Estruturas de dados

Sistemas para Internet
UFSM

Vetores com ponteiros e passagem de parâmetro por referência

Prof. Leandro O. Freitas
leandro@politecnico.ufsm.br

Retomando...

Sistemas para Internet
UFSM

- Ponteiros:
  - Armazenam endereços de memória nos quais existem variáveis;
  - Podem ser manipulados pelos operadores de endereço (&) e conteúdo (*);

```
int nota;
int *pNota;
pNota = &nota;
```

Ponteiros e vetores

Sistemas para Internet
UFSM

- A forma de acesso aos elementos de vetores vista até aqui utiliza um índice para acessar cada elemento do vetor. Exemplo:

```
vet[0] = 3;
printf (“%d”, vet[0]);
```

- **O nome de um vetor é, na realidade, um ponteiro para o primeiro elemento do vetor**.

- Pode-se usar o nome do vetor (ponteiro) como uma forma de acesso alternativa e mais ágil que a forma usual (via índices).

Ponteiros e vetores

Sistemas para Internet
UFSM

- Dado um vetor vet[4] de inteiros, para acessar o seu primeiro elemento podemos utilizar os seguintes comandos:

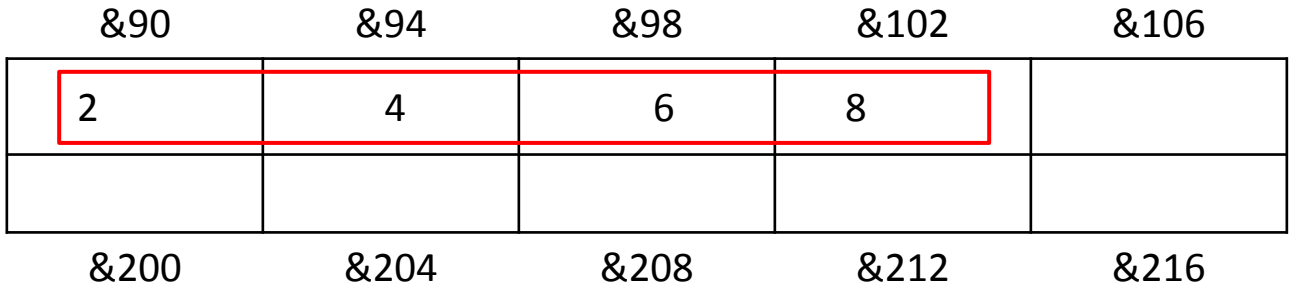
```
aux = vet[0];
aux = *vet;
```

Atribui o valor armazenado na primeira posição do vetor para a variável aux.

- Para acessar o segundo elemento:

```
aux = vet[1];
aux = *(vet + 1);
```

Atribui o valor armazenado na segunda posição do vetor para a variável aux.



Ponteiros e vetores

Sistemas para Internet
UFSM

- Analise o código a seguir:

```
char str[15], *p1;
p1 = str;
```

- O **endereço de memória** do primeiro elemento do vetor *str* foi associado a *p1*.

- Na linguagem C, o nome de um vetor sem um índice é o endereço para o início do vetor.

- É o mesmo que usar o seguinte comando:

```
p1 = &str[0];
```

Ponteiros e vetores

Sistemas para Internet
UFSM

- Portanto, o acesso ao 5º elemento de um vetor pode ser feito das seguintes maneiras:

str[4];

ou

***(p1 + 4);**

- Geralmente, o uso de aritmética é mais rápido do que o uso de indexação.

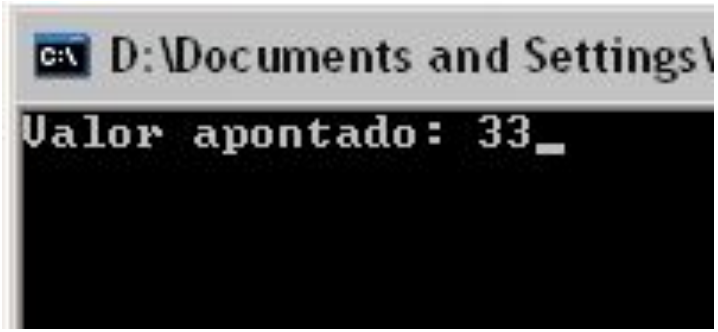
Ponteiros e vetores

Sistemas para Internet
UFSM

- Exemplo: analise o código:

```
int main ()
{
    int *p, vet[4] = {11, 22, 33, 44};
    p = &vet[0];
    printf ("Valor apontado: %d",   *(p + 2)  );
    return 0;
}
```

O que será impresso na tela?



Ponteiros e vetores – Exercício 1

Sistemas para Internet
UFSM

- Exercício 1:
  - Crie um vetor de inteiros com 5 posições e adicione valores quaisquer. Utilizando o método visto anteriormente, mostre o conteúdo do vetor na tela.

- Exercício 2:
  - Crie um vetor com 5 posições;
  - Escreva uma função para atribuir ao vetor os valores recebidos a partir do teclado;
  - Crie funções (sempre usando ponteiros) para realizar as seguintes operações:
    - Localizar o maior valor no vetor;
    - Localizar o menor valor no vetor;
    - Mostrar os números pares do vetor.

