Programação (CK0226 – 2017.2)

Universidade Federal do Ceará
Departamento de Computação
Prof. Lincoln Souza Rocha
(lincoln@dc.ufc.br)

INTRODUÇÃO À PROGRAMAÇÃO NA LINGUAGEM C

Vetores e Matrizes

Sumário

- Vetores
- Vetores e Ponteiros
- Matrizes

Vetores

São uma estrutura de dados que definem um conjunto enumerável armazenado sequencialmente na memória.

`int vetor[10];`	declaração de vetor com 10 posições
`int vetor[10] = {1,2,3,4,5,6,7,8,9,10};`	declaração/inicialização de vetor com 10 posições
`int vetor[] = {1,2,3,4,5,6,7,8,9,10};`	declaração/inicialização de vetor com 10 posições

`vetor[0] = 1;`
`vetor[1] = 2;`
`vetor[2] = 3;`
`(...)`
`vetor[6] = 7;`
`vetor[7] = 8;`
`vetor[8] = 9;`
	atribuição indexada de valores ao vetor

`vetor[9] = 10;`	atribuição incorreta (invasão de memória)

Vetores

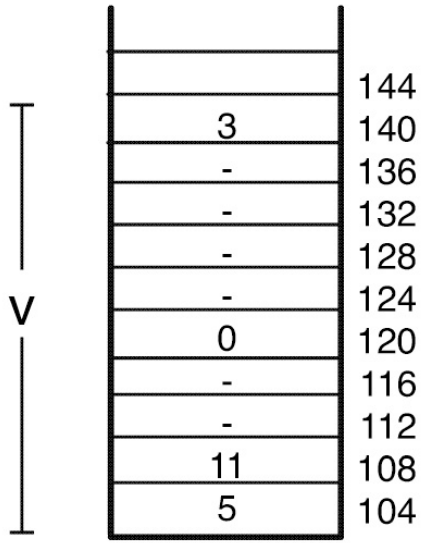
```
int v [10];

v[0] = 5;
v[1] = 11;
v[4] = 0;
v[9] = 3;
```

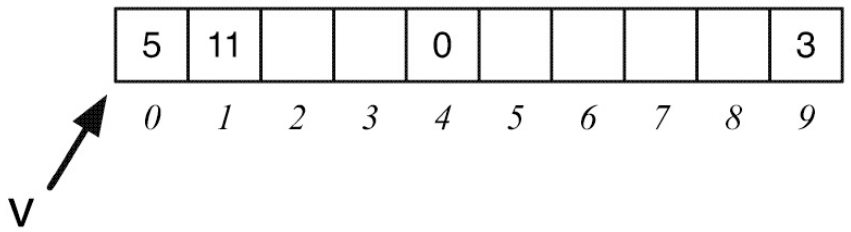
Nota!

O espaço de memória ocupado por um vetor de 10 posições é:

- 10 x 4 bytes = 40 bytes



(a)



(b)

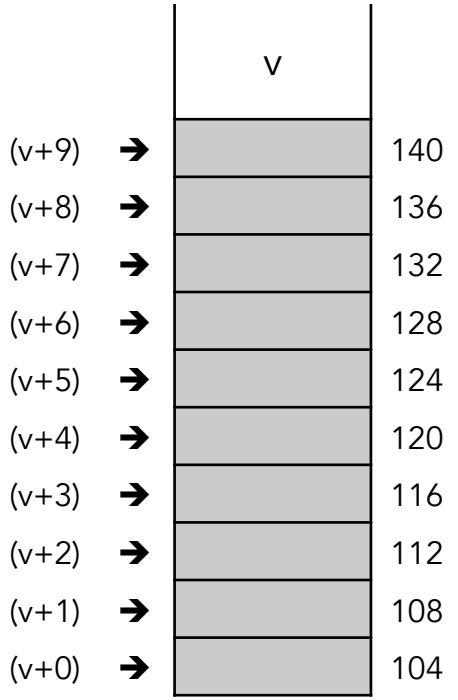
Vetores e Ponteiros

No nome da variável vetor aponta para o endereço do primeiro espaço de memória do vetor. C permite aritmética de ponteiros.

v+0: é o primeiro elemento de v
v+1: é o segundo elemento de v
v+3: é o terceiro elemento de v
(...)
v+9: é o último elemento de v

Nota!

