Aula 3

Linguagem de Programação

Prof. Sandro de Araujo

- **Registros – STRUCT, UNION, ENUN e TYPEDEF**

Conversa Inicial

- **O objetivo desta aula é conhecer os principais conceitos de struct, union, enun e typedef na linguagem de programação C**

- **Vamos aprender como representá-los facilmente em diversos algoritmos para resolver problemas computacionais**

- **A aula apresenta a seguinte estrutura de conteúdo:**
  - **STRUCT**
  - **UNION**
  - **ENUN**
  - **TYPEDEF**
  - **TYPEDEF e STRUCT**

STRUCT

Na linguagem de programação C podemos declarar tipos de variáveis como:

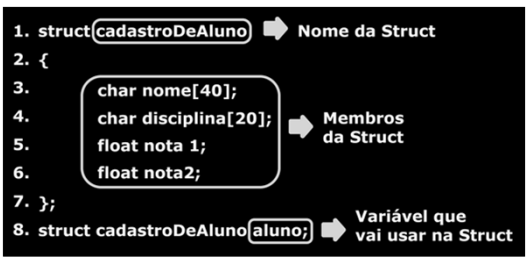
- Tipos básicos: char, int, float, double;
  - Exemplo: int x; float y;
- Tipos compostos homogêneos: array
  - Exemplo: int x[5]; char nome[25];

STRUCT

- Pode ser vista como um conjunto de variáveis referenciadas pelo mesmo nome, sendo que cada uma delas pode ter o mesmo tipo de dado ou vários tipos

STRUCT

```
1. struct <nome_da_struct>
2. {
3.      <tipo 1> e <variável 1>;
4.      <tipo 2> e <variável 2>;
5.      <tipo 3> e <variável 3>;
6.      ...
7.      <tipo n> e <variável n>;
8.
9. };  struct < nome_da_struct > <nome_variavel>;
```



STRUCT

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

int man(){

printf("\n CADATRO DE CLIENTE\n\n");

struct ficha_do_cliente    /*Criando a struct */
{
      char nome[50];
      char rua[50];
      int telefone[11];
      char email[40];
 }; struct ficha_do_cliente cliente;
```

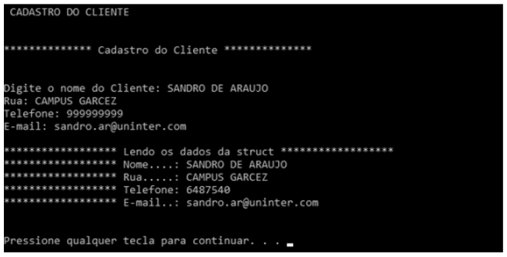
STRUCT

```
printf("Digite o nome do Cliente: ");
flush(stdin);
fgets(cliente.nome, 50, stdin);

printf("Telefone: ");
scanf("%d",&cliente.telefone);

printf("E-mail: ");
fflush(stdin);
fgets(cliente.email, 30, stdin);
```

STRUCT

```
    printf("\n*** Imprimindo os dados da struct ***\n");
    printf("*** Nome....: %s", cliente.nome);
    printf("*** Rua.....: %s", cliente.rua2);
    printf("\n*** E-mail..: %s", cliente.email);
    printf("\n\n");

    getch();
    return 0;
}
```

STRUCT



UNION

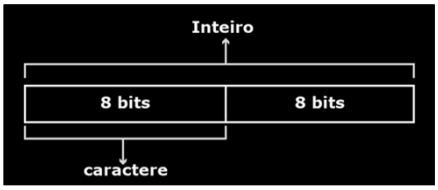
UNION

- **Com a union, pode-se criar variáveis capazes de suportar dados diferentes, alocados no mesmo espaço de memória, em momentos diferentes. Isto é, a union permite que um conjunto de variáveis compartilhem o mesmo espaço na memória**

UNION

1. ***union <nome_da_union>***
2. ***{***
3. ***<tipo 1> e <variável 1>;***
4. ***<tipo 2> e <variável 2>;***
5. ***<tipo 3> e <variável 3>;***
6. ***…***
7. ***<tipo n> e <variável n>;***
8. ***};***

UNION



UNION

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

union ex_union{

        int inteiro_1;
        char caractere_1;
        float decimal_1;

 };
```

