

Estruturas de Dados

Exercícios – Capítulos 5 e 6



Referências

Waldemar Celes, Renato Cerqueira, José Lucas Rangel,
Introdução a Estruturas de Dados, Editora Campus
(2004)

Capítulo 5 – Vetores e Alocação Dinâmica

Capítulo 6 – Matrizes

Tópicos

- Recordação de vetores
- Exemplos de uso de vetores:
 - mínimo
 - ordenação
 - pesquisa
- Recordação de matrizes
- Exemplos de uso de matrizes:
 - multiplicação de matrizes

Recordação

`n++` incrementa `n` de uma unidade, depois de ser usado

`n--` decrementa `n` de uma unidade, depois de ser usado

```
if ( expr ) { bloco de comandos } else { bloco de comandos }
```

```
for ( expr_inicial; expr_booleana; expr_de_incremento ) { bloco de comandos }
```

“passar um vetor para uma função” = “passar o endereço inicial do vetor”

exemplo 1: `float variancia (int n, float* v, float m)`

exemplo 2: `void incr_vetor (int n, int *v)`

alocação dinâmica de vetor: através da função `malloc`

exemplo 3: `int *v;`

`v = (int *) malloc(10*sizeof(int));`

Exemplos de uso de vetores

- Função Mínimo:

entrada: v – vetor de inteiros, de dimensão n

saída: M – valor mínimo ocorrendo em v

$M = v[0]$;

para $i = 1$ até $n - 1$, de 1 em 1

se $v[i] < M$ então $M = v[i]$

- Exemplo:

v									
4	2	5	1	1	7	0	8	1	3
0	1	2	3	4	5	6	7	8	9

M	i
4	-
2	1
2	2
1	3
1	4

```
/* Calcula Mínimo - Versão 1 */

#include <stdio.h>

int minimo ( int n, int *v )
{
    int i, M;
    M = v[0];
    for (i = 1; i < n; i++)
        if (v[i] < M) {M = v[i];}
    return M;
}

int main ( void )
{
    int a[ ] = {4, 2, 5, 1, 1, 7, 0, 8, 1, 3};
    printf("%d\n", minimo(10, a));
    return 0;
}
```

Exemplos de uso de vetores

- Função Mínimo:

entrada: v – vetor de inteiros, de dimensão n

saída: k – posição do valor mínimo ocorrendo em v

$k = 0$

para $i=1$ até $n-1$, de 1 em 1

se $v[i] < v[k]$ então $k = i$

- Exemplo:

v									
4	2	5	1	1	7	0	8	1	3
0	1	2	3	4	5	6	7	8	9

k	i
0	-
1	1
1	2
3	3
3	4

```
/* Calcula Mínimo - Versão 2 */

#include <stdio.h>

int minimo ( int n, int *v )
{
    int i, k;
    k = 0;
    for (i = 1; i < n; i++)
        if (v[i] < v[k]) {k = i;};
    return k;
}

int main ( void )
{
    int a[ ] = {4, 2, 5, 1, 1, 7, 0, 8, 1, 3};
    printf("%d\n", minimo(10, a));
    return 0;
}
```

Exemplos de uso de vetores

- Função Mínimo:

entrada: v – vetor de inteiros, de dimensão n

saída: v – vetor modificado com o menor valor em $v[0]$

para $i=n-1$ até 1, decrescendo de 1 em 1

se $v[i] < v[i-1]$ então { troque $v[i]$ com $v[i-1]$ }

- Exemplo:

v										i
4	2	5	1	1	7	0	8	1	3	9
4	2	5	1	1	7	0	1	8	3	8
4	2	5	1	1	7	0	1	8	3	7
4	2	5	1	1	0	7	1	8	3	6
4	2	5	1	0	1	7	1	8	3	5
0	1	2	3	4	5	6	7	8	9	

```

/* Cálculo do Mínimo - Versão 3 */
#include <stdio.h>

void minimo ( int n, int *v )
{ int i, t;
  for (i = n-1; i>0; i--)
    if (v[i] < v[i-1])
      { t = v[i-1]; v[i-1] = v[i]; v[i] = t; };
}

int main ( void )
{
  int a[ ] = {4, 2, 5, 1, 1, 7, 0, 8, 1, 3};
  minimo(10, a);
  printf("%d\n",a[0]);
  return 0;
}

