

Exercício:

Escrever um programa que executa o seguinte algoritmo:

1. preenche a matriz mat (LxC) por leitura (valores fornecidos por coluna à coluna);
2. imprime os valores da matriz no formato linha à linha (alinhado!);
3. calcula a soma de cada coluna da matriz e armazena esta soma em um vetor V;
4. imprime o vetor;
5. procura o maior elemento da matriz e sua posição;
6. informa o valor e esta posição (não precisa prever duplicidade).

Os passos 1 a 5 deverão ser executados por funções, chamadas a partir do programa principal.

O passo 6 deverá ser executado no programa principal.

Testar o programa com L = 3 e C = 4.

Não usar variáveis globais

Slide 1

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Tela da execução:

```
Forneca valores da coluna 1:
Linha 1: 1
Linha 2: 3
Linha 3: 5
Forneca valores da coluna 2:
Linha 1: 4
Linha 2: 18
Linha 3: 11
Forneca valores da coluna 3:
Linha 1: 2
Linha 2: 6
Linha 3: 8
Forneca valores da coluna 4:
Linha 1: 12
Linha 2: 13
Linha 3: 14

Matriz lida:
  1   4   2  12
  3  18   6  13
  5  11   8  14

Soma das colunas da matriz:
  9  33  16  39

Maior valor:18
Posicao: [2,2]

Pressione qualquer tecla para continuar.
```

Slide 2

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

```
// Trabalho da aula 19:
#include <stdio.h>
#include <stdlib.h>
#define L 3
#define C 4
// protótipo das funções a serem desenvolvidas:
void le_matriz(int [ ][C]);
void imprime_matriz(char [ ][C]); // texto a ser impresso + matriz
void soma_col_mat(int [L][C], int [C]);
void imprime_vetor(char [ ][C]); // texto a ser impr.vetor,num elem do vet
void maior_elemento(int mat[L][C], int *pos_lin_maior, int *pos_col_maior)
// programa principal:
int main()
{
    int mat[L][C], vet[C]; // matriz e vetor somatório das colunas
    int lin_maior, col_maior; // posição do maior valor
    system("color f1");
    le_matriz(mat);
    imprime_matriz("Matriz lida",mat);
    soma_col_mat(mat, vet);
    imprime_vetor("Soma das colunas da matriz",vet,C);
    maior_elemento(mat, &lin_maior, &col_maior); //obtem e imprime a seguir!
    printf("\nMaior valor:%d \nPosicao: [%d,%d]\n", mat[lin_maior][col_maior],
        lin_maior + 1, col_maior + 1);
    printf("\n");
    system("pause");
    return 0;
}
```

```
// Leitura da matriz:
void le_matriz(int matriz[ ][C]) // precisa incluir outras dimensões
{
    int il,ic; // índices para linha e coluna
    // Leitura da matriz, coluna à coluna:
    for (ic=0; ic < C; ic++)
    {
        printf("Forneca valores da coluna %d:\n",ic + 1);
        for (il=0; il < L; il++)
        {
            printf("  Linha %d: ",il + 1);
            scanf("%d",&matriz[il][ic]);
        }
    }
}
```

```
Forneca valores da coluna 1:
Linha 1: 1
Linha 2: 3
Linha 3: 5
Forneca valores da coluna 2:
Linha 1: 4
Linha 2: 18
Linha 3: 11
Forneca valores da coluna 3:
Linha 1: 2
Linha 2: 6
Linha 3: 8
Forneca valores da coluna 4:
Linha 1: 12
Linha 2: 13
Linha 3: 14
```

Slide 4

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

```
// Impressão da matriz:
void imprime_matriz(char nome_matriz[ ],int matriz[ ][C])
{
    int il,ic; // índices de linha e coluna
    // Impressão de cabeçalho, informado em nome_matriz:
    printf("\n%s:\n",nome_matriz);
    // Impressão da matriz, linha à linha:
    for (il=0; il < L; il++)
    {
        for (ic=0; ic < C; ic++)
            printf("%5d",matriz[il][ic]);
        printf("\n");
    }
}
```

```
Matriz lida:
  1   4   2  12
  3  18   6  13
  5  11   8  14
```

Slide 5

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

```
// Cálculo do somatório das colunas:
void soma_col_mat(int mat[L][C], int vet_soma[ ])
{
    int lin,col; // índices
    for(col=0; col < C; col++)
    {
        vet_soma[col]=0; // zera somatório da coluna no vetor
        for(lin=0; lin < L; lin++) // percorre coluna na matriz
            vet_soma[col] = vet_soma[col] + mat[lin][col];
    }
}
```

Slide 6

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

```
// Impressão do vetor somatório:
void imprime_vetor(char nome_vetor[ ],int vetor[ ],int num_elem)
{
    int i;
    // Impressão de cabeçalho, informado em nome_matriz:
    printf("\n%s:\n",nome_vetor);
    // Impressão do vetor em 1 única linha:
    for (i=0; i < num_elem; i++)
        printf("%5d",vetor[i]);
    printf("\n");
}
```

Soma das colunas da matriz:
 9 33 16 39

Slide 7

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

```
// Identificação da linha e coluna (ponteiros) do maior valor da matriz:
// (impressão feita no programa principal, a partir dos valores retornados!
void maior_elemento(int mat[L][C], int *pos_lin_maior, int *pos_col_maior)
{
    int lin,col;
    *pos_lin_maior = 0; // assume mat[0][0] como maior
    *pos_col_maior = 0; // e utiliza posição para comparação
    for(lin=0; lin < L; lin++)
        for(col=0; col < C; col++)
            if (mat[*pos_lin_maior][*pos_col_maior] < mat[lin][col])
                {
                    *pos_lin_maior = lin;
                    *pos_col_maior = col;
                }
}
```

Maior valor:18
Posicao: [2,2]

Pressione qualquer tecla para continuar. . .

