ESTRUTURAS COMPOSTAS

Variáveis Compostas Unidimensionais

VETOR

slides desenvolvidos pela Profa. Rosely Sanches

ESTRUTURAS COMPOSTAS

- Pode-se organizar tipos simples em tipos mais complexos formando as ESTRUTURAS COMPOSTAS
- Exemplo:
  - variáveis compostas unidimensionais (VETOR)

VETOR

- O conceito de VETOR será introduzido através de um exemplo.
- Suponhamos o seguinte problema:

Calcular a média aritmética das notas de 3 alunos.

Exibir a média e as notas que estão abaixo da média

Calcular a média aritmética das notas de 3 alunos.
Exibir a média e as notas que estão abaixo da média

1ª Solução (PÉSSIMA)

- Ler as três notas uma primeira vez para calcular a média.
- Ler novamente cada nota para comparar com a média e verificar quais notas estão abaixo da média

```
int main(){ // PESSIMO.C
 float sum=0, nota, media;
 int i;
for(i= 0; i<3;i++){
  printf("Nota="); scanf("%
  sum = sum + nota;
 }

 media = sum/3;
 printf("Media=%.1f",media);

 for(i= 0; i<3;i++){
  printf("Nota="); scanf("%f",&nota);
  if(nota<media){
     printf("%.1f está abaixo d
  }
 }

return 0;
}
```

Ler as notas uma primeira vez para calcular a média

Ler novamente cada nota para comparar com a média e verificar quais notas estão abaixo da média

```
int main(){ // PESSIMO.C
 float sum=0, nota, media;
 int i;
 for(i= 0; i<3;i++){
   printf(“Nota=”); scanf(“%f”,&nota);
   sum = sum + nota;
  }

  media = sum/3;
  printf(“Media=%.1f”,media);

  for(i= 0; i<3;i++){
   printf(“Nota=”); scanf(“%f”,&nota);
   if(nota<media){
      printf(“%.1f está abaixo da média”,nota);
   }
  }

 return 0;
}
```

INEFICIENTE

Calcular a média aritmética das notas de 3 alunos.
Exibir a média e as notas que estão abaixo da média

2ª Solução (RUIM)

- Ler as três notas e armazenar na memória do computador, dando um nome diferente para cada nota.

```
int main(){ // RUIM.C
  float n1, n2, n3, media;

  printf("Notas:"); scanf("%f %f %f",&n1, &n2, &n3);
  media = (n1+n2+n3)/3;
  printf("Média=%.1f",media);

   if(n1 < media){
       printf("%.1f está abaixo da média",n1);
     }
     if(n2 < media){
        printf("%.1f está abaixo da média",n2);
     }
     if(n3 < media){
        printf("%.1f está abaixo
     }
```

O programa só vale para três notas

IMPRATICÁVEL!

Qual seria o algoritmo para uma relação de **1000** notas?
Associarmos um nome para cada nota?

Calcular a média aritmética das notas de 1000 alunos.
Exibir a média e as notas que estão abaixo da média

3ª Solução (ABORDAGEM MAIS REALISTA)

- Associar o nome NOTA ao CONJUNTO ORDENADO de notas

NOTA = {N1, N2, ... N1000}

1ª 2ª 1000ª

Calcular a média aritmética das notas de 1000 alunos.
Exibir a média e as notas que estão abaixo da média

3ª Solução (ABORDAGEM MAIS REALISTA)

NOTA = {N1, N2, ... N1000}

1ª 2ª 1000ª

- para fazer referência ou selecionar uma nota específica usar um **índice**
- Exemplo:
  - a 3ª nota é indicada por NOTA[3]
  - a 1000ª nota é indicada por NOTA[1000]
  - uma kª nota é indicada por NOTA[k]

Calcular a média aritmética das notas de 1000 alunos.
Exibir a média e as notas que estão abaixo da média

3ª Solução (ABORDAGEM MAIS REALISTA)

- Chamada VARIÁVEL INDEXADA
- Junta-se o nome dado ao conjunto ordenado com um índice