```

Exemplos de uso de vetores

- Função Ordena (“Bubble Sort”):

```
entrada: v – vetor de inteiros, de dimensão n
saída:   v – vetor com os elementos em ordem crescente
para k=1 até n, de 1 em 1
  para i = n-1 até k, decrescendo de 1 em 1
    se v[ i ] < v[ i -1 ] então troque v[ i ] com v[ i -1 ]
```

- “Lógica” do algoritmo:
 - mova o menor elemento de $v[0..n-1]$ para a posição 0
 - mova o menor elemento de $v[1..n-1]$ para a posição 1
 - mova o menor elemento de $v[2..n-1]$ para a posição 2
 - ...

Recordação

Matriz `mat` representada através de vetor de ponteiros, alocado dinamicamente

exemplo: `/* matriz com m linhas e n colunas */`

```
float** mat;
```

```
...
```

```
mat = (float**) malloc(m*sizeof(float*));
```

```
for (i=0; i<m; i++)
```

```
    mat[i] = (float*) malloc(n*sizeof(float));
```

exemplo: `/* passagem de matriz para função */`

```
float acessa (int m, int n, float** mat, int i, int j)
```

Exemplos de uso de matrizes

- Multiplicação de Matrizes:

- entrada: matriz A de dimensão $m \times p$
matriz B de dimensão $p \times n$

- saída: matriz M de dimensão $m \times n$, definida como

$$M_{i,j} = \sum_{k=1}^p A_{i,k} \times B_{k,j}$$

para $i = 0$ até $m - 1$, de 1 em 1

para $j = 0$ até $n - 1$, de 1 em 1

$M[i, j] = 0$

para $k = 0$ até $p - 1$, de 1 em 1

$M[i, j] = M[i, j] + A[i, k] * B[k, j]$

```
/* Multiplicação de Matrizes (representadas por vetor de ponteiros) */  
void mult ( int m, int p, int n, int** A, int** B, int** M)  
{ int i, j, k, t;  
  for (i = 0; i<m; i++)  
    for (j = 0; j<n; j++)  
      {  
        t = 0;  
        for (k = 0; k<p; k++)  
          t = t + A[ i ][ k ]*B[ k ][ j ];  
        M[ i ][ j ] = t;  
      }  
}
```

```

/* Alocação Dinâmica das Matrizes */
int main ( void )
{
    int **A, **B, **M;
    int m = 4, p = 3, n = 5;
    int i,j;
    /* Alocação das matrizes */
    A = (int**) malloc(m*sizeof(int*));
    for (i=0; i<m; i++)
        A[i] = (int*) malloc(p*sizeof(int));
    B = (int**) malloc(p*sizeof(int*));
    for (i=0; i<p; i++)
        B[i] = (int*) malloc(n*sizeof(int));
    M = (int**) malloc(m*sizeof(int*));
    for (i=0; i<m; i++)
        M[i] = (int*) malloc(n*sizeof(int));
    ...
    mult(m,p,n,A,B,M);
    ...
    /* inicialização das matrizes */
    /* multiplicação das matrizes */
    /* etc... */
}

```

Recordação

Matriz `mat` com `n` colunas representada no vetor `v`, alocado dinamicamente

`mat[i][j]` mapeado em `v[i * n + j]`

exemplo: `/* matriz com m linhas e n colunas */`

```
float *v;      /* matriz m x n representada por um vetor */
```

```
...
```

```
v = (float*) malloc(m*n*sizeof(float));
```

Exemplos de uso de matrizes

- Multiplicação de Matrizes:
 - (repita o exemplo para matrizes representadas por vetores)