Slide 8

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Exercício: Faça programa com procedimento que retorne os n primeiros termos da série de Fibonacci:

//calcular e apresentar os n primeiros termos da serie de fibonacci

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

```
void termos_fibonacci (int n)
```

```
{
    // C:\Cora\Disciplinas\INFO1202 C\Pro...
    Informe o nro de termos desejados da serie de Fibonacci:8
    Pressione qualquer tecla para continuar. . .
}
```

```
int main ( )
{
    int nro_termos;
    system("color f1");
    printf("\nInforme o nro de termos desejados da serie de Fibonacci:");
    scanf("%d", &nro_termos);
    termos_fibonacci(nro_termos);
    printf("\n");
    system("pause");
    return 0;
}
```

Slide 9

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

```
// Versão 1:
void termos_fibonacci (int n)
{
    int t,t_1=1, t_2=1; // declara termos, inicializando os 2 primeiros
    int cont; // variável de controle do for
    for (cont=1; cont <= n; cont++)
        if (cont == 1 || cont == 2) // 2 primeiros termos iniciam iguais
            printf("%d ",t_1); // ou t_2, tanto faz
        else
        {
            t=t_1+t_2; // calcula termo atual;
            printf("%d ",t); // imprime termo calculado e
            t_2=t_1; // atualiza conteúdos anteriores, já
            t_1=t; // preparando para o calculo do próximo termo
        }
}
int main ( )
{
    .....
}
```

Slide 10

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

```
// Versão 2:
/* Inicializa t_1 com 0 e t_2 com 1, para também obter os 2 1ºs termos a
partir da forma genérica t = t_1 + t_2;*/
void termos_fibonacci (int n)
{
    int t,t_1=0, t_2=1; // declara termos, inicializando os 2 primeiros
    int cont; // variável de controle do for
    for (cont=1; cont <= n; cont++)
        // 1ª vez: t=0+1;t_2=1;t_1=0; ---- t == 1
        // 2ª vez: t=1+0;t_2=0;t_1=1; ---- t == 1
        {
            t=t_1+t_2; // calcula termo atual;
            printf("%d ",t); // imprime termo calculado e
            t_2=t_1; // atualiza conteúdos anteriores, já
            t_1=t; // preparando para o calculo do próximo termo
        }
}
int main ( )
{
    .....
}
```

Slide 11

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Funções void / Funções com retorno

• Funções void

- Não retornam nenhum valor associado a execução da função;
- Podem alterar conteúdos de parâmetros, se passados por referência.

• Funções com retorno

- Devolvem um único valor associado a execução da função, usando o comando **return**.

Slide 12

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Declaração de funções com retorno

```
tipo <identificador> [( <lista de parâmetros formais> ) ]  
{  
    [ <declaração de variáveis locais>; ]  
    <comandos separados por ';' >  
    return;  
}
```

tipo: básico (char, int, float) ou ponteiro (próximo semestre!).

opcionais

```
float teste ( int a, int b)
```

// exemplo da sintaxe de função com retorno:

```
float x, res;  
scanf("%f", &x);  
res = a + b + x;  
return res;  
}
```

```
void teste (int a, int b, float *c)
```

// exemplo da sintaxe de função sem retorno:

```
float x;  
scanf("%f", &x);  
*c = a + b + x;  
}
```

Slide 13

Profa. Cora H.

O comando return

- A função termina sua execução quando seu último comando for executado ou quando o comando **return** for encontrado.
- O comando **return** provoca a **saída imediata** da função que o contém e retorna a execução ao código chamador.
- O comando **return** pode ser usado para **devolver apenas um** valor.

Slide 14

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Exemplo – solução indesejável

```
int caract_in_str (char c, char s[ ])
/* recebe uma string e um caractere, devolvendo a 1ª posição do
caractere c na string s ou -1, se o caractere não for encontrado */
{ // interrupção de for: contra princípios da programação estruturada!
    int i;
    for (i=0; i < strlen(s); i++) // Única solução ???
        if (tolower(s[i]) == tolower(c))
            return i; // encerra execução e retorna i
    return -1;
}
```

1º retorno
2º retorno

Apenas 1 valor é retornado!

