Vetores

Universidade Federal de Rio Grande
Centro de Ciências Computacionais

Thiago Ferreira Dantas

Estruturas compostas

- Estruturas que armazenam dados do mesmo tipo.
- São chamadas de estruturas compostas homogêneas.
- Facilitam o acesso e a manipulação de uma grande quantidade de dados.

Exemplo

- Faça um programa que leia as notas de uma prova para uma classe de 5 alunos, calcule a média da classe e depois mostre todas as notas maiores que a média.

```c
#include <stdio.h>

int main(){
    int nota0, nota1, nota2, nota3,
        nota4, media;

    printf("Digite a nota do aluno 0: ");
    scanf("%d",&nota0);
    printf("Digite a nota do aluno 1: ");
    scanf("%d",&nota1);
    printf("Digite a nota do aluno 2: ");
    scanf("%d",&nota2);
    printf("Digite a nota do aluno 3: ");
    scanf("%d",&nota3);
    printf("Digite a nota do aluno 4: ");
    scanf("%d",&nota4);
```

```c
    media = (nota0+nota1+nota2+nota3+nota4)/5;
    printf("media: %d\n",media);

    if (nota0>media)
       printf("Nota 0: %d \n",nota0);
    if (nota1>media)
       printf("Nota 1: %d \n",nota1);
    if (nota2>media)
       printf("Nota 2: %d \n",nota2);
    if (nota3>media)
       printf("Nota 3: %d \n",nota3);
    if (nota4>media)
       printf("Nota 4: %d \n",nota4);
}
```

Exemplo

- Mas e se for necessário fazer para todos os 50 alunos da disciplina ou para todos alunos da escola?
- Se torna inviável usar a mesma estratégia de solução;
- Uma alternativa é o uso do vetor, o vetor é uma variável que representa uma quantidade de variáveis do mesmo tipo.

Declaração

- Sintaxe de um vetor em c;

tipo *identificador* [numero de variáveis];

- tipo é o tipo das variáveis que devem ser criadas. Exemplo: int, char, float, entre outros;
- *identicador* e o nome que será utilizado para referenciar o conjunto de variáveis;
- numero de variáveis e o tamanho do vetor.

Exemplos de declaração de vetor em C:

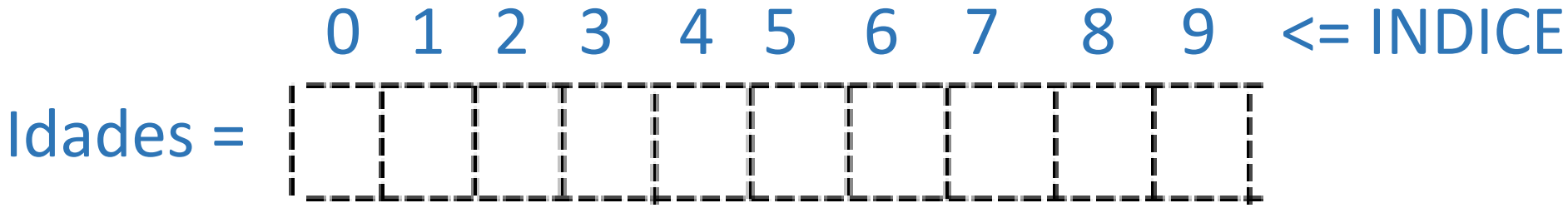
```
char nome[15];
int ano[6];
float peso[200];
double num[50];
```

Representação de um vetor?

- Exemplo, um vetor chamado idades é declarado e apresenta a seguinte estrutura:

```
// Declaração
float idades[10];

// Representação na memória
```



- IMPORTANTE: Os índices vão de 0 a 9 porque em linguagem C os vetores sempre começam pelo índice 0.

Como acessar um vetor?

- Acessar um elemento individual:

É utilizado os colchetes e entre eles é colocado o número referente a posição que deve ser acessada:

```
int idade [8];
idade[4]= 18;// Acessa a posição 4 do vetor
```

- Todo vetor começa na posição zero [0].

```
int idade [8];
idade[4]= 18;// Acessa o quinto elemento do vetor
```

Vetores, Índices e variáveis?

