Curso de programação em C - FK

1 – Estrutura, variáveis e operadores.

2 – Estrutura condicional (If, If-Else e Switch-Case).

3 – Estrutura de repetição (For, While e Do-While).

4 – Vetores, Strings, Matrizes e Structs.

5 – Funções e Arquivos.

Programação em C

Aula 6 – Vetores, Strings, Matrizes e Structs.

- Vetor unidimensional - array
- Vetor de caracteres – funções da string
- Vetores multidimensionais - matriz
- Estruturas - registros

Vetor (array) – matriz unidimensional

- Um vetor é uma sequência de variáveis de mesmo tipo de dado e referenciadas por um nome único.
- A posição de um elemento em um vetor é identificada por um índice, que é um número inteiro.

- O índice do primeiro elemento de um vetor é sempre 0, e os índices variam entre 0 e n-1, onde n é o número de elementos do vetor.

Exemplos de vetores

- Sintaxe:

<tipo_da_variável> <nome_da_variável>[<tamanho>];

- Exemplos:

int num1 [10];

float media [4];

char nomeCliente [30];

int vetorA [10];

Exemplo 1

1. Faça um programa que leia 5 valores inteiros, armazeno-os em um vetor, calcule e apresente a soma destes valores.

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

int main()
{
    int num [5];
    int i, soma = 0;

    for(i = 0; i < 5; i++)
    {
        printf("Digite o numero da posicao %d: ",i);
        scanf("%d",&num[i]);
        soma = soma + num[i];
    }
    printf("Soma = %d",soma);

    return 0;
}
```

```
Digite o numero da posicao 0: 10
Digite o numero da posicao 1: 0
Digite o numero da posicao 2: 4
Digite o numero da posicao 3: 5
Digite o numero da posicao 4: 21
Soma = 40
--------------------------------
Process exited after 46.89 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

String

- A string consiste em uma cadeia de caracteres, terminando com o caractere especial ‘\0’, que indica o fim da string.

- Como ao final da string é armazenado o ‘\0’, temos que declarar a string sempre com uma posição a mais do que o número de caracteres que desejamos armazenar.

- As constantes strings aparecem entre aspas duplas e não é necessário acrescentar o ‘\0’, isso é realizado em tempo de execução.

Sintaxe e exemplos

- Sintaxe:

```
char <nome_string>[<tamanho>];
char nomeCliente[40];
char cidade [50];
```

Funções para manipulação de strings

Funções	Operadores
gets(x)	Lê caracteres até encontrar o de nova linha ('\n'), que é gerado quando se pressiona a tecla [enter].
puts(x)	Imprime uma string de cada vez.
strlen(x)	Retorna o tamanho da string armazenada, sem '\0'.
strcat(x,y)	Concatena a string x à string y, sem alterar y.
strcmp(x,y)	Retorna a diferença (baseada na tabela ASCII) entre os dois primeiros caracteres diferentes, ou zero para igualdade entre as strings x e y.
strcpy(x,y)	Copia uma string em outra, ou seja, copia y em x.
strncpy(x,y,n)	Armazena em x os n primeiros caracteres de y.
strlwr(x)	Retorna o minúsculo da string x.
strupr(x)	Retorna o maiúsculo da string x.
strstr(x,y)	Verifica se y é subcadeia da string x.
atoi(x)	Converte x para o tipo int.
atol(x)	Converte x para o tipo long.
atof(x)	Converte x para o tipo float.

Uso de string

- Necessário utilizar a biblioteca string.h

- gets ou scanf(“%s”,&var);

- O Scanf não lê espaços em branco e vai parar de ler a linha no primeiro espaço em branco que ele encontrar e o gets vai ler a linha até o final incluindo os espaços em branco.

- fflush(stdin) - Função reponsável por "limpar o buffer do teclado".

- Em alguns dispositivos computacionais, é possível que operações de leitura de dados deem problema em função do buffer do teclado armazenar lixos de memória.