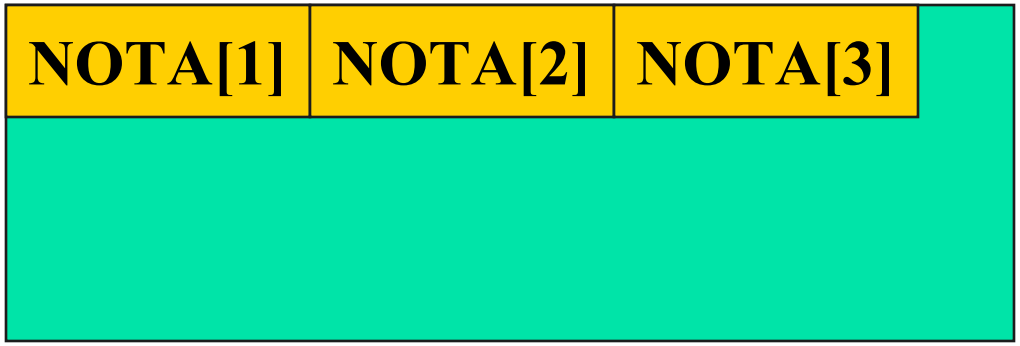
- p… a
  n…
- Exemplo:
  - a 3ª nota é indicad…]
  - a 1000ª nota é indi… NOTA[1000]
  - uma kª nota é indicada por NOTA[k]

NOME

ÍNDI CE

VARIÁVEL INDEXADA

- Cada variável indexada é associada a uma posição de memória, como acontece com variáveis simples.
- Exemplo:



VETOR

- Um VETOR é um conjunto ordenado que contém um número fixo de elementos
- Todos os elementos do vetor devem ser do mesmo tipo

Declaração de Variável Indexada Unidimensional

- Deve ser especificado o número máximo de elementos do conjunto
- Deve ser especificado o tipo dos elementos do conjunto
- Exemplo:

```
float X[100];
```

Declaração de Variável Indexada Unidimensional

- cificado o número máximo de
  conjunto
- v o tip
  conj
- Exemplo

Tipo dos elementos do conjunto

Nome da Variável

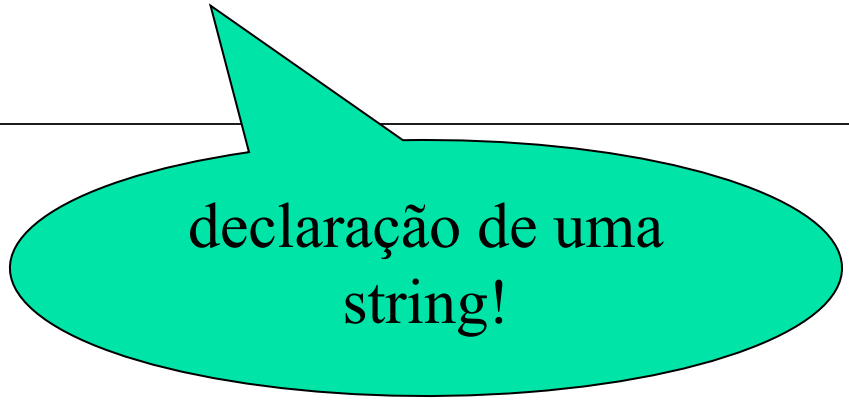
Número máximo de elementos do conjunto

```
float X[100];
```

Declaração de Variável Indexada Unidimensional - Exemplos

- Declaração de um vetor S com no máximo 100 elementos do tipo caracter

```
char S[100];
```



Declaração de Variável Indexada Unidimensional

- Pode ser definida uma constante e esta ser utilizada no dimensionamento.
- Exemplo:

```
//Início do programa C
const int MAX = 5; OU
#define MAX 5
//declaração:
 float X[MAX];
```

VETOR - Exemplo 1

- Ler um conjunto de 100 notas, armazená-las no vetor denominado NOTA e escrever este vetor.

VETOR - Exemplo 1

ler NOTA[1]
ler NOTA[2]
ler NOTA[3]
ler NOTA[4]
ler NOTA[5]
ler NOTA[6]
……..
ler NOTA[98]
ler NOTA[99]
ler NOTA[100]

usar COMANDO DE REPETIÇÃO

Qual o Comando de Repetição mais indicado?

