Struct

Prof. Lilian Berton

São José dos Campos, 2019

Structs em C

- Já vimos dois tipo básicos de estrutura em C: **vetores e matrizes**.

- A vantagem desse tipo de estrutura é não precisar trabalhar individualmente com cada variável. Ao invés disso**, trabalhamos com um grupo de variáveis mas todas são do mesmo tipo (int, char, float)**.

Vetor **A**

12	9	10	16	25	13	20	14		14
0	1	2	3	4	5	6	7	8	9

Matriz **B**

	0	1	2	3	4
0	A	E	I	O	U
1	X	A	Y	I	O
2	G	A	S	J	A
3	A	F	D	F	A
4	T	X	C	Q	E

- Em muitas aplicações, precisamos trabalhar com vários elementos de tipos diferentes, para isso existe uma estrutura chamada *struct*, onde **é possível agrupar os mais variados tipos de variáveis e formar um bloco de informação, que é a *struct*.**

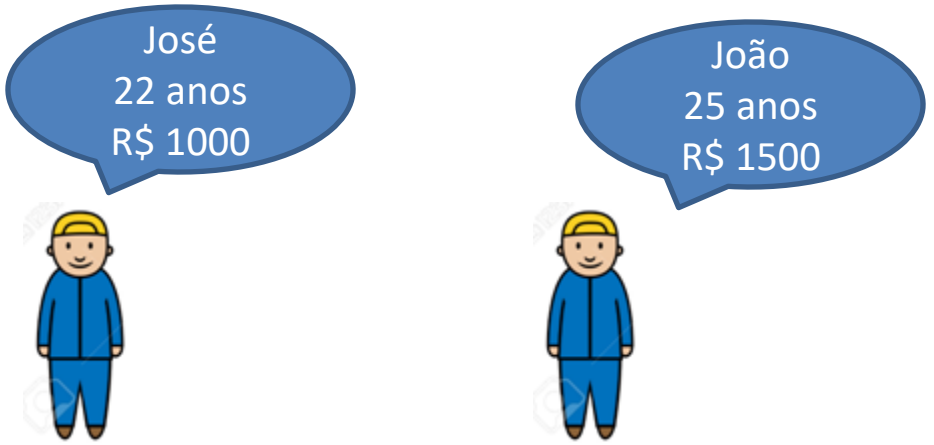
Structs em C

- **Struct é um conjunto, ou bloco, de variáveis**. A sintaxe é a seguinte:

```
struct Nome_de_sua_struct {
    tipos nome_dos_tipos;
};
```

Vamos declarar, como exemplo, uma struct para representar os dados dos funcionários:

```
struct Funcionario {
    int idade;
    char nome;
    float salario;
};
```



Structs em C

- Uma struct pode ser vista como **um novo tipo de variável**, assim é possível criar e declarar mais variáveis do tipo “struct Funcionario”.
- A sintaxe para declarar outras structs do tipo “struct Funcionario” é:

```
struct Funcionario {
   int idade;
   char nome;
   float salario;
};

struct Funcionario empregado;
struct Funcionario chefe;
struct Funcionario secretaria;
```

Structs em C

```
struct Funcionario {
  int idade;
  char nome;
  float salario;
};

struct Funcionario empregado[6];
struct Funcionario chefe[1];
struct Funcionario secretaria[2];
```

chefe

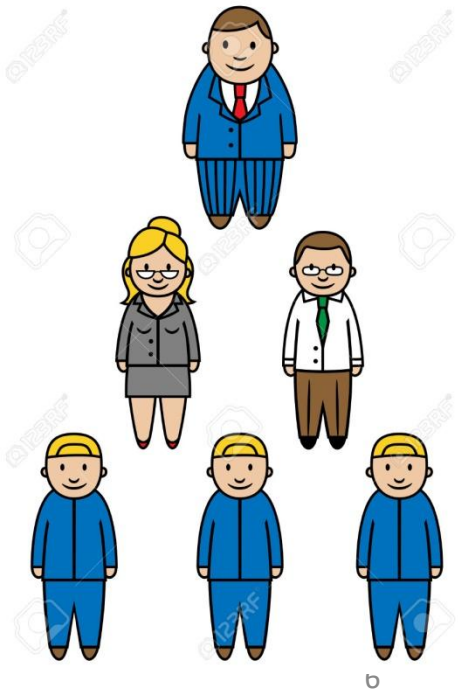
idade nome salario

secretaria

idade nome salario	idade nome salario

empregado

idade nome salario	idade nome salario	idade nome salario	...		



Structs em C

- Para acessar o elemento da struct usamos a sintaxe:

- **Chefe[0].idade** -> é um inteiro como outro qualquer.
- **Empregado[0].nome** -> é uma string como outra qualquer.
- **Secretaria[0].salario** -> é um float como outro qualquer.

