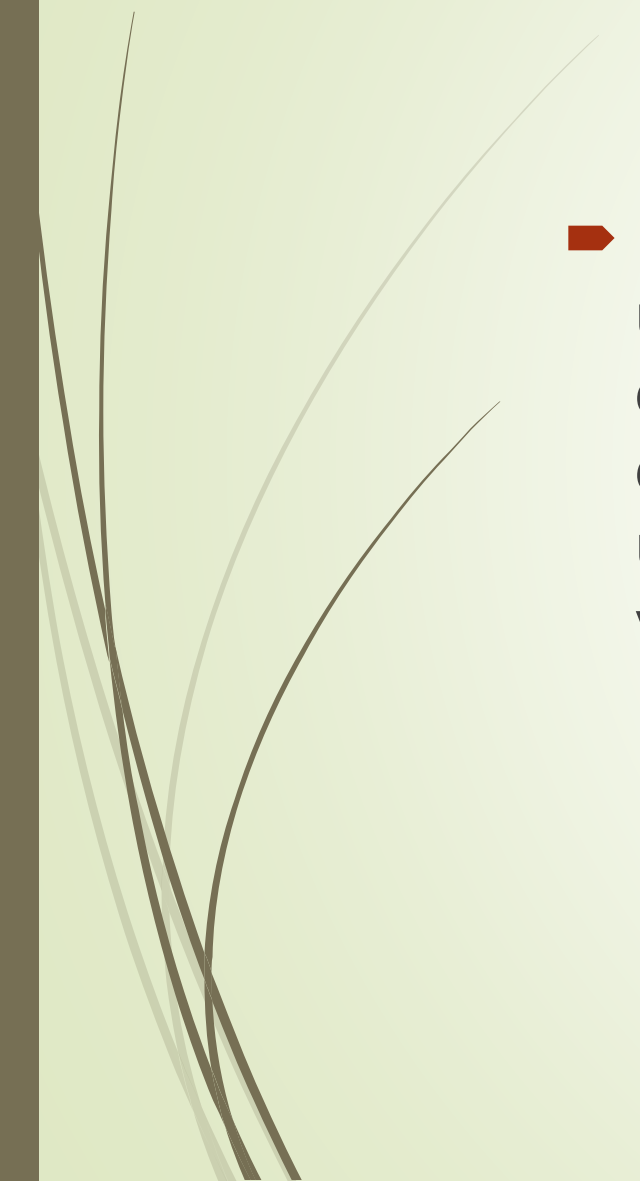


Programação de Computadores Aplicada à Engenharia de Agrimensura e Cartográfica

Miguel Domingos Pereira
migueldomingos2020@gmail.com



Vetor

- É uma estrutura de dados indexada, que pode armazenar uma determinada quantidade de valores do mesmo tipo. Os dados armazenados em um **vetor** são chamados de itens do **vetor**. Para localizar a posição de um item em um **vetor** usamos um número inteiro denominado índice do **vetor**.
- 



Vetores

- Para que seja possível então a solução de outros problemas computacionais, será visto um tipo de estrutura homogênea.
- Esta estrutura denomina-se **Vetor**. O vetor caracteriza-se pelo fato de poder ser definida uma única variável dimensionada com um determinado Tamanho.
- Preencher um vetor significa atribuir valores a todas as suas posições. Assim, deve-se implementar um mecanismo que controle o valor do índice.

Acesso aos elementos do vetor

Para acessar os elementos de um vetor usa-se índices. O índice define a posição da variável dentro do vetor.

A declaração do vetor se faz utilizando um, **nome**, um **tamanho** e um **tipo** para o vetor.

Em todos os vetores tem o primeiro elemento na posição 0(zero).

Assim, se tomarmos "K" como sendo o tamanho do vetor a última posição é a de índice "K-1"

*Vetor[0] = 4; // Coloca 4 na **primeira** posição de "Vetor"*

*Vetor[4] = 8; // Coloca 8 na **última** posição de "Vetor"*

É possível declarar variáveis que guardam vários valores ao mesmo tempo.

```
int x[5];
```

| | | | | |
|------|------|------|------|------|
| x[0] | x[1] | x[2] | x[3] | x[4] |
|------|------|------|------|------|

```
int x[5]; // define um vetor com 5 inteiros
float y[4]; // define um vetor com 4 fracionários

printf("%i", x[2]); // imprime a componente 2 do vetor x
printf("%f", y[0]); // imprime a componente 0 do vetor y

scanf("%i", &x[1]); // lê a componente 1 do vetor x
scanf("%f", &y[3]); // lê a componente 3 do vetor y
```




Memória do vetor



- É um conjunto de variáveis de mesmo tipo, que possuem o mesmo identificador (nome) e são alocadas seqüencialmente na memória.
- Como as variáveis têm o mesmo nome, o que as distingue é um índice que referencia sua localização dentro da estrutura.
- As atribuições em vetor exigem que seja informada em qual de suas posições o valor ficará armazenado.



Exemplos




Faça um algoritmo que preencha um vetor com quinze números inteiros e verifique a existência de elementos iguais a 30, mostrando a posição em que aparecem.




```
#include<stdio.h>
#define MAX 5
int main(){
    int i,vet[MAX];
    for(i=0; i<MAX; i++){
        printf("entre com outro valor:");
        scanf("%d",&vet[i]);}
    for(i=0;i<MAX;i++)
        if(vet[i]==30)
            printf("%d\n",i);
}
```




Exemplos



Faça um algoritmo que preencha um vetor com dez números inteiros, calcule e mostre a quantidade de números negativos e a soma dos números positivos.



```
#include<stdio.h>
#define MAX 5
int main(){
    int i,nega=0,soma=0,vet[MAX];
    for(i=0;i<MAX;i++){
        printf("entre com numero");
        scanf("%d",&vet[i]);}
    for(i=0;i<MAX;i++){
        if(vet[i] < 0){
            nega=nega+1;}
        else if(vet[i]>=0)
            soma=soma+vet[i]}
    printf("numeros de valor negativos: %d\n", nega);
    printf("soma dos numeros positivos: %d\n ", soma);
}
```

- 
- No exemplo, a estrutura de repetição ***para*** foi utilizada para garantir que a variável ***i*** assuma todos os valores possíveis para o índice do vetor.
 - Assim para cada execução de repetição, será utilizada uma posição diferente do vetor e, desta forma, todos os valores do vetor serão mostrados.