REPETIÇÃO CONTADA

VETOR - Exemplo 1

```
scanf("%f",&nota[0]);
scanf("%f",&nota[1]);
scanf("%f",&nota[2]);
scanf("%f",&nota[3]);
scanf("%f",&nota[4]);
scanf("%f",&nota[5]);
scanf("%f",&nota[6]);
 ……..
scanf("%f",&nota[97]);
scanf("%f",&nota[98]);
scanf("%f",&nota[99]);
```

```
for(j=0; j< 100; j++){
    scanf("%f",&nota[j]);
}
```

```
scanf("%f",&nota[0]);
scanf("%f",&nota[1]);
scanf("%f",&nota[2]);
scanf("%f",&nota[3]);
scanf("%f",&nota[4]);
scanf("%f",&nota[5]);
scanf("%f",&nota[6]);
 ……..
scanf("%f",&nota[97]);
scanf("%f",&nota[98]);
scanf("%f",&nota[99]);
```

```
for(j=0; j< 100; j++){
    scanf("%f",&nota[j]);
}
```

mesmo efeito que

```
for(i=0; i< 100; i++){
    scanf("%f",&nota[i]);
}
```

mesmo efeito que

```
for(k=0; k< 100; k++){
    scanf("%f",&nota[k]);
}
```

I, J e K são apenas índices que assumem um valor e que junto com o nome dado ao conjunto formam a variável indexada NOTA

```
for(j=0; j< 100; j++){
    scanf("%f",&nota[j]);
}
```

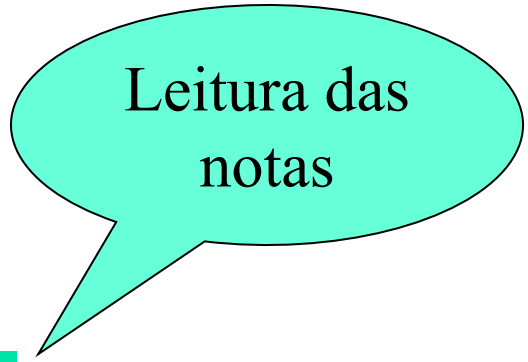
mesmo efeito que

```
for(i=0; i< 100; i++){
    scanf("%f",&nota[i]);
}
```

mesmo efeito que

```
for(k=0; k< 100; k++){
    scanf("%f",&nota[k]);
}
```

```
#include <stdio.h>
#include <stdlib.h>
# define MAX 10
int main()
{
    float nota[MAX], media, soma = 0;
    int i;
    printf("Entre com as notas dos alunos:");
    for(i=0; i<MAX; i++) //leitura das notas
            scanf("%f", &nota[i]);
```



```
for(i=0; i<MAX; i++)
        soma = soma + nota[i];
    media = soma/MAX;
```