Slide 15

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Exercício – solução de acordo com programação estruturada

```
int caract_in_str (char c, char s[ ])
/* recebe uma string e um caractere, devolvendo a 1ª
posição do caractere c na string s ou -1, se o caractere não
for encontrado */
int i=0;
/* laço condicional - while, enquanto não encontrou nem
percorreu toda a string: */
while ( (i < strlen(s)) && (tolower(s[i]) != tolower(c)))
    i++;
if (i == strlen(s)) // chegou ao caractere de final
    return -1; // não encontrou
else
    return i; // encontrou: retorna i
```

Mutualmente exclusivos!

Slide 16

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Chamada da função

< nome da função > (< lista de parâmetros reais >)

→ Utilizar diretamente o resultado da chamada da função

Ex:

```
float soma (float a, float b)
// soma os valores de A e B
{
    return a + b;
}

int main()
{
    float v1, v2, somadois;
    printf("valores a serem somados");
    scanf("%f%f", &v1, &v2);
    somadois = soma(v1, v2);
    printf("Soma= %f\n", somadois);
    printf("Outra vez: %f\n", soma(v1, v2));
    system("pause");
    return 0;
}
```

Declaração da função

Chamadas da função

Slide 17

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Exercício

- Faça um bloco **main()** que chame a função abaixo (solução estruturada):

```
int caract_in_str (char c, char s[ ])
/* recebe uma string e um caractere, devolvendo a 1ª
posição do caractere c na string s ou -1, se o caractere não
for encontrado */
int i=0;
/* laço condicional - while, enquanto não encontrou nem
percorreu toda a string: */
while ( (i < strlen(s)) && (tolower(s[i]) != tolower(c)))
    i++;
if (i == strlen(s))
    return -1; // não encontrou
else
    return i; // encontrou: retorna i
}
```

Slide 18

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Solução do Exercício

```
int main()
{
    char palavra[20], letra;
    int posicao;
    printf("digite a palavra: ");
    scanf("%s", palavra);
    printf("digite a letra: ");
    fflush(stdin);
    scanf("%c", &letra);
    posicao = caract_in_str(letra, palavra); // chama função
    if (posicao >= 0) // encontrou
        printf("%c e a %da letra da palavra.\n", letra, posicao+1);
    else
        printf("caractere %c nao encontrado.\n", letra);
    system("pause");
    return 0;
}
```

Slide 19

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Ex:

Função void para trocar o conteúdo de 2 variáveis, inteiras:

```
void troca(int *x, int *y)
{
    int temp;
    temp=*x;
    *x = *y;
    *y = temp;
}
```

Precisa ser
Função void

Poderia
retornar
um float

Ex: Função void que devolve o maior de 3 valores reais

```
void maior ( float v1, float v2, float v3, float *omaior )
// recebe 3 valores reais e devolve o maior dos 3 no 4º parâmetro
{
    *omaior = v1;
    if (v2 > *omaior) *omaior = v2;
    if (v3 > *omaior) *omaior = v3;
}
```

Exercício 2:
Escrever função com
retorno que devolve o
maior de 3 valores.

Informática

Solução do Exercício

```
float maior ( float v1, float v2, float v3 )
// recebe 3 valores reais e devolve o maior dos 3
{
    float omaior;
    omaior = v1;
    if (v2 > omaior) omaior = v2;
    if (v3 > omaior) omaior = v3;
    return omaior; // return no final !!!
}
```

```
int main()
{
    float big;
    big=maior (2,4,6);
    printf("o maior eh %.2f\n", big);
    system("pause");
    return 0;
}
```

Slide 21

Profa. Cora H.F.

Informática

Exercício 3:

Fazer um programa que leia uma resposta "s" ou "n", utilizando uma função le_s_n, e imprima "Respondeu Sim" ou "Respondeu Não", dependendo do caractere retornado pela função - S ou N.

A função deve ler e consistir um caractere digitado, só concluindo quando o caractere for s, S, n ou N, retornando então o caractere digitado como letra maiúscula.

Slide 22

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Solução do Exercício

```
char le_s_n()
// devolve um caractere 'S' ou 'N', lido do teclado
{
    char c;
    do // só sai do laço se resposta certa
    {
        printf("Resposta? (S/N) ");
        c=toupper(getchar()); // fica maiúscula
        if ((c != 'N') && (c != 'S'))
            printf("\a"); // BEEP
    } while ((c != 'N') && (c != 'S'));
    return c;
}
```

```
int main()
{
    char lido;
    lido = le_s_n();
    if (lido == 'S')
        printf("Respondeu Sim\n");
    else
        printf("Respondeu Nao\n");
    system("pause");
    return 0;
}
```

Slide 23

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática

Fazer um programa completo que leia dois valores, n e p, e calcule a combinação de n elementos, p a p.

Fórmula:

$$\frac{n!}{(p! * (n-p)!)}$$

- Função void para calcular a fatorial
- Função com retorno para calcular a combinação

```
// implementa combinações:
#include <stdio.h>
#include <stdlib.h>

// função que devolve a fatorial de N na variável fat:
void fatorial (int n, float *fat)
{
    // utiliza a função fatorial
    float combinacoes (int n, int p) // retorna o resultado
    {
        // .....
    }
}
```

Slide 24

Profa. Cora H.F. Pinto Ribeiro

UFRGS Informática