

AULA 06

{introcomp}

VETORES I

Vetores

Vetores e sua importância

Representação

Definição

Operações



Vetores

•Vantagens:

- Armazenar vários dados de mesmo tipo
- Utilizados em diversos algoritmos

Algoritmos de Ordenação

Pesquisa Linear

Busca Binária



Definição

- Estrutura de dados linear, composta por um número fixo (finito) de elementos de mesmo tipo.

5	35	1	234	76	23	45
---	----	---	-----	----	----	----

23.48	12.65	54.52	31.44	74.22	23.60	90.65	12.31
-------	-------	-------	-------	-------	-------	-------	-------

l	n	t	r	o	c	o	m	p	\0
---	---	---	---	---	---	---	---	---	----



Representação

placar[8]

23.48	12.65	54.52	31.44	74.22	23.60	90.65	12.31
-------	-------	-------	-------	-------	-------	-------	-------

vet[7]

5	35	1	234	76	23	45
---	----	---	-----	----	----	----

nome[10]

I	n	t	r	o	c	o	m	p	\0
---	---	---	---	---	---	---	---	---	----



Declaração

```
tipo nome_do_vetor[tamanho_do_vetor];
```

- Exemplo:

```
int vet[32];
```

```
char var[242];
```

```
double numero[34];
```



Operações

- **Acesso** através dos índices.
- **Atribuição** através dos índices.

	0	1	2	3	4	5	6	7
placar[8]	23.48	12.65	54.52	31.44	74.22	23.60	90.65	12.31

```
valor = placar[0]; /*valor = 23.48*/
```

```
num = placar[5]; /*num = 23.60*/
```

```
placar[3] = 41.00;
```



Operações

- **Acesso** através dos índices.
- **Atribuição** através dos índices.

	0	1	2	3	4	5	6	7
placar[8]	23.48	12.65	54.52	41.00	74.22	23.60	90.65	12.31

```
valor = placar[0]; /*valor = 23.48*/
```

```
num = placar[5]; /*num = 23.60*/
```

```
placar[3] = 41.00;
```



Operações

Atribuição de valores.

SEM UTILIZAR VETOR

```
int a,b,c,d,e;
```

```
a = 1;
```

```
b = 2;
```

```
c = 3;
```

```
d = 4;
```

```
e = 5;
```

UTILIZANDO VETOR

```
int vet[5];
```

```
vet [5] = {1,2,3,4,5};
```



Operações

Comandos de Repetição podem ser utilizados para atribuir valores ao vetor por meio dos índices

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

```
int main()
```

```
{
```

```
    int i;
```

```
    int vetor[5];
```

```
    for(i=0;i<5;i++)
```

```
        vetor[i] = 10;
```

```
    return 0;
```

```
}
```

Índices	0	1	2	3	4
vetor[5]	10	10	10	10	10



Operações

Atribuição entre vetores (**INCORRETO**)

```
#include <stdio.h>
int main()
{
    int idades_1[3] = {4, 2, 3};
    int idades_2[3] = {5, 23, 4};

    idades_1 = idades_2;
    return 0;
}
```



Operações

Atribuição entre vetores (**CORRETO**)

```
#include <stdio.h>
int main()
{
    int idades_1[3] = {25,34,24};
    int idades_2[3] = {12,2,32};
    int i;
    for (i = 0; i < 3; i++)
    {
        idades_1[i] = idades_2[i];
    }
    return 0;
}
```



Operações

Acesso indevido (Segmentation Fault)

```
#include <stdio.h>
int main()
{
    int idades[25], i;
    for (i= 0; i < 5000; i++)
        scanf ("%d", &idades[i]);
    return 0;
}
```



Operações

Definição de tamanho em tempo de execução

```
#include <stdio.h>
int main()
{
    int n;
    printf ("Digite a quantidade de notas: ");
    scanf ("%d", &n);
    int notas[n];

    for (i = 0; i < n; i++)
        scanf ("%d", &notas[i]);

    printf ("Notas: ");
    for (i = 0; i < n; i++)
        printf ("%d ", notas[i]);
    return 0;
}
```



Operações

Passando vetores como parâmetros de uma função

```
void ImprimeVetor(int vet[],int tam)
{
    int i;

    for(i=0;i<tam;i++)
        printf("%d",vet[i]);
}

int main()
{
    int vet[5] = {1,2,3,4,5};

    ImprimeVetor(vet,5);

    return 0;
}
```



Operações

Vetores como retorno de uma função (**INCORRETO**)

```
#include <stdio.h>

int RetornaVetor(int vetor[], int tam)
{
    ...
    return vetor;
}

int main()
{
    ...
    RetornaVetor(vet, 5);
    return 0;
}
```



Operações

Vetores como retorno de uma função (**CORRETO**)

```
#include <stdio.h>

int RetornaVetor(int vetor[], int tam)
{
    ...
    return vetor[3];
}

int main()
{
    ...
    RetornaVetor(vet, 5);
    return 0;
}
```



Exemplos

Dados 10 números, imprimi-los na tela utilizando vetores.