Cálculo da média

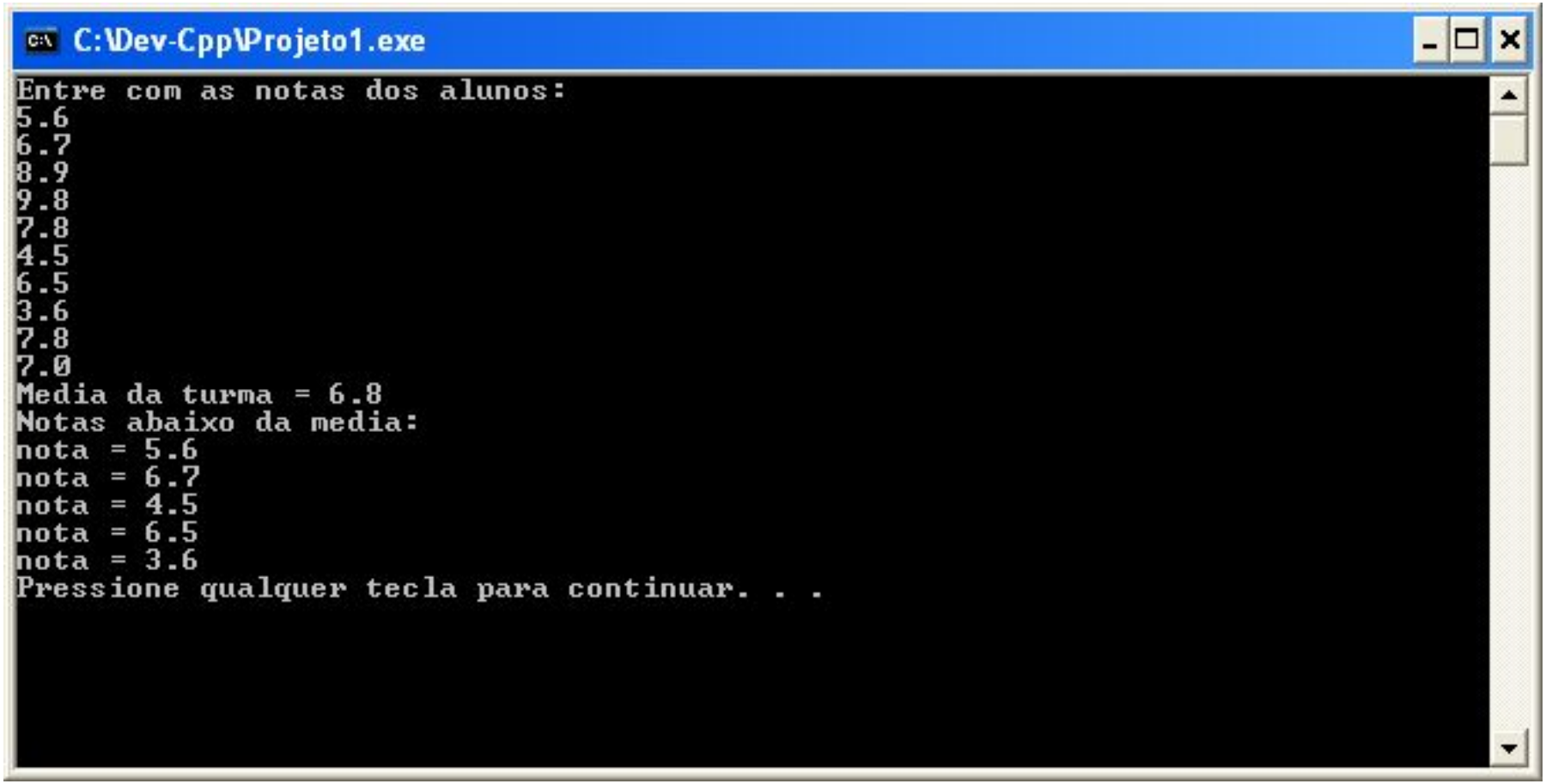
```
    printf("média da turma = %.1f\n", media);
```

Exibe a média

```
    printf("Notas abaixo da media:\n");
    for(i=0; i<MAX; i++)
            if (nota[i] < media)
               printf("nota = %.1f\n", nota[i]);
```

Escrita de todas as notas

```
  return 0;
}
```


C:\Dev-Cpp\Projeto1.exe
Entre com as notas dos alunos:
5.6
6.7
8.9
9.8
7.8
4.5
6.5
3.6
7.8
7.0
Media da turma = 6.8
Notas abaixo da media:
nota = 5.6
nota = 6.7
nota = 4.5
nota = 6.5
nota = 3.6
Pressione qualquer tecla para continuar. . .

