Aula 03 - Estrutura de dados 1 – Uniube

Prof. Marcos Lopes

34 9 9878 0925

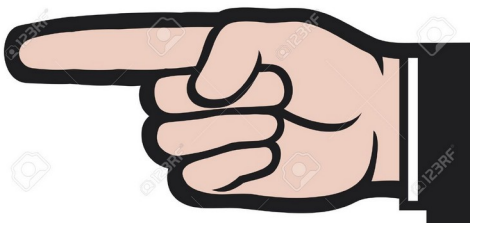
Registros e structs:

Um registro (= record) é um pacote de variáveis, possivelmente de tipos diferentes.

Cada variável é um campo (= field) do registro.

Na linguagem C, registros são conhecidos como structs (o nome é uma abreviatura de structure).

- Definição e manipulação de structs
- Structs e ponteiros
- Perguntas e respostas



Definição e manipulação de structs:

O seguinte exemplo declara (ou seja, cria) um registro x com três campos que pode ser usado para armazenar datas:

```
struct {
   int dia;
   int mes;
   int ano;
} x;
```

É uma boa ideia dar um nome à classe de todos os registros de um mesmo tipo. No nosso exemplo, acho que dma é um nome apropriado:

```
struct dma {
   int dia;
   int mes;
   int ano;
};
struct dma x;  // um registro x do tipo dma
struct dma y;  // um registro y do tipo dma
```

Definição e manipulação de structs: continuação

Para se referir a um campo de um registro, basta escrever o nome do registro e o nome do campo separados por um ponto:

```
x.dia = 31;
x.mes = 12;
x.ano = 2018;
```

Registros podem ser tratados como um novo tipo-de-dados. Depois da seguinte definição, por exemplo, poderemos passar a dizer data no lugar de struct dma:

```
typedef struct dma data;
data x, y;
```

Exemplo. A seguinte função calcula a data de fim de um evento ao receber a data de início do evento e a duração do evento em dias.

```
data fimEvento (data inicio, int duracao) {
   data fim;
   . . .
   . . .
   fim.dia = ...
   fim.mes = ...
   fim.ano = ...
   return fim;
}
```

O código foi omitido porque é um tanto enfadonho, uma vez que deve levar em conta o número de dias em diferentes meses e os anos bissextos.

Eis como a função fimEvento poderia ser usada:

```
int main (void) {
   data a, b;
   scanf ("%d %d %d", &a.dia, &a.mes, &a.ano);
   // &a.dia significa &(a.dia)
   int dura;
   scanf ("%d", &dura);
   b = fimEvento (a, dura);
   printf ("%d %d %d\n", b.dia, b.mes, b.ano);
   return EXIT_SUCCESS;
}
```

Exercícios 1

a) Complete o código da função fimEvento do slide anterior. Considere que todo ano tem 365 dias e todo mês tem 30 dias.

b) Escreva uma função que decida se uma dada struct do tipo dma representa uma data válida.

c) Escreva uma função que receba duas structs do tipo dma, cada uma representando uma data válida, e devolva o número de dias que decorreram entre as duas datas.

d) Escreva uma função que receba um número inteiro que representa um intervalo de tempo medido em minutos e devolva o número equivalente de horas e minutos (por exemplo, 131 minutos equivalem a 2 horas e 11 minutos). Use uma struct como a seguinte:

```
struct hm {
   int horas, minutos;
};
```

Structs e ponteiros:

Cada registro tem um endereço na memória do computador. O endereço de um registro pode ser armazenado em um ponteiro.

Dizemos que um tal ponteiro aponta para o registro (struct). Por exemplo,

```
data *p;          // p é um ponteiro para registros dma
data  x;
p = &x;           // agora p aponta para x
(*p).dia = 31;    // mesmo efeito que x.dia = 31
```

A expressão p->dia é uma abreviatura muito útil de (*p).dia . Portanto, a última linha do código acima pode ser escrita assim:

```
p->dia = 31;
```

Exercícios 2

a) Defina um registro empregado para armazenar os dados (nome, sobrenome, data de nascimento, CPF, data de admissão, salário) de um colaborador de sua empresa. Defina um vetor de empregados para armazenar os colaboradores de sua empresa (cerca de 50).

b) Um racional é qualquer número da forma p/q, sendo p um inteiro e q um inteiro não nulo. É conveniente representar um racional por um registro:

```
typedef struct {
   int p, q;
} racional;
```

Vamos convencionar que o campo q de todo racional é estritamente positivo e que o máximo divisor comum dos campos p e q é 1. Escreva funções

reduz, que receba inteiros a e b e devolva o racional que representa a/b;
neg, que receba um racional x e devolva o racional −x;
soma, que receba racionais x e y e devolva o racional que representa a soma de x e y;
produ, que receba racionais x e y e devolva o racional que representa o produto de x por y;
quoci, que receba racionais x e y e devolva o racional que representa o quociente de x por y.