Arranjos

Roberto Rocha

PUC Minas Virtual

Exercícios de fixação

1 - Elaborar um programa que leia um vetor A com 15 elementos inteiros. Construir um vetor B do mesmo tipo, em que cada elemento de B deva ser o resultado do fatorial correspondente de cada elemento da matriz A. Apresentar A e B.

2 - Construir um programa que leia dois vetores A e B com 10 elementos quaisquer inteiros. Construir um vetor C, sendo este o resultado da união dos elementos de A e B – sem repetição. Apresentar C.

3 - Elaborar um programa que leia 20 elementos do tipo real em um vetor A, em seguida crie um procedimento que inverta os elementos armazenados. Ou seja, o primeiro elemento de A passará a ser o ultimo, o segundo elemento passará a ser o penúltimo e assim por diante. Apresentar A.

4 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Crie um vetor ParImpar de 2 posições e armazene no índice 0 quantos elementos de A são par e no índice 1 quantos elementos de A são ímpar. Apresentar o vetor ParImpar. Obs.: não utilize o comando condicional se.

5 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Ordene e imprima o vetor A.

Exercícios de fixação

5 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Ordene e imprima o vetor A.

Uma das operações mais importantes executadas com arranjos é, sem duvida, a organização dos dados armazenados.

Uma das formas de organizar dados é proceder à classificação (ordenação) nas ordens numérica, alfabética ou alfanumérica.

A classificação numérica de dados pode ser efetuada na ordem crescente (do menor valor numérico para o maior) ou na ordem decrescente (do maior valor numérico para o menor).

A classificação alfabética de dados pode ser efetuada na ordem ascendente (de "A" até "Z" ou de "a" até "z") ou na ordem descendente (de "Z" até "A" ou de "z" até "a").

Como o computador sabe que o “A” vem antes do que o “B”?

A tabela ASCII

A representação dos 256 caracteres segue à estrutura de uma tabela de caracteres oficial chamada ASCII (American Standard Code for Information Interchange - Código Americano Padrão para Intercambio de Informações)

$2^8=256$

A tabela ASCII foi desenvolvida entre os anos de 1963 e 1968, com o objetivo de substituir o até então utilizado código de Baudot, o qual usava apenas cinco bits e possibilitava a obtenção de 32 combinações diferentes.

$2^5=32$

Utilizado na rede TELEX, operacionalizada pelos CORREIOS para envio de telegramas.
Esse código utilizava apenas os símbolos numéricos e letras maiúsculas.

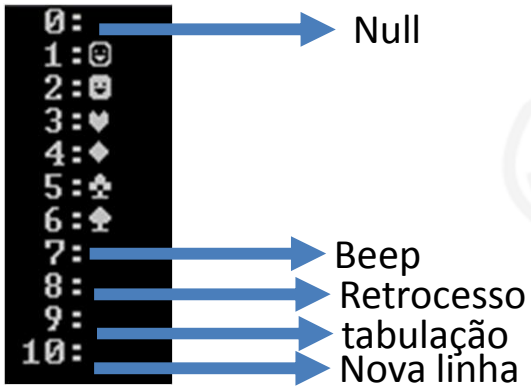
A tabela ASCII