Exemplo 2

Faça um programa que lê uma string e retorne o tamanho do texto e converta todos as letras em caracteres maiúsculos.

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

int main()
{
    char cidade [20];
    int tamanho;

    printf("Digite sua cidade: ");
    gets(cidade);

    fflush(stdin);

    tamanho = strlen(cidade);
    strupr(cidade);

    printf("\nA cidade digitada foi %s",cidade);
    printf("\nForam usados %d caracteres",tamanho);


    return 0;
}
```

```
Digite sua cidade: Curitiba

A cidade digitada foi CURITIBA
Foram usados 8 caracteres
--------------------------------
Process exited after 4.489 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

```
Digite sua cidade: rio de janeiro

A cidade digitada foi RIO DE JANEIRO
Foram usados 14 caracteres
--------------------------------
Process exited after 10.48 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

Exemplo 3

- Faça um programa que receba uma palavra em uma string e crie uma outra string para receber a palavra de trás para frente e a imprima. Use um algoritmo para ler a string com scanf() e depois outro com gets().

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

int main()
{
    char palavra [15];
    char contrario [15];
    int tamanho, i, j = 0;

    printf("Digite uma palavra: ");
    scanf("%s",&palavra);

    fflush(stdin);

    tamanho = strlen(palavra);    //Exemplo se a palavra for futebol, são 7 letras
    i = tamanho - 1;              //o indice da string inicia em 0 e vai até 6.

    for (i; i >= 0; i--)
    {
        contrario [j] = palavra[i];
        j++;
    }
    contrario [tamanho] = '\0';  //é necessário após incluir as 7 letras que na posição  7
                                 //seja adicionado o '\0' para informar a string que o texto
    puts(contrario);             //finalizou. Caso contrário pode imprimir lixo de memória.

    return 0;
}
```

```
Digite uma palavra: futebol
lobetuf

--------------------------------
Process exited after 3.311 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

```
Digite uma palavra ou frase: ano 2024
ona

--------------------------------
Process exited after 11.45 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

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

int main()
{
    char palavra [15];
    char contrario [15];
    int tamanho, i, j = 0;

    // Teste no programa usando gets em vez de scanf
    printf("Digite a mesma palavra ou frase: ");
    gets(palavra);

    fflush(stdin);

    tamanho = strlen(palavra);
    i = tamanho - 1;

    for (i; i >= 0; i--)
    {
        contrario [j] = palavra[i];
        j++;
    }
    contrario [tamanho] = '\0';

    puts(contrario);