- &v[i] é equivalente a (v+i)
- *(v+i) é equivalente a v[i]



Exercício @ Classe

Implemente um programa que faz o cálculo da média (*m*) e da variância (*v*) de uma amostra de tamanho *n*. Conforme as fórmulas abaixo:

$$m = \frac{\sum_{i=1}^{n} x_i}{n}, v = \frac{\sum_{i=1}^{n} (x_i - m)^2}{n}$$

Vetores: Média e Variância

```
#include <stdio.h>
# include <stdlib.h>
#define N 100 /* dimensão do vetor */

int main (void ) {
   int n; /* número de valores */
   float x[N]; /* vetor dos valores */

   printf("Entre com o numero de valores: ");
   scanf("%d", &n);

   if (n > N) {
      printf("Valor ultrapassa o limite de % d.\ n", N);
      return 1;
   }
   printf("Entre com os valores:\ n");
      for (int i=0; i<n; ++i) {
      scanf("% f", & x[i]);
   }
(...)
```

Vetores: Média e Variância

```
(...)

   float m = 0.0f;
   for (int i=0; i<n; ++i) {
      m += x[i];
   }
   m /= n;

   float v = 0.0f;
   for (int i=0; i<n; ++i) {
      v += (x[i]-m) * (x[i]-m);
   }
   v /= n;

   printf("Media: % f\ nVariancia: % f\ n", m, v);
   return 0;
}
```

Passagem de Vetor para Função

Consiste em passar o endereço da primeira posição do vetor. A função deve ter parâmetro do tipo ponteiro para armazenar valor. Nesse caso, "passar um vetor para uma função" é equivalente a "passar o endereço inicial do vetor". Os elementos do vetor não são copiados para a função, apenas o endereço do primeiro elemento.

A função pode alterar os valores dos elementos do vetor pois recebe o endereço do primeiro elemento do vetor (e não os elementos propriamente ditos).

Vetores: Média e Variância (v2)

```
#include <stdio.h>
# include <stdlib.h>
#define N 100 /* dimensão do vetor */

void captura (int n, float* x);
float media (int n, float* x);
float variancia (int n, float* x, float m);

int main (void ) {
   int n; /* número de valores */
   float x[N]; /* vetor dos valores */

   printf("Entre com o numero de valores: ");
   scanf("% d", & n);

   if (n > N) {
      printf("Valor ultrapassa o limite de % d.\ n", N);
      return 1;
   }
   captura(n,x);
   float m = media(n,x);
   float v = variancia(n,x,m);
   printf("Media: % f\ nVariancia: % f\ n", m, v);
   return 0;
}
```

Vetores: Média e Variância (v2)

```
void captura (int n, float* x) {
   printf("Entre com os valores:\ n");
   for (int i=0; i<n; ++i)
      scanf("% f", & x[i]);
}

float media (int n, float* x) {
   float m = 0.0 f;
   for (int i=0; i<n; ++i)
      m += x[i];
   return m / n;
}

float variancia (int n, float* x, float m) {
   float v = 0.0 f;
   for (int i=0; i<n; ++i)
      v += (x[i]-m) * (x[i]-m);
   return v / n;
}
```

Recursão com Vetores

```
float maximo (int n, float* v) {
   float m = v[0]; /* armazena valor máximo */
   int i;
   for (i=1; i<n; ++i) {
      if (v[i] > m)
         m = v[i];
   }
   return m;
}
```

Recursão com Vetores

$$\mathbf{vetor}\ (n, v) \begin{cases} v[0], & (n = 1) \\ v[0] \oplus \mathbf{vetor}(n-1, \&v[1]), & (n > 1) \end{cases}$$

Recursão com Vetores

```
float maximo (int n, float* x) {
   if (n == 1) {
      return x[0];
   } else {
      float msub = maximo(n-1, & x[1]);
      return x[0] > msub ? x[0] : msub;
   }
}
```

Exercício @ Classe

Avaliar um polinômio significa avaliar o valor numérico do polinômio, y = a(x), para determinado x. Implemente um programa que calcule o valor numérico de um polinômio usando a fórmula abaixo.

$$y = \sum_{i=0}^{g} a_i x^i$$

Matrizes Bidimensionais

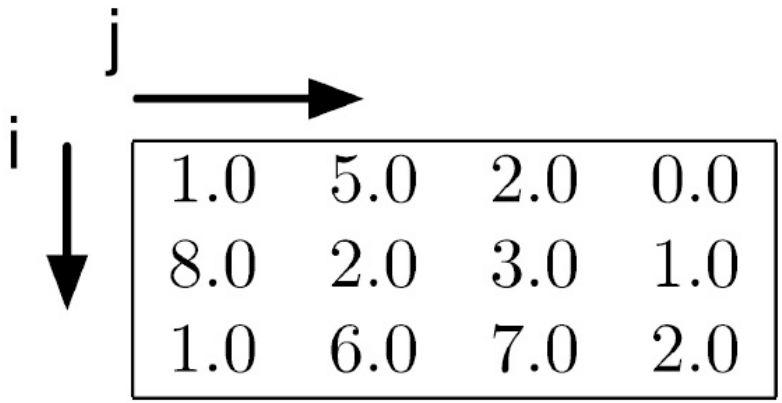
São uma estrutura de dados que definem um conjunto bidimensional enumerável armazenado na memória.

