Structs

Structs

- Até agora, vimos uma estrutura de dados: vetor
  - Uma string é um vetor de caracteres; uma matriz é um vetor de duas dimensões
- Propriedades importantes de um vetor:
  - Todos os elementos de um vetor são do mesmo tipo
  - Para selecionar um elemento de um vetor, especificamos a posição (índice) do elemento
- Usamos uma **struct** para armazenar uma coleção de dados de tipos possivelmente diferentes
- Propriedades importantes de uma struct:
  - Os elementos (**membros**) de uma struct podem ser de tipos diferentes
  - Para selecionar um elemento de uma struct, especificamos o nome do elemento

Structs - Declaração de variáveis

- Para declarar variáveis que são structs, podemos escrever:

```c
struct {
  int dia;
  int mes;
  int ano;
} data1, data2;
```

```c
struct {
  int id;
  char nome[30];
  double salario;
} funcionario1, funcionario2;
```

- Representação de `data1` na memória do computador:

- Os nomes dos membros de uma struct não conflitam com outros nomes de fora da struct

Structs - Inicialização de variáveis

- Assim como vetores, variáveis que são structs podem ser inicializadas quando declaradas

```c
struct {
  int dia;
  int mes;
  int ano;
} data1 = { 9, 11, 2003 },
  data2 = { 3, 1, 2008 };
```

```c
struct {
  int id;
  char nome[30];
  double salario;
} funcionario1 = { 51, "Jose Silva", 5000.00 },
  funcionario2 = { 89, "Maria Souza", 5000.00 };
```

Structs - Operações

- Para acessar um membro de uma variável que é uma struct, escrevemos o nome da variável seguido de um . seguido do nome do membro

```c
struct {
  int dia;
  int mes;
  int ano;
} data1, data2;
```

```c
struct {
  int id;
  char nome[30];
  double salario;
} funcionario1, funcionario2;
```

```c
printf("Dia: %d\n", data1.dia);
printf("Nome do funcionario: %s\n", funcionario1.nome);
```

Structs - Operações

- Podemos atribuir valores aos membros de uma variável que é uma struct e usá-los em operações aritméticas (quando cabível)

```c
struct {
  int dia;
  int mes;
  int ano;
} data1, data2;
```

```c
struct {
  int id;
  char nome[30];
  double salario;
} funcionario1, funcionario2;
```

```c
data1.dia = 3;
media = (funcionario1.salario + funcionario2.salario) / 2;

scanf("%d", &data2.mes);
scanf("%lf", &funcionario1.salario);
```

Structs - Operações

- Diferente do que vale para vetores, podemos usar o operador `=` para atribuir uma struct a outra struct - desde que as structs sejam de tipos compatíveis

```
struct {
  int dia;
  int mes;
  int ano;
} data1, data2;
```

```
struct {
  int id;
  char nome[30];
  double salario;
} funcionario1, funcionario2;
```

```
data1 = data2;
funcionario2 = funcionario1;
```

- O efeito do comando `data1 = data2;` é copiar `data2.dia` para `data1.dia`, `data2.mes` para `data1.mes` e `data2.ano` para `data1.ano`.

Structs - Nomeando tipos

- Para passar uma variável que é uma struct como argumento para uma função, precisamos definir um nome que indique o tipo desta variável
- **Opção 1**: Definir uma struct tag

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

```c
struct funcionario {
  int id;
  char nome[30];
  double salario;
} funcionario1, funcionario2;
```

```c
struct data data1, data2;                       /* não é possível omitir */
struct funcionario funcionario3, funcionario4; /* a palavra struct! */
```

- Nas declarações acima, não é possível omitir a palavra `struct`!

Structs - Nomeando tipos

- Para passar uma variável que é uma struct como argumento para uma função, precisamos definir um nome que indique o tipo desta variável
- **Opção 2**: Definir um novo tipo usando `typedef`

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

```c
typedef struct funcionario {
  int id;
  char nome[30];
  double salario;
} Funcionario;
```

```c
Data data1, data2;
Funcionario funcionario1, funcionario2;
```

Structs - Vetores de Structs

- Podemos criar vetores de elementos de um tipo struct:

```c
typedef struct funcionario {
  int id;
  char nome[30];
  double salario;
} Funcionario;
```

```c
Funcionario equipe[4];

equipe[0].id = 5;
strcpy(equipe[0].nome, "Renato");
equipe[0].salario = 5000.00
```

Exercício

- Crie uma estrutura representando os alunos de um determinado curso. A estrutura deve conter a matrícula do aluno, nome, nota da primeira prova, nota da segunda prova e nota da terceira prova.
  - Permita ao usuário entrar com os dados de 5 alunos.
  - Encontre o aluno com maior nota da primeira prova.
  - Encontre o aluno com maior média geral.
  - Encontre o aluno com menor média geral.
  - Para cada aluno, diga se ele foi aprovado ou reprovado, considerando o valor 6 para aprovação.