Tipos de dados definidos pelo programador

Bacharelado em Ciência da Computação
Disciplina: Algoritmos e programação
Prof. Laércio Ives Santos
laercio.ives@gmail.com

Tipos básicos x definidos

- Tipos de dados primitivos: char, int, float, double e void;
- Tipos compostos homogênios: Arrays
- Tipos definidos pelo programador: A linguagem C permite criar novos tipos (ditos heterogênios) para tanto, um dos seguintes comandos deve ser utilizado: struct, union, enum ou typedef

Estruturas: struct

- A ideia básica da estrutura é criar apenas um tipo de dado que contenha vários membros, que são nada mais que variáveis (int, char, float, array).
- Pode ser vista como um conjunto de variáveis sob um mesmo nome

Estruturas: struct

- Declarando um struct

```
 9  #include <stdio.h>
10  struct aluno{
11          char nome[50];
12          int idade;
13          float nota;
14  };
15
16  int main()
17  {
18      return 0;
19  }
```

Criamos um novo tipo de dados chamado “aluno”;

O novo tipo é composto por 3 membros;

Por que o escopo global?

Estruturas: struct

- A ideia básica da estrutura é criar apenas um tipo de dado que contenha vários membros, que são nada mais que variáveis (int, char, float, array).
- Pode ser vista como um conjunto de variáveis sob um mesmo nome
- A principal vantagem é que agora podemos agrupar vários tipos de dados em uma única variável.

Estruturas: struct

- Declarando uma variável do tipo aluno

```
 9  #include <stdio.h>
10  struct aluno{
11          char nome[50];
12          int idade;
13          float nota;
14  };
15
16  int main()
17  {
18      struct aluno a;
19      return 0;
20  }
```

A variável “a” é do tipo aluno

Então, “a” é composta por 3 partes (nome, idade e nota)

Estruturas: struct

- Declarando muitas variáveis do tipo aluno

```
 9  #include <stdio.h>
10  struct aluno{
11          char nome[50];
12          int idade;
13          float nota;
14  };
15
16  int main()
17  {
18      struct aluno a, b, novoAluno, aluno1;
19      return 0;
20  }
```

Estruturas: struct

- Acessando partes (membros) da estrutura... (<estrutura>.<membro>)

```
17  int main()
18  {
19      struct aluno a;
20      strcpy(a.nome,"Ana Luiza");
21      a.nota = 8.5;
22      a.idade = 19;
23
24      return 0;
25  }
```

Estruturas: struct

- Acessando partes (membros) da estrutura...

```
18  int main()
19  {
20      struct aluno a;
21      //scanf("%s", a.nome);
22      gets(a.nome);
23      scanf("%f", &a.nota);
24      scanf("%d", &a.idade);
25
26      return 0;
27  }
```

Estruturas: struct

- Iniciando valores para a estrutura...

```
12  struct aluno{
13          char nome[50];
14          int idade;
15          float nota;
16  };
17
18  int main()
19  {
20      struct aluno a={"Laercio Ives Santos", 36, 9.9};
21      printf("%s %f %d", a.nome, a.nota, a.idade);
22      return 0;
23  }
```

Estruturas: struct

- Arrays de estruturas (nome_array[posicao].membro_struct)

```
18  int main()
19  {
20      struct aluno a[5];
21      strcpy(a[0].nome, "Ivanderley souza costa");
22      a[0].idade = 19;
23      a[0].nota = 10;
24
25      return 0;
26  }
```

Declara o array

Acessa o primeiro elemento do array

Estruturas: struct

- Arrays de estruturas (nome_array[posicao].membro_struct)

```
18  int main()
19  {
20      struct aluno a[3];
21      for (int i=0; i<3; i++){
22        gets(a[i].nome);
23        scanf("%d%*c", &a[i].idade);
24        scanf("%f%*c", &a[i].nota);
25      }
26      return 0;
27  }
```

Estruturas: struct

- Atribuição de estruturas

```
18  int main()
19  {
20      struct aluno a[3], b;
21      for (int i=0; i<3; i++){
22          gets(a[i].nome);
23          scanf("%d%*c", &a[i].idade);
24          scanf("%f%*c", &a[i].nota);
25      }
26      b = a[0];
27      return 0;
```

Só é permitido entre estruturas do mesmo tipo;

Estudo Dirigido...

1 - Escreva um programa para fazer a criação dos novos tipos de dados conforme solicitado abaixo:

Horário: composto de hora, minutos e segundos.

Data: composto de dia, mês e ano.

Compromisso: composto de uma data, horário e texto que descreve o compromisso.

2 – declare um vetor de 5 posições para armazenar a estrutura compromisso do exercício 1, leia e mostre os dados de compromisso na tela;

3 - Crie um programa que declare e leia a estrutura aluno[5] dos slides anteriores, em seguida percorra o vetor e verifique quais alunos estão aprovados ou reprovado de acordo com a nota (“aprovado” nota >= 6 e reprovado nota < 6). Mostre essa informação para o usuário.

Estudo Dirigido...

4 – Utilize o vetor declarado do exercício 3 e crie uma função para ordenar de forma crescente o vetor de acordo com a nota do aluno. A função deve ter o seguinte protótipo:

void ordenarAlunos(struct aluno *vet, int n);

Onde vet é o vetor de alunos e n é o tamanho do vetor.

Ps: A passagem de arrays de structs para funções segue as mesmas regras já vistas em sala de aula.

Estudo Dirigido...

5 – Crie um programa em C, com 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. Na função int main(), declare um vetor de 5 posições para armazenar os dados dos alunos.

(a) Crie uma função que permita ao usuário entrar com os dados de 5 alunos.

(b) Crie uma função que permita: encontrar o nome do aluno com maior nota da primeira prova, Encontrar o aluno com maior média geral, encontrar o aluno com menor média geral. Esses dados devem ser retornados para a função chamadora.

(c) crie uma função que mostre o status de cada aluno, informando se ele foi aprovado ou reprovado, considerando a média das 3 notas igual ou superior a 6 para aprovação.

1)

```
18  struct data{
19      int dia, mes, ano;
20  };
21  struct horario{
22    int hora, minuto, segundo;
23  };
24  struct compromisso{
25    struct data dataC;
26    struct horario horaC;
27    char texto[100];
28  };
```

2)

```
31  int main()
32  {
33      struct compromisso c[5];
34      for (int i=0; i<5; i++){
35        printf("Descreva o comprimisso: ");
36        gets(c[i].texto);
37        printf("Entre com a Data do comprimisso: ");
38        scanf("%d%*c", &c[i].dataC.dia);
39        scanf("%d%*c", &c[i].dataC.mes);
40        scanf("%d%*c", &c[i].dataC.ano);
41        printf("Entre com a Hora do comprimisso: ");
42        scanf("%d%*c", &c[i].horaC.hora);
43        scanf("%d%*c", &c[i].horaC.minuto);
44        scanf("%d%*c", &c[i].horaC.segundo);
45       }
46      return 0;
```

3)

Dúvidas???