    return 0;
}
```

```
Digite a mesma palavra ou frase: futebol
lobetuf

--------------------------------
Process exited after 3.846 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

```
Digite a mesma palavra ou frase: ano 2024
4202 ona

--------------------------------
Process exited after 3.981 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

Matrizes

- Matriz é a uma estrutura de dados do tipo vetor com duas ou mais dimensões.

- Os itens de uma matriz tem que ser todos do mesmo tipo de dado.

- Na prática, as matrizes formam tabelas na memória.

Sintaxe

A sintaxe para a declaração de uma matriz é dada por:

<tipo> <nome> [<tamanho1>][<tamanho2>]...[<tamanhoN>];

- Como exemplo de declaração de matrizes binárias, temos:

int matriz [2] [3];

- onde nessa matriz constam 2 linhas e 3 colunas.

Exemplo 4

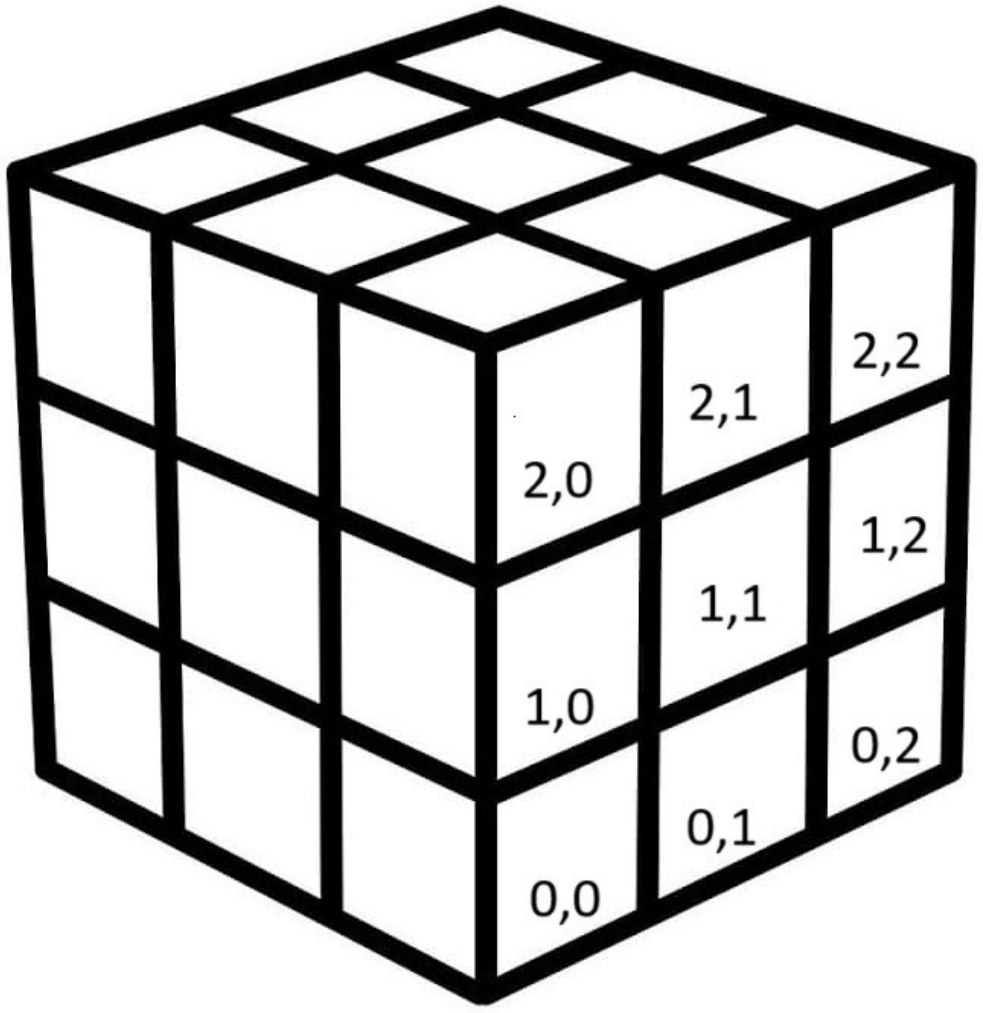
Digite uma matriz que receba o endereço de um usuário. Com os dados de rua, bairro e cidade. E depois imprima a matriz;

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


int main()
{
    char endereco [3][20];
    printf("Digite a rua: ");
    gets(endereco[0]);

    printf("Digite a bairro: ");
    gets(endereco[1]);

    printf("Digite a cidade: ");
    gets(endereco[2]);

    puts(endereco[0]);
    puts(endereco[1]);
    puts(endereco[2]);
    return 0;
}
```

```
Digite a rua: Dom Pedro I
Digite a bairro: Centro
Digite a cidade: Campo Grande
Dom Pedro I
Centro
Campo Grande

--------------------------------
Process exited after 17.89 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

Exemplo 5

Leia uma matriz de 3 x 3 elementos. Calcule a soma dos elementos que estão na diagonal secundaria.


2,2
2,1
2,0
1,2
1,1
1,0
0,2
0,1
0,0

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

int main()
{
    int matrizA[3][3];
    int i, j, soma;

    for (i =0;i < 3; i++)
    {
        for(j = 0;j < 3;j++)
        {
            printf("Digito numero da posica %d %d: ",i,j);
            scanf("%d",&matrizA[i][j]);
        }
    }
    printf("\n\n");
    //imprimir a tabela
    for (i =0;i < 3; i++)
    {
        for(j = 0;j < 3;j++)
        {
            printf("%d ",matrizA[i][j]);
        }
        printf("\n\n");
    }

    soma = matrizA[0][0] + matrizA[1][1] + matrizA[2][2];
    printf("\nA soma da diagonal principal da matriz A e: %d",soma);

    return 0;
}
```

```
Digito numero da posica 0 0: 1
Digito numero da posica 0 1: 2
Digito numero da posica 0 2: 3
Digito numero da posica 1 0: 4
Digito numero da posica 1 1: 5
Digito numero da posica 1 2: 6
Digito numero da posica 2 0: 7
Digito numero da posica 2 1: 8
Digito numero da posica 2 2: 9


1 2 3

4 5 6

7 8 9


A soma da diagonal principal da matriz A e: 15
--------------------------------
Process exited after 7.556 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

Structs

- As estruturas (structs), também chamadas registros (records) são variáveis compostas pelo agrupamento de diversas variáveis e vistas/tratadas pelo programa como uma única variável.

- A estrutura permite agregar diversas informações, que podem ser de diferentes tipos.

-As variáveis que compõem uma estrutura podem ser simples (int, float, pointer) ou compostas (arrays e outras estruturas).

Sintaxe - Structs