Exercício 1

- Elaborar um código em linguagem C que lê um conjunto de 30 valores inteiros e os coloca em um vetor. Calcular e mostrar:
  - Os números pares;
  - A quantidade de números pares;
  - Os números ímpares
  - A quantidade de números ímpares;

```
#include <stdio.h>
#include <stdlib.h>
const int MAX = 30;
int main()
{
    int num[MAX], i, qpar=0, qimpar=0;

    printf("Entre com os numeros:");
    for(i=0; i<MAX; i++) //leitura das notas
            scanf("%d", &num[i]);
```

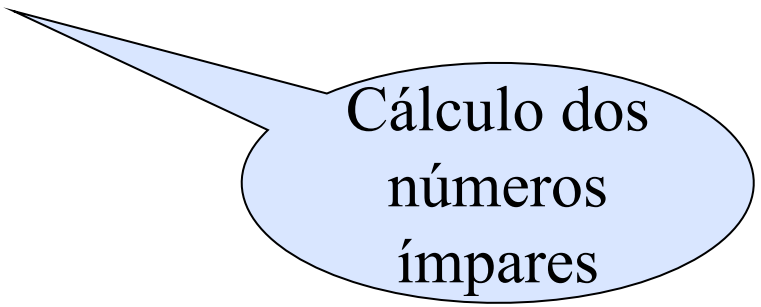
Leitura dos números

Cálculo dos números pares

```
printf(" Os numeros pares são: \n");
for(i=0; i<MAX; i++)
    if (num[i] % 2 == 0) // eh par
    {
        qpar++;
        printf("%d\n", num[i]);
    }
printf("O total de numeros pares eh: %d\n", qpar);
```

```
printf(" Os numeros impares são: \n");
for(i=0; i<MAX; i++)
    if (num[i] % 2 != 0) // eh impar
    {
        qimpar++;
        printf("%d\n", num[i]);
    }
printf("O total de numeros impares eh: %d\n", qimpar);

  system("PAUSE");
  return 0;
}
```



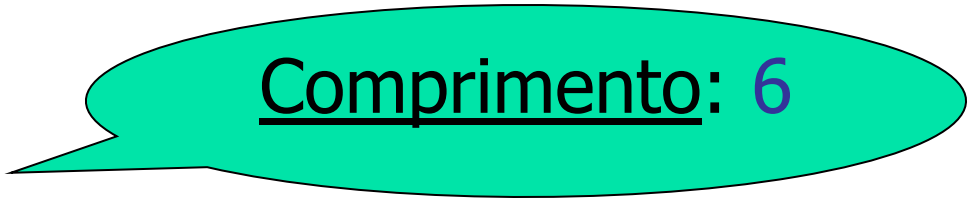
Cadeia de Caracteres

- CARACTER: letras, dígitos e símbolos
  - Exemplo: ‘a’, ‘%’, ‘2’
- CADEIA DE CARACTERES: um conjunto de caracteres
  - Exemplo: “A B3*g”, “1234”

Cadeia de Caracteres

- COMPRIMENTO DA CADEIA: número de caracteres que formam a cadeia

- Exemplo: "A B3*g"



Declaração

- Cadeia de caracteres ou strings são vetores:

```
char nome[20], alunos[40][20];
char B;
```

Manipulação

- É possível acessar uma posição da string:

```
char nome[20] = "JOAO";

printf("%c", nome[0]);

printf("%s", nome);
```

Manipulação

- Leitura de uma string:

```
char nome[20];

gets(nome);


printf("%s", nome);
```

Lê uma string e acrescenta a marca de fim de string \0

Manipulação

- Atribuição de uma string:

```
char nome[20];

strcpy(nome, "JOAO");

printf("%s", nome);
```

A ATRIBUIÇÃO DE STRINGS EM C USA A FUNÇÃO **STRCPY().** SIMILAR A:
Nome = "JOAO"

inserir biblioteca:
string.h

Expressões com Strings

- Os operadores relacionais podem ser usados com operandos do tipo caracter
- Operadores Relacionais

Para efeito de comparação entre os caracteres, toma-se como base a seqüência comparativa do código ASCII

