Vetores e Matrizes

Vetores

Imagina a seguinte situação:

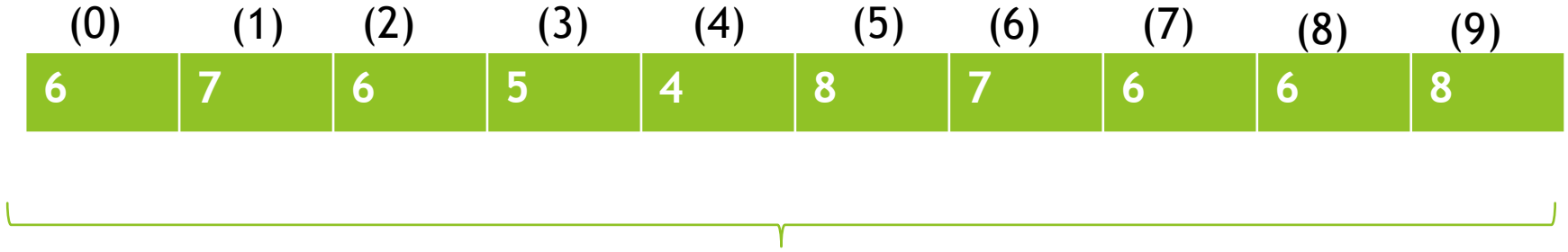
- Você deseja entrar com uma lista de itens e depois de digitar cada item necessita fazer algum tipo de processamento sobre os dados?

- Qual seria a solução?
  - Uma nada inteligente: ter para cada dado uma variável correspondente.

  - **A solução inteligente: usar uma estrutura de memória onde os dados digitados possa ficar armazenados para posterior uso, ou seja, um VETOR!**

Vetores

Vetor de notas das provas finais dos alunos da turma X1



Vetor de tamanho 10

Em **C** um vetor inicia na posição 0 (zero)

Vetores

- C não faz qualquer controle no índice limite da matriz ou de um vetor!
- Se vc tem: **int vetor[20]:**
  - Vetor[0]=1;
  - Vetor[1]=2;
  - Vetor[12]=123;
  - Vetor[15]=8;
  - Vetor[20]=13;
  - **Vetor[21]=45;** **ERRO**

Declarando e usando um vetor

Vetor de notas das provas finais dos alunos da turma X1


```
int vetnotas[10];
int indice;
```

Como se **declara**

```
for(indice=0 ; indice <= 9 ; indice++)
        {
            printf("Entre com a nota %d: ", indice+1);
            scanf("%d", &vetnotas[indice]);
        }
```

Como se **usa**
Lendo dados

Imprimindo os dados de um vetor

Vetor de notas das provas finais dos alunos da turma X1


```
int vetnotas[10];
int indice;
```

Como se **declara**

Como se **usa Imprimindo**

```
for(indice=0 ; indice <= 9 ; indice++)
        printf("Notas %d = %d\n", indice+1, vetnotas[indice]);
```

Exemplo: Faça um programa em C que peça ao usuário duas notas que ele tirou e mostre a média.

```
#include <stdio.h>

int main()
{
    float notas[3];

    printf("Insira sua primeira nota: ");
    scanf("%f", &notas[0]);

    printf("Insira sua segunda nota: ");
    scanf("%f", &notas[1]);

    notas[2] = (notas[0] + notas[1])/2;

    printf("Sua media e: %.2f\n", notas[2]);
}
```