Exemplos

```
#include <stdio.h>

int main()
{
    int vet[10];
    register int i;

    for (i = 0; i < 10; i++)
        scanf ("%d", &vet[i]);

    for (i = 0; i < 10; i++)
        printf ("%d ", vet[i]);

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



Exemplos

Dada uma sequência de N números, exibi-los na ordem inversa.



Exemplos

```
#include <stdio.h>

int main()
{
    int N, i;
    scanf ("%d", &N);
    int vet[N];

    for (i = 0; i < N; i++)
        scanf ("%d", &vet[i]);

    for (i = N-1; i >= 0; i--)
        printf ("%d ", vet[i]);

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



Exemplos

Dada uma sequência de N números, utilizando vetor, exiba apenas os pares.



Exemplos

```
#include <stdio.h>
void pegaNumeros(int vetor[], int tam)
{
    int i;
    for (i = 0; i < tam; i++)
        scanf ("%d", &vetor[i]);
}
void imprimeNumeros(int vetor[], int tam)
{
    int i;
    for (i = 0; i < tam; i++)
        if (vetor[i]%2 == 0)
            printf ("%d ", vetor[i]);

    printf ("\n");
}
```



Exemplos

```
int main()
{
    int N, i;

    scanf ("%d", &N);
    int vet[N];

    pegaNumeros(vet, N); /*Lê nos números do teclado*/

    imprimeNumeros(vet, N); /*Exibe os números na tela*/
    return 0;
}
```



Exemplos

Dada uma sequência de N números, verificar se um número qualquer, lido pelo teclado, está incluído nesta.



Exemplos

```
#include <stdio.h>
#define TRUE 1
#define FALSE 0

int buscaLinear (int seq[], int chave, int tam)
{
    register int i;
    for (i = 0; i < tam; i++)
    {
        if (seq[i] == chave)
            return TRUE;
    }
    return FALSE;
}
```



Exemplos

```
int main()
{
    int N, num, i;
    scanf ("%d", &N);
    int vet[N];

    for (i = 0; i < N; i++)
        scanf ("%d", &vet[i]);

    scanf ("%d", &num);

    if (buscaLinear(vet, num, N))
        printf ("O número %d pertence à sequência!\n", num);
    else
        printf ("O número %d não pertence à sequência!\n",
num);
    return 0;
}
```



Exemplos

Dada uma sequência de N números, verificar se o conjunto é palíndromo.



Exemplos

```
#include <stdio.h>
void pegaNumeros(int vetor[], int tam)
{
    register int i;
    for (i = 0; i < tam; i++)
        scanf ("%d", &vetor[i]);
}
int ehPalindromo(int vetor[], int tam)
{
    register int i, j;
    for (i = 0, j = tam-1; i < tam; i++, j--)
    {
        if (vetor[i] != vetor[j])
            return 0;
    }

    return 1;
}
```



Exemplos

```
int main()
{
    int N;
    scanf ("%d", &N);
    int vet[N];

    pegaNumeros (vet, N); /*Lê os números do teclado*/

    if (ehPalindromo (vet, N))
        printf ("O conjunto de números é palíndromo!\n");
    else
        printf ("O conjunto de números não é palíndromo!\n");

    return 0;
}
```



Exemplos

Dado o gabarito de uma prova de 10 questões de múltipla escolha, verificar a pontuação do aluno.



Exemplos

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

void preencheVetor(char vet[], int tam)
{
    register int i;
    for (i = 0; i < tam; i++)
    {
        printf ("Questão %d = ", i+1);
        scanf ("%c", &vet[i]);
        getchar();
    }
}
```



Exemplos

```
int contaPontos(char gab[], char vet[], int tam)
{
    register int i;
    int pontos = 0;
    for (i = 0; i < tam; i++)
    {
        if (gab[i] == vet[i])
            pontos++;
    }

    return pontos;
}
```



Exemplos

```
int main()
{
    char escolha[10], gabarito[10];

    printf ("\t\t\t\tGABARITO\n");
    preencheVetor(gabarito, 10);
    system("clear");
    printf ("\t\t\t\tRESPOSTAS\n");
    preencheVetor(escolha, 10);
    system("clear");
    printf ("\t\t\t*****PONTUAÇÃO***** \n\t\t\t\t%d\n",
contaPontos(gabarito, escolha, 10));

    return 0;
}
```



Algoritmos de Ordenação

Algoritmo de ordenação é uma sequência de instruções (algoritmo) que coloca os elementos de uma dada sequência em uma certa ordem.

- Exemplos de Algoritmos de Ordenação
 - Insert Sort
 - Select Sort
 - Bubble Sort
 - Quicksort
- Exemplos de Ordem de ordenação
 - Ordem Crescente
 - Ordem Decrescente
 - Ordem Lexicográfica



Bubble Sort

```
void bsort (int vetor[], int n) {  
    int k,j, aux;  
    for (k = 1; k < n; k++)  
    {  
        for (j = 0; j < n-1; j++)  
        {  
            if (vetor[j] > vetor[j + 1])  
            {  
                aux = vetor[j];  
                vetor[j] = vetor[j + 1];  
                vetor[j + 1] = aux;  
            }  
        }  
    }  
}
```



Exemplo Bubble Sort

```
int main()
{
    int N, i;
    scanf ("%d", &N);
    int vet[N];
    for (i = 0; i < N; i++)
        scanf ("%d", &vet[i]);
    bsort(vet,N);
    imprimeNumeros(vet,N);
    return 0;
}
```



AULA 06

{introcomp}

VETORES I