```
struct <nome_da_estrutura>
{
   <tipo_de_dado1> <campo1>;
   <tipo_de_dado2> <campo2>;
   ...
   <tipo_de_dadoN> <campoN>;
};
```

Criando uma struct

Ex.: A declaração da estrutura para a ficha de um produto seria antes da função main algo como:

```
struct produto
{
   int codigo;
   char descricao[50];
   float preco;
   int saldo;
};
```

```
int main()
{
   struct produto ficha; //Esta declaração nos indica que temos uma variável denominada ficha
   printf(“Digite o código do produto:”);
   scanf(“%d”, &ficha.codigo);
   printf(“Digite a descrição do produto:”);
   scanf(“%s”, ficha.descricao);
   printf(“Digite o preço do produto:”);
   scanf(“%f”, &ficha.preco);
   printf(“Digite o saldo do produto:”);
   scanf(“%d”, &ficha.saldo);
   return 0;
}
```

Exemplo 6

- Implemente um programa que leia o nome, a idade, altura e o endereço de um cliente e armazene os dados em uma estrutura. E depois imprima os dados.

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

struct cliente
{
    char nome [50];
    int idade;
    float altura;
    char endereco [50];
};

int main()
{
    struct cliente cadastro;

    printf("Digite o nome do cliente: ");
    gets(cadastro.nome);
    printf("Digite a idade do cliente: ");
    scanf("%d", &cadastro.idade);
    printf("Digite a altura do cliente: ");
    scanf("%f", &cadastro.altura);
    fflush(stdin);
    printf("Digite o endereco do cliente: ");
    gets(cadastro.endereco);
    fflush(stdin);

    printf("\nCliente: %s. Idade: %d. Altura: %.2f.",cadastro.nome,cadastro.idade, cadastro.altura);
    printf("\nEndereco: %s",cadastro.endereco);

    return 0;
}
```

```
Digite o nome do cliente: Roberto Carlos
Digite a idade do cliente: 65
Digite a altura do cliente: 1.71
Digite o endereco do cliente: Rua Nova Lima, 3400

Cliente: Roberto Carlos. Idade: 65. Altura: 1.71.
Endereco: Rua Nova Lima, 3400
--------------------------------
Process exited after 60.56 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

Fim da aula 6

- Autor: Fernando Eduardo.

- Dúvidas encaminhe e-mail para:
dunano@outlook.com