Vetor de String em C

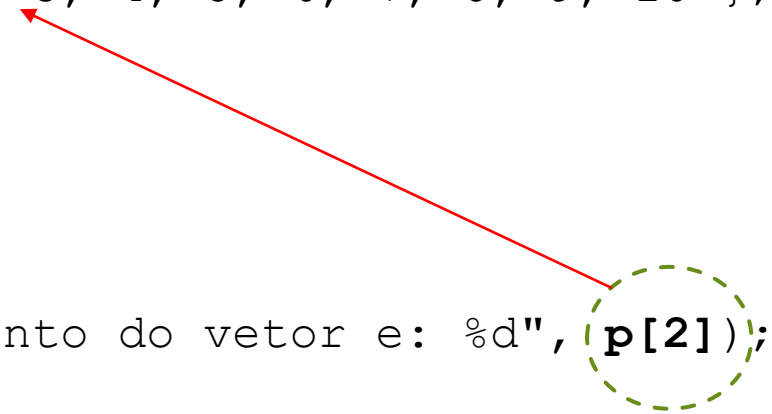
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
'H'	'e'	'l'	'l'	'o'	' '	'w'	'o'	'r'	'l'	'd'	'\o'	?	?	?	?	?	?	?	?

Vetor de string

```c
#include <stdio.h>

int main()
{
    char ola[4];
    char ola2[4]= {'A','L','O','\0'};
    char ola3[4]="ALE";

    ola[0] = 'o';
    ola[1] = 'l';
    ola[2] = 'a';
    ola[3] = 0;

    printf(ola);
    printf("\n");

    printf(ola2);
    printf("\n");

    printf(ola3);
    printf("\n");
    return 0;
}
```

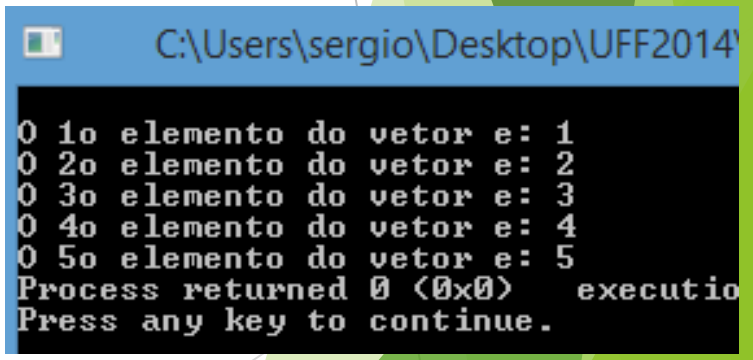
Exemplo 2

```
#include <stdio.h>

int main ()
{
        int matrix [10] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

        int *p;

        p=matrix;

        printf ("O terceiro elemento do vetor e: %d", p[2]);

        return(0);
}
```



Exemplo 3

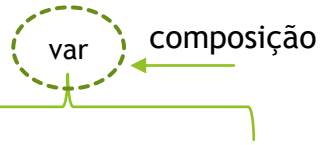
```
#include <stdio.h>

int main ()
{
        int matrix [10] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
        int *p;
        p=matrix;

    printf ("\nO 1o elemento do vetor e: %d", *(p));
    printf ("\nO 2o elemento do vetor e: %d", *(p+1));
    printf ("\nO 3o elemento do vetor e: %d", *(p+2));
    printf ("\nO 4o elemento do vetor e: %d", *(p+3));
    printf ("\nO 5o elemento do vetor e: %d", *(p+4));

    return(0);
}
```