UNION

```
struct ex_struct{

        int inteiro_2;
        char caractere_2;
        float decimal_2;

 };
```

UNION

```
int main()
{
printf("Tamanho da Union: %d\n", sizeof(union
ex_union));
printf("Tamanho da Struct: %d\n", sizeof(struct
ex_struct));

return 0;
}
```

UNION

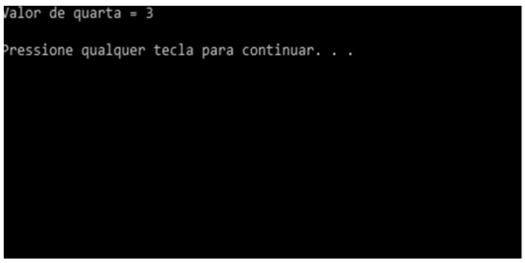
```
Tamanho da Union: 4
Tamanho da Struct: 12

Process returned 0 (0x0)   execution time : 0.115 s
Press any key to continue.
```

ENUM

ENUM

- **Enum ou enumeração é um tipo de dado definido pelo usuário, com o uso de uma lista de identificadores. Os identificadores podem ser vistos como uma lista de constantes, em que cada constante tem um nome significativo**

ENUM



ENUM

```
enum semana{

domingo = 1, segunda, terca, quarta, quinta =
8, sexta, sábado

};
```

TYPEDEF

TYPEDEF

- **O comando typedef é usado para criar "sinônimo" ou um "aliás" para tipos de dados existentes. Na prática, podemos dizer que estamos renomeando um tipo de dados**

TYPEDEF

- **É importante ressaltar que o comando typedef não cria um novo tipo. Ele apenas permite que um tipo existente seja denominado de uma forma diferente, de acordo com a especificação desejada pelo programador**

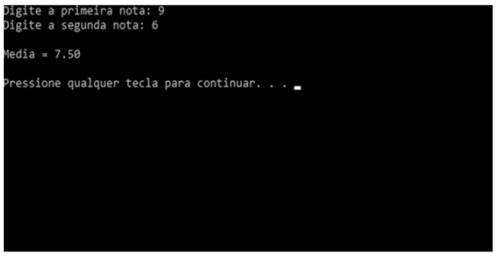
TYPEDEF

```
typedef float prova;     //redefinição do tipo float para o tipo prova

prova nota1, nota2, media;     // variáveis usando o tipo prova
```

TYPEDEF

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

int main (void)
{
      typedef float prova;
      prova nota1, nota2, media;
```

TYPEDEF



TYPEDEF E STRUCT

TYPEDEF e STRUCT

- **É muito frequente o uso de typedef para criar apelidos a fim de tornar os nomes mais curtos, desta forma podemos representar uma estrutura usando apenas seu sinônimo**

TYPEDEF e STRUCT

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

typedef float prova;     //redefinindo float
typedef int RU;          //redefinindo int
```

TYPEDEF e STRUCT

```
struct notasAluno
{
    RU matricula;
    prova nota1;
    prova nota2;
}; typedef struct notasAluno n_aluno ;
```

TYPEDEF e STRUCT

```
int main ()
{
    n_aluno aluno; // Não é mais necessário escrever "struct n_aluno"
    prova media = 0;

    printf ("Digite a matricula do aluno: ");
    scanf ("%d", &aluno.matricula);

    printf ("Digite a primeira nota: ");
    scanf ("%f", &aluno.nota1);
```

TYPEDEF e STRUCT