- Também é possível referenciar posições de um vetor usando variáveis inteiras, ou expressões que resultem em números inteiros.

```
// Declaração de variáveis e vetores
int idade [10];
float numeros[15];
int i = 2;
// Usando vetores
idade[i] = 85;
numeros[10+5] = idade[i];
idade[3]=idade[i]+notas[2]
```

Índices inválidos

- Não acesse índices inválidos, ou o sistema operacional fechara seu programa.

```
int peso[10];

peso[0] = 50.3;//CORRETO!
peso[1] = 6,1; //CORRETO!

peso[10]= 2.1; //ERRADO: o índice deve ir de 0 a o tamanho do vetor, ou seja 0 a 9

peso[-1]=4.5; //ERRADO: o índice deve ser positivo
```

Exemplo de Notas usando vetores

```c
#include <stdio.h>

int main() {
  int notas[5];
  int soma=0;
  float media;

  for(i=0; i<5; i++){ /* Percorre o vetor de 0 a 4 */
    printf("Digite a nota do aluno %d: ",i);
    scanf("%d",&notas[i]);
    soma=soma+notas[i];
  }

  media=soma/5;
  printf("Media das notas: %d.\n",media);

  for(i=0; i<5; i++) /* Percorre o vetor de 0 a 4 */
    if(notas[i]>media)
       printf(" Nota[%d]: %d.\n",i,notas[i]);
}
```

Tamanho do vetor

- O tamanho do vetor é determinado em tempo de compilação, portando não é possível colocar uma variável como tamanho do vetor.

```
int y =50;

int nome[y]; //ERRADO
```

- Porém é possível simplificar alterações no código usando constantes.

Constantes em C: Comando #define

- #define deve ficar no início do programa, logo após a declaração das bibliotecas (#include).
- As operações #define são executadas pelo pre-processador do compilador. Ou seja, são processadas antes de compilar o código.
- O pre-processador substitui (literalmente) no código fonte cada constante pelo seu respectivo valor.
- Depois de declarada a constante pode ser usada no programa como se fosse uma variável. Porem ela não pode ter seu valor alterado usando atribuição.

Exemplo:

```c
#include <stdio.h>

#define MAX 100

int main(){
    int vetor[MAX];
    int i;
    for (i=0;i<MAX;i++)
        vetor[i]=i*MAX;
}
```

Inicializando vetores:

- Assim como as variáveis, pode-se atribuir um valor inicial a um vetor.
- Esta inicialização deve ser feita no momento da criação do vetor.
- O trecho de código a seguir exemplifica a inicialização do vetor moedas com os valores 50 na posição 0, 25 na posição 1, 10 na posição 2, 5 na posição 3 e 1 na posição 4.

```
int moedas[5]={50,25,10,5,1};
```

Vetores e Strings:

- Em C uma string e um vetor do tipo char terminada pelo caractere null (‘\0’).
- O exemplo a seguir mostra a criação de um vetor de caracteres ou string.

```
char str[50];
```

Strings

- Podemos armazenar informações caractere a caractere, como mostra o segmento de código a seguir:

```
int i;
char str[20];

for(i=0; i<20; i++){
  printf("Digite o caracter %d: ",i);
  scanf("%c",&str[i]);
}
```

Strings

- Ou podemos ler todos os caracteres como uma string, usando scanf().

```
char str[20];

printf("Digite a sequencia de caracteres: ");
scanf("%s",str);
```

- Note que não tem & antes da variável no scanf. Isso acontece porque str e um vetor e não uma variável de tipo primitivo (int, double, char, ...).

Mais alguns exemplos de string

```
char str[20]="teste";

/* 1 - QUAL A LETRA QUE IMPRIME? */
printf("%c",str[2]);

/* 2 - QUAL O NOVO CONTEUDO DA STRING */
str[0]='P';
```

Mais alguns exemplos de string

```c
char str[20]="teste";

/* 1 - QUAL A LETRA QUE IMPRIME? */
printf("%c",str[2]);

/* 2 - QUAL O NOVO CONTEUDO DA STRING */
str[0]='P';
```

- **Respostas:**

  1. Vai imprimir a letra 's', que é a terceira letra da *string*.
  2. A palavra em `str` passa a ser "Peste". Já que a primeira letra de `str` é substituída por `'P'`.

- Na linguagem **C**, um `char` é representado entre aspas simples (' '), já uma *string*, que é um vetor de `char`, é representado entre aspas duplas (" ").