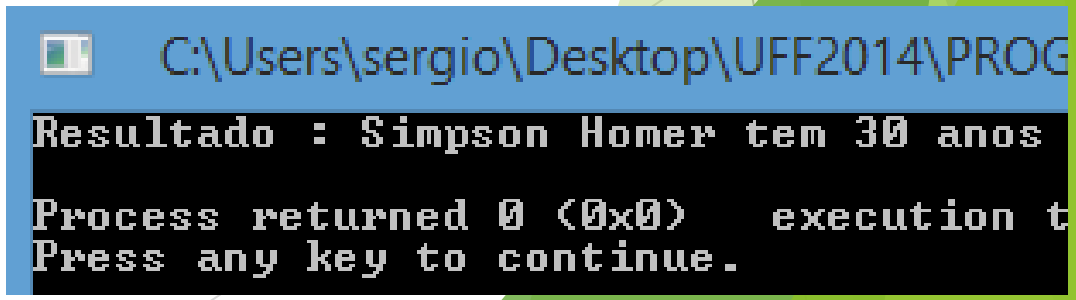


sprintf

Descrição: A diferença entre **printf** e **sprintf** e que printf retorna o resultado para a saída padrão (tela), enquanto **sprintf** retorna o resultado em uma variável. Isto é muito conveniente, porque você pode simplesmente digitar a frase que você quer ter e **sprintf** lida com a própria conversão e coloca o resultado na string que você deseja.

```
#include <stdio.h>
    #include <string.h>

    int main() {
        char var[256];
        char sobrenome[] = "Simpson";
        char nome[] = "Homer";

        int idade = 30;

        sprintf(var, "%s %s tem %d anos", sobrenome, nome, idade);

        printf ("Resultado : %s\n", var);

        return 0;
    }
```



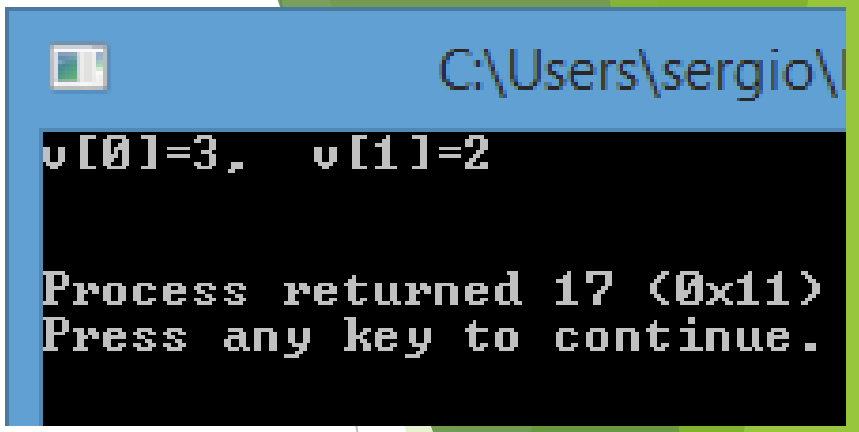


http://pt.wikibooks.org/wiki/Programar_em_C/Strings

Vetor e função

```
#include  <stdio.h>

void troca(); /*Declara a função como não retornando qualquer valor*/

main()
{

   int v[2];
   v[0]=2;
   v[1]=3;

   troca (v);

   printf ("v[0]=%d,v[1]=%d", v[0],v[1]);

}

void troca(int x[2])
{

   int t;
   t=x[0];
   x[0]=x[1];
   x[1]=t;

}
```

Vetor passado com parâmetro



Quando o nome do vetor ou matriz é usado em uma função, como parâmetro, desacompanhado dos [], representa o **endereço de memória** em que o vetor ou matriz estão armazenados!!!

Em outras palavras, v é o endereço de v[0]

Ou seja, o vetor ou matriz são passados por **referência**!!!

```c
#include <stdio.h>
#define MAX 5

void bu(int *a);


int main()
{
 int i, vet[MAX];

 // Lê MAX valores
 for(i = 0; i < MAX; i++)
 {
  printf("Digite um valor %d/%d: ", i+1, MAX);
  scanf("%d", &vet[i]);
 }

 // mexe nos valores
 bu(vet);

 // Imprime os valores
 printf("\n\nValores\n");
 for(i = 0; i < MAX; i++)
 {
  printf("%d - ", vet[i]);
 }

 return 0;
}
```

O que faz este programa???

```c
// Função bu
void bu(int *a)
{
 int i, j, tmp;
 for(i = 0; i < MAX; i++)
 {
  for(j = i+1; j  < MAX; j++)
  {
   if(a[j] < a[i])
   {
    tmp = a[i];
    a[i] = a[j];
    a[j] = tmp;
   }
  }
 }
}
```

Uma ou Outra

```c
void bu(a[5])
{
 int i, j, tmp;
 for(i = 0; i < MAX; i++)
 {
  for(j = i+1; j  < MAX; j++)
  {
   if(a[j] < a[i])
   {
    tmp = a[i];
    a[i] = a[j];
    a[j] = tmp;
   }
  }
 }
}
```