- Apenas isso, basta botar um ponto após o nome que você escolheu para a struct.

Exemplo 1 struct em C

1. Defina uma struct para tratar de **alunos**. Dentro dessa struct, crie uma variável para armazenar o nome do aluno, e outras para armazenar as notas de Matemática, Física e a média dessas duas notas.

2. Preencha os nomes e notas dos alunos, calculando automaticamente a média deles.

3. Depois exiba tudo isso.

Declarando a struct alunos

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

Duas maneiras de declarar struct:

Sem usar typedef

```
struct Alunos {
    char nome[30];
    float matematica, fisica, media;
};

struct Alunos alunos[5];
```

Usando typedef

```
typedef struct {
    char nome[30];
    float matematica, fisica, media;
} Alunos;

Alunos alunos[5];
```

Cria uma estrutura com **nome** alunos do **tipo** Alunos, com 5 **posições de memória**.

Inserindo dados na struct alunos

```
for( int count = 0 ; count < 5 ; count++)  {

    getchar( );
     printf("Nome do aluno %d: ", count+1);
     gets(alunos[count].nome);

     printf("Nota de matematica: ");
     scanf("%f", &alunos[count].matematica);

     printf("Nota de fisica: ");
     scanf("%f", &alunos[count].fisica);

     alunos[count].media = (alunos[count].matematica + alunos[count].fisica)/2;
  }
```

Usa-se o operador . para acessar os valores da struct!

Imprimindo dados da struct alunos

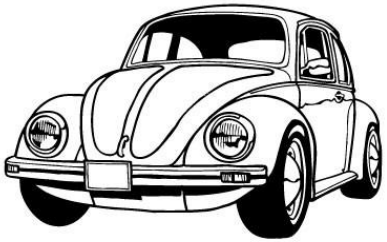
```
printf("Exibindo nomes e medias:\n");

for(count = 0 ; count < 5 ; count++)    {
        printf("Aluno %d\n", count+1);
        printf("Nome: %s\n",alunos[count].nome);
        printf("Media: %.2f\n", alunos[count].media);
}
```

Exemplo 2 struct em C

Defina uma struct de um carro. Crie um modelo de carro, preencha seus dados e exiba eles através de uma função que recebe esse tipo de struct.

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

typedef struct {
    char modelo[30];
    float potenciaMotor;
    int anoFabricacao, numPortas;
 } CARRO;
```

Pode declarar a struct dentro do main ou abaixo dos #include bibliotecas.



Passando struct com um elemento para função

```
void Exibe(CARRO car) {
    printf("Modelo: %s\n", car.modelo);
    printf("Motor: %.1f\n", car.potenciaMotor);
    printf("Ano: %d\n", car.anoFabricacao);
    printf("%d portas\n", car.numPortas);
}

int main(void) {
    CARRO fusca = {"Fusca", 1.5, 74, 2};
    Exibe(fusca);
    return 0;
}
```

Passando struct com vários elementos (vetor) para função

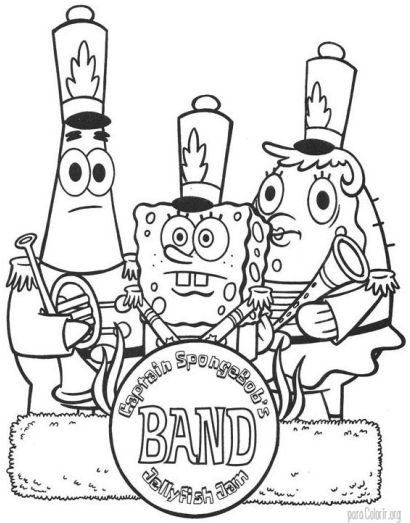
```
int main(void) {
      CARRO carro[5];
      Preenche(carro);
      return 0;
}

void Preenche(CARRO car[]) {
      for(i = 0; i < 5; i++) {
        printf("Modelo do carro: ");
        gets( car[i].modelo );
        printf("Motor: ");
        scanf("%f", &car[i].potenciaMotor);
        printf("Ano: ");
        scanf("%d", &car[i].anoFabricacao);
        printf("Numero de portas: ");
        scanf("%d", &car[i].numPortas);
      }
}
```

Exercício

1. Defina uma estrutura que irá representar **bandas de música**. Essa estrutura deve ter o nome da banda, que tipo de música ela toca, o número de integrantes e em que posição do ranking essa banda está dentre as suas 5 bandas favoritas.

   Crie um looping para preencher 5 estruturas de bandas.

   Após criar e preencher, exiba todas as informações das bandas/estruturas.




PROGRAMADOR: FIZ ALGO SIMPLES, SUPER INTUITIVO
USUÁRIOS
DownsHacker