==
<
>
!=
<=
>=

Expressões com Strings

a) Entre as letras vale a ordem alfabética

‘C’ < ‘D ‘

“ABACATE” < “ABACAXI”

b) Para os dígitos vale a ordem numérica

‘1’ < ‘3 ‘

Expressões com Strings

c) O branco sempre é menor que qualquer letra ou dígito

d) Os dígitos são menores que as letras

‘9’ < ‘a‘

e) Letras maiúsculas são menores que letras minúsculas

‘M’ < ‘a‘

Expressões com Strings

- Para comparar duas strings em C:

```
char n1[20], n2[20];

strcpy(n1, "ANA");
strcpy(n2, "ANAMARIA");

x = strcmp(n1, n2);
```

inserir biblioteca:
string.h

Se n1 < n2 → x recebe valor < 0
Se n1 > n2 → x recebe valor > 0
Se n1 == n2 → x recebe valor == 0

string-compara.c

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

int main(int argc, char *argv[])
{
  char str1[20], str2[20];
  int x;

  printf("****** Programa que compara duas strings de entrada******\n\n");

  printf("entre com a 1a string:");
  gets(str1);

  printf("entre com a 2a string:");
  gets(str2);

  x = strcmp(str1,str2);

  if (x < 0)
     printf("'%s' eh menor que '%s'\n", str1, str2);
  else if (x > 0)
     printf("'%s' eh maior que '%s'\n", str1, str2);
  else
     printf("'%s' e '%s' sao iguais\n", str1, str2);

  system("PAUSE");
  return 0;
}
```

string-compara2.c

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

int main(int argc, char *argv[])
{
  char str1[20], str2[20];
  int x;

  printf("****** Programa que compara duas strings de entrada******\n\n");

  printf("entre com a 1a string:");
  gets(str1);

  printf("entre com a 2a string:");
  gets(str2);


  printf("primeira comparacao - diferencia maiusculas de minusculas\n");
  x = strcmp(str1,str2);

  if (x < 0)
     printf("'%s' eh menor que '%s'\n", str1, str2);
  else if (x > 0)
     printf("'%s' eh maior que '%s'\n", str1, str2);
  else
     printf("'%s' e '%s' sao iguais\n", str1, str2);
  getch();
  printf("\n\n\segunda comparacao - NAO diferencia maiusculas de minusculas\n");
  x = stricmp(str1,str2);
  if (x < 0)
     printf("'%s' eh menor que '%s'\n", str1, str2);
  else if (x > 0)
     printf("'%s' eh maior que '%s'\n", str1, str2);
  else
     printf("'%s' e '%s' sao iguais\n", str1, str2);


  system("PAUSE");
  return 0;
}
```

string-converte-letras.c

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

int main(int argc, char *argv[])
{
  char str[20], str2[20];
  int i;

  printf("****** Programa que converte letras em uma string de entrada******\n\n");

  printf("entre com a string:");
  gets(str);

  //converte para maiusculo
  for(i=0 ; str[i]!= '\0'; i++)
          str2[i] = toupper(str[i]);
  str2[i] = '\0';

  printf("\n\n string convertida  para maiusculo = %s\n\n", str2);

  //converte para minusculo
  for(i=0 ; str[i]!= '\0'; i++)
          str2[i] = tolower(str[i]);
  str2[i] = '\0';

  printf("\n\n string convertida para minusculo = %s\n\n", str2);


system("PAUSE");
return 0;

}

```

Concatenação de Strings

- Permite concatenar (juntar) duas strings em uma só.
  - **strcat(s1, s2)**

```
char s1[20], s2[20];

strcpy(s1, "ANA");
strcpy(s2, "MARIA");

strcat(s1, s2);
```

s1 = "ANAMARIA"

Exercícios

- Faça um programa que lê uma frase, calcula e mostra a quantidade de palavras da frase.

```
int main(){
 char frase[100];
 int i=0,count=0;
 fgets(frase,100,stdin);
 while(i< strlen(frase)){
   while (isspace(frase[i++]));
    ++count;
   while (!isspace(frase[i++]));
 }
 printf("%d",count);
 return 0;
}
```

Exercícios propostos

1. Escrever um algoritmo que lê dois vetores de 10 elementos inteiros e multiplica os elementos de mesmo índice, colocando o resultado em um terceiro vetor. No final, mostrar os dois vetores lidos e o vetor resultante.
2. Faça um algoritmo que lê um vetor de 30 números inteiros e um número ***n*** a ser procurado no vetor. Escrever quantas vezes ***n*** aparece no vetor e em quais posições.
3. Desenvolva uma solução (pode ser somente os passos) para ordenar um vetor de 100 números.