

- Crie uma estrutura chamada **`Departamento`** com os campos para nome, identificador, quantidade de funcionários e os funcionários. Crie uma função que recebe um nome e um identificador e então retorna um **`Departamento`** com 0 funcionários e um array vazio de funcionários. Crie também uma função que recebe um departamento e um funcionário e então aloca esse funcionário ao departamento. Crie ainda uma função que recebe um departamento e um nome de funcionário e caso esse funcionário trabalhe no departamento, ele seja removido. Crie também uma função que recebe um **`Departamento`** e imprime todos os dados na tela. Dica: Utilize o tipo **`Colaborador`** do exercício anterior.

Exercício

- Crie uma estrutura chamada **`Empresa`** com os campos para nome, cnpj, quantidade de departamentos e os departamentos. Crie uma função que consiga construir uma **`Empresa`**, que consiga criar um **`Departamento`** na **`Empresa`** e que consiga alocar um **`Colaborador`** no **`Departamento`** de uma **`Empresa`**. Crie ainda uma função que imprima os dados de uma **`Empresa`** na tela. Dica: utilize os tipos **`Departamento`** e **`Colaborador`** dos exercícios anteriores.