Exercícios

1. A declaração do vetor está correto? Qual o erro?

```
int vetor(25);
```

2. Qual o elemento da vetor referenciado pela expressão?

```
vetor[4]
```

Exercícios

3. Faça um programa para ler 10 números e mostre-os na ordem inversa.

Dica usar laço com decremento

Exercícios

3. Faça um programa para ler 10 números e mostre-os na ordem inversa.

Resposta:

```c
#include <stdio.h>
#define  N  10
int main() {
  int numeros[ N ];
  int i;

  /* LE OS NUMEROS */
  for (i=0; i < N; i++)
    scanf("%d",&numeros[i]);

  /* MOSTRA OS NUMEROS EM ORDEM INVERSA */
  for (i=(N-1);i >=0; i--)
    printf("%d",numeros[i]);
}
```

Exercícios

4. Faça um programa para ler as notas de 20 alunos e armazená-las em um vetor. Em seguida o programa deve descobrir se existe alguma nota x escolhida pelo usuário. O programa deve permitir que o usuário procure quantas notas quiser. O programa deve encerrar quando uma nota negativa for digitada.

Resposta

```c
#include <stdio.h>
#define N 20
int main() {
  float nota[N],x;
  int i;

  /* LE AS NOTAS E COLOCA NO VETOR */
  printf("Entre com %d notas\n",N);
  for (i=0; i < N; i++) {
    scanf("%f",&nota[i]);
  }

  /* PERGUNTA A NOTA A SER PROCURADA */
  printf("Qual nota deseja procurar? ");
  scanf("%f",&x);

  /* REPETE ATE DIGITAR NOTA NEGATIVA */
  while (x>=0) {
     i = 0;

     /* PROCURA PELA NOTA */
     while ((nota[i] != x) && (i<N))
        i++;

     /* VERIFICA SE A NOTA FOI ACHADA */
     if (i < N)
        printf("nota %f encontrada
                na posicao %d\n",nota[i],i);
     else
        printf("nota %f nao encontrada\n",x);

     /* PERGUNTA PELA PROXIMA NOTA */
     printf("Qual nota deseja procurar? ");
     scanf("%f",&x);
  }
  return 0;
}
```

Exercícios

5. Faça um programa para ler uma palavra e que em seguida mostre a primeira e a última letra da palavra. Dica: Para descobrir o final da palavra, procure pelo caractere especial $'\backslash 0'$ (*null*).

Resposta

```c
#include <stdio.h>
#define TAM 40
int main() {
  char palavra[TAM];
  char prim, ult;
  int i;

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

  /* Primeira letra esta na posicao 0 */
  printf("Primeira letra: %c\n",palavra[0]);


  /* Ultima letra esta na posicao anterior ao '\0' */

  /* Procura o '\0' */
  i=0;
  while(palavra[i]!='\0')
      i++;

  /* Mostra a letra antes do '\0' */
  printf("Ultima letra: %c\n",palavra[i-1]);

  return 0;
}
```

Matrizes e Estruturas de N Dimensões

- Vetores são estruturas de uma dimensão.
- Matrizes são estruturas com duas dimensões.
- A linguagem C permite o uso de estruturas de N dimensões.
- Só precisamos tomar cuidado com o tamanho de memoria usada.

Declaração de Matrizes

- A sintaxe em C para a criação de uma matriz e:

tipo identificador [*nro var 1*][*nro var 2*];

Onde:

- *tipo* e o tipo das variáveis que devem ser criados.

Exemplo: int, char, foat, entre outros;

- identificador e o nome que será utilizado para referenciar o conjunto de variáveis;
- nro var 1 e o numero de variáveis de cada vetor que será criado.
- nro var 2 e o numero de vetores que será criado.

Declaração de Matrizes

- A sintaxe em C para a criação de uma matriz e:

tipo identificador [*nro var 1*][*nro var 2*];

Onde:

- *tipo* é o tipo das variáveis que devem ser criados.

Exemplo: int, char, foat, entre outros;

- identificador e o nome que será utilizado para referenciar o conjunto de variáveis;
- nro var 1 e o numero de variáveis de cada vetor que será criado.
- nro var 2 e o numero de vetores que será criado.

Exemplo de declaração de matriz:

```
int notas[3][8];
int matriz[4][4];
float m2[10][7];
```

Declaração de Estruturas com N Dimensões

- Podemos criar também vetores de vetores de vetores de vetores de vetores ...,
- Ou seja, podemos ter estruturas de múltiplas dimensões.
- Um exemplo de declaração de uma estrutura de 3 dimensões em C:

```
int notas[3][8][4];
double var5d[4][3][5][7][10];
```

Pergunta: Quantas notas foram declaradas?

- 96 Notas.
- Pergunta: Quantas var5d foram declaradas?

4200

Acessando Matrizes

- Para acessar os elementos de uma matriz, usamos dois índices. Um em cada par de colchetes.