Solução exercício 1

Sistemas para Internet
UFSM

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

int main(){
    int v[5], i;
    for (i = 0; i < 5; i++)
        scanf("%d", (v + i));

    for (i = 0; i < 5; i++)
        printf("%d ", *(v + i));
    return 0;
}
```

Solução exercício 2

Sistemas para Internet
UFSM

```c
#include<stdio.h>

void lerVetor (int *p);
void mostraVetor (int *p);
void mostraMenor (int *p);
void mostraMaior (int *p);
void mostraPares (int *p);
 int main(){
     int vet[5], *p;
     p = vet;
     lerVetor (p);
     mostraVetor(p);
     mostraMenor(p);
     mostraMaior(p);
     mostraPares(p);
     return 0;
}

void lerVetor (int *p){
     int i;
     printf("\n Informe os valores do vetor:\n");
     for (i = 0; i < 5; i++)
          scanf("%d", p + i);
}
```

Solução exercício 2

Sistemas para Internet
UFSM

```
26 void mostraVetor (int *p){
27     int i;
28     printf("\n Vetor: ");
29     for (i = 0; i < 5; i++)
30         printf("%d ", *(p + i));
31 }
32
33 void mostraMenor (int *p){
34     int i, menor;
35     menor = *p;
36     for (i = 1; i < 5; i++){
37         if (*(p + i) < menor)
38             menor = *(p + i);}
39     printf("\n Menor valor: %d", menor);
40 }
```

Solução exercício 2

Sistemas para Internet
UFSM

```
42 void mostraMaior (int *p){
43     int i, maior;
44     maior = *p;
45     for (i = 1; i < 5; i++){
46         if (*(p + i) > maior)
47             maior = *(p + i);}
48     printf("\n Maior valor: %d", maior);
49 }
50
51 void mostraPares (int *p){
52     int i;
53     printf("\n Valores pares: ");
54     for (i = 0; i < 5; i++)
55         if (*(p + i) % 2 == 0)
56             printf("%d ", *(p + i));
57 }
```

Passagem de parâmetros por referência

Passagem de parâmetros

Sistemas para Internet
UFSM

- Por valor:
  - Na chamada da função uma cópia dos valores é enviada para os parâmetros da declaração da função;
  - Alterações feitas nestas cópias não geram nenhuma alteração no valor original das variáveis que foram utilizadas na chamada da função.

```
11  void exemploPorValor(int a, int b){
12      a = 10;
13      b = 20;
14  }
15
16  int main()
17  {
18      int x = 5;
19      int y = 2;
20
21      exemploPorValor(x, y);
22
23      printf("%d \n", x);
24      printf("%d \n", y);
25
26      return 0;
27  }
```

a = x;

b = y;

Passagem de parâmetros

Sistemas para Internet
UFSM

- Por referência:
  - Na chamada da função, deve-se passar os endereços de memória dos argumentos para os parâmetros declarados na função.

```
Teste (&x, &y);
```

  - Na declaração da função, os parâmetros que recebem os argumentos devem ser **ponteiros**.

```
void Teste (int *a, int *b) {...}
```

  - Alterações feitas dentro da função, afetam os valores das variáveis usadas na chamada (argumentos).

Passagem de parâmetros

Sistemas para Internet
UFSM

- Por referência:
  - Possibilita “retornar” quantos valores forem necessários, diretamente pelos parâmetros.

```
void teste (int *a, int *b) 3
{
     *a =307;
4    *b = 212;
     }

int main(){
{
      int x = 5;
     1int y = 2;
     2teste(&x, &y);
     5printf("%d \n", x);
      printf("%d \n", y);
      return 0;
      }
```

1. Declaração de **x** e **y** e atribuição valores para eles;
2. Chamada da função **teste** passando os endereços de **x** e **y**;
3. Declaração da função **teste** com ponteiros ***a** e ***b** como parâmetros contendo os endereços de **x** e **y**;
4. Alteração do conteúdo das variáveis apontadas pelos ponteiros **a** e **b**;
5. Imprime os valores de **x** e **y**.

Passagem de parâmetros

Sistemas para Internet
UFSM

- Por referência:
  - ATENÇÃO: Matrizes e *strings* são vetores, portanto, já são consideradas passagem de parâmetros por referência;
  - Não é necessário utilizar o operador de endereço (&) na chamada da função, pois o endereço inicial (índice zero) será repassado para a função.

```
11 void porReferencia(char strg[10]){
12     ...
13
14 }
15
16 int main()
17 {
18     char aula[10] = {"PARAMETRO"};
19     porReferencia(aula);
20     ...
21
22     return 0;
23 }
```

```
void teste(char *pAula)
{
    ...
}

int main()
{
    char aula[10] = {"PARAMETRO"};
    teste(aula);
}
```

	&50	&51	&52	&53	&54	&55	&56	&57	&58	&59
aula =	P	A	R	A	M	E	T	R	O	17 \0

Passagem de parâmetros

Sistemas para Internet
UFSM

- Exemplo

```
void exReferencia (int a[5])
{
      int i;
      for (i=0;i<5;i++)
      {
          a[i] = i;
      }
}

int main(){
{
       int i, x[5] = {3,4,5,6,7};
       exReferencia(x);
       for (i=0;i<5;i++)
       {
            printf("%d \n", x[i]);
       }
       return 0;
}
```

```
void exReferencia(int *pX)
{
     int i;
     for(i=0; i<5; i++)
     {
          *(pX + i) = i;
     }
}

int main()
{
     int i, x[5] = {100,200,300,400,500};
     exReferencia(x);
     for(i=0; i<5; i++)
     {
          printf("%d ", x[i]);
     }
     return 0;
}
```

Exercício 3

Sistemas para Internet
UFSM

- Desenvolver um algoritmo que leia dois valores inteiros. Estes valores devem ser passados por referência para uma função que verificará se são pares. Caso sejam pares, deve-se adicionar 5 a cada um deles. Caso contrário, adicionar 1. Os novos valores das variáveis devem ser apresentados na função main.