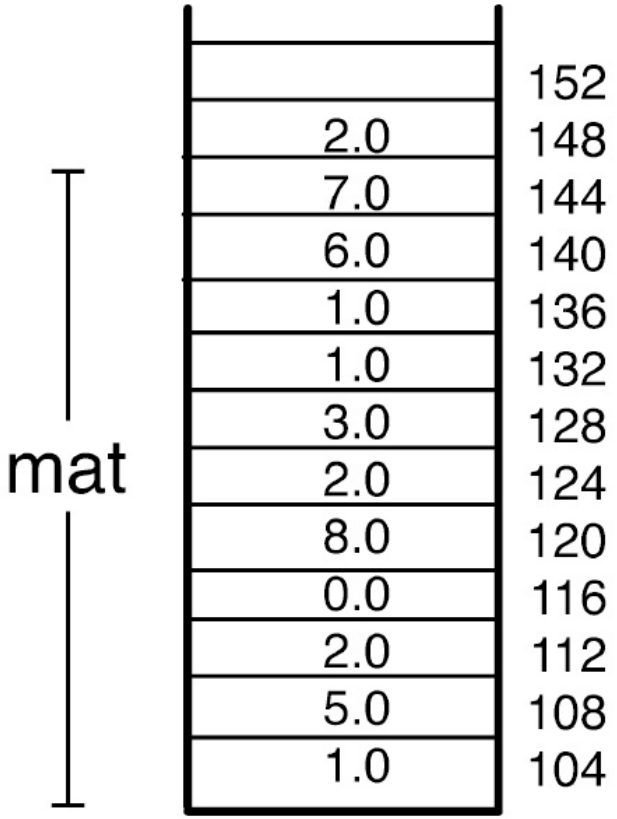
```
float  mat[3][4];
```
declaração de matriz 3x4 posições

```
float mat [3][4] = {
   {1.0f, 5.0f, 2.0f, 0.0f},
   {8.0f, 2.0f, 3.0f, 1.0f},
   {1.0f, 6.0f, 7.0f, 2.0f}
};
```
declaração/inicialização de matriz 3x4 posições

```
mat [0][0] = 1.0f;
mat [0][1] = 5.0f;
mat [0][2] = 2.0f;
mat [0][3] = 0.0f;
```
atribuição indexada de valores à matriz

Matrizes Bidimensionais





Laços Aninhados

```
#include <stdio.h>

int main (void ) {
   float mat [3][4] = {
      {1.0f, 5.0f, 2.0f, 0.0 f},
      {8.0f, 2.0f, 3.0f, 1.0 f},
      {1.0f, 6.0f, 7.0f, 2.0 f}
   };

   for (int i=0; i<3; i++) {
     for (int j=0; j<4; j++) {
        printf("M(% d,% d)=%.1f ", i, j, mat[i][j]);
     }
     printf("\ n");
   }
   return 0;
}
```

```
M (0 ,0)=1.0    M (0 ,1)=5.0    M (0 ,2)=2.0    M (0 ,3)=0.0
M (1 ,0)=8.0    M (1 ,1)=2.0    M (1 ,2)=3.0    M (1 ,3)=1.0
M (2 ,0)=1.0    M (2 ,1)=6.0    M (2 ,2)=7.0    M (2 ,3)=2.0
```

Passagem de Matrizes para Funções

```
#include <stdio.h>

void imprime(int m, float mat[][4]); /* OU void imprime(int m, float (* mat)[4]) */

int main (void ) {
   float mat [3][4] = {
      {1.0f, 5.0f, 2.0f, 0.0 f},
      {8.0f, 2.0f, 3.0f, 1.0 f},
      {1.0f, 6.0f, 7.0f, 2.0 f}
   };
  imprime(3, mat);
  return 0;
}

void imprime(int m, float mat[][4]) {
   for (int i=0; i<m; i++) {
      for (int j=0; j<4; j++) {
         printf("M(% d,% d)=%.1 f ", i, j, mat[i][j]);
      }
      printf("\ n");
   }
}
```

Exercício @ Classe

Uma matriz quadrada, M , é dita simétrica se $M_{i,j} = M_{j,i}$ para qualquer elemento da matriz. Isto é, elementos em lados opostos da diagonal principal da matriz devem ser iguais. Implemente uma função que verifique se uma dada matriz quadrada M é simétrica ou não.

Exercício @ Classe

Uma função que também pode ser útil calcula a transposta de uma matriz. Se M é uma matriz, sua transposta é definida por: $T_{j,i} = M_{i,j}$. Implemente uma função que calcule a transposta de uma matriz quadrada M.

Programação (CK0226 – 2017.2)

Universidade Federal do Ceará
Departamento de Computação
Prof. Lincoln Souza Rocha
(lincoln@dc.ufc.br)