Exemplo:

```
int matriz[5][3];
```

matriz[0][0]=1
matriz[3][2]=5
matriz[4][1]=3

	0	1	2
0	**1**		
1			
2			
3			**5**
4		**3**	

```
matriz[5][1]; /* ERRADO - Cuidado */
matriz[1][3]; /* ERRADO - Cuidado */
```

Exercícios

6. Faca um programa que leia uma matriz 4 X 4. O programa deve perguntar o valor de cada posição da matriz, em seguida imprimir esta matriz na tela.

Resposta

```c
#include <stdio.h>
#define COL 4
#define LIN 3
int main(){
  int mat[LIN][COL];
  int i=0,j=0;
  for (i=0; i<LIN; i++) {
    for (j=0; j<COL; j++) {
      printf("Digite o valor da posicao (%d,%d): ",i,j);
      scanf("%d",&mat[i][j]);
    }
  }
  for (i=0; i<LIN i++){
    for (j=0; j<COL; j++)
      printf("%d ",mat[i][j]);
    printf("\n");
  }
  return 0;
}
```

Exercícios

- 7. Refaça o exercício anterior, mas agora imprima a matriz transposta da matriz lida.

Resposta

```c
#include <stdio.h>
#define COL 4
#define LIN 3
int main(){
  int mat[LIN][COL];
  int i=0,j=0;
  for (i=0; i<LIN; i++) {
    for (j=0; j<COL; j++) {
      printf("Digite o valor da posicao (%d,%d): ",i,j);
      scanf("%d",&mat[i][j]);
    }
  }

   /* Mostra a  Transposta - INVERTE O i E O j */
  for (j=0; j<COL; j++){
    for (i=0; i<LIN; i++)
      printf("%d ",mat[i][j]);
    printf("\n");
  }
  return 0;
}
```

Exercícios

- 8. Faca um programa que calcule a soma de duas matrizes 10 X10. Os elementos da primeira matriz devem ser lidos pelo teclado e os elementos da segunda devem ser definidos pela formula $a_{i\,j} = i.j$.

Resposta

```c
#include <stdio.h>
#define LIM 10
int main(){
  int mat1[LIM][LIM];
  int mat2[LIM][LIM];
  int matSoma[LIM][LIM];
  int i=0,j=0;

  for (i=0; i<LIM; i++) {
    for (j=0; j<LIM; j++) {
      printf("Digite o valor da posicao (%d,%d) da matriz 1: ",i,j);
      scanf("%d",&mat1[i][j]);
      /* Calcula a matriz 2 */
      mat2[i][j]=i*j;
    }
  }

  /* Soma as matrizes mat1 e mat2 e coloca a resposta em matSoma */
  for (i=0; i<LIM; i++)
    for (j=0; j<LIM; j++)
        matSoma[i][j]=mat1[i][j]+mat2[i][j];

  printf("\n Matriz 1 + Matriz 2\n");
  for (i=0; i<LIM; i++){
    for (j=0; j<LIM; j++)
      printf("%d ",matSoma[i][j]);
    printf("\n");
  }
  return 0;
}
```

Exercícios

- 9. Leia 2 matrizes 8 X 8, mostre elas na tela e então calcule e mostre a multiplicação entre elas.

Resposta

```c
#include <stdio.h>
#define LIM 8
int main(){
  int mat1[LIM][LIM];
  int mat2[LIM][LIM];
  int resp[LIM][LIM];
  int i=0,j=0,k=0;

  for (i=0;i<LIM; i++) {
    for (j=0;j<LIM; j++) {
      printf("Digite o valor da posicao (%d,%d)
              da matriz 1: ",i,j);
      scanf("%d",&mat1[i][j]);

      printf("Digite o valor da posicao (%d,%d)
              da matriz 2: ",i,j);
      scanf("%d",&mat2[i][j]);
    }
  }
```

```c
  /* NAO ESQUECER DE ZERAR A MATRIZ RESPOSTA */
  for (i=0; i<LIM; i++){
    for (j=0; j<LIM; j++)
      resp[i][j]=0;


  /*  --------  MULTIPLICACAO ---------*/
  for (i=0; i<LIM; i++)
    for (j=0; j<LIM; j++)
      for (k=0; k<LIM; k++)
        resp[i][j]= resp[i][j]+
                    (mat1[i][k]*mat2[k][j]);

 /* Mostra o resultado */
  printf("\n Matriz 1 * Matriz 2\n");
  for (i=0; i<LIM; i++){
    for (j=0; j<LIM; j++)
      printf("%d ",resp[i][j]);
    printf("\n");
  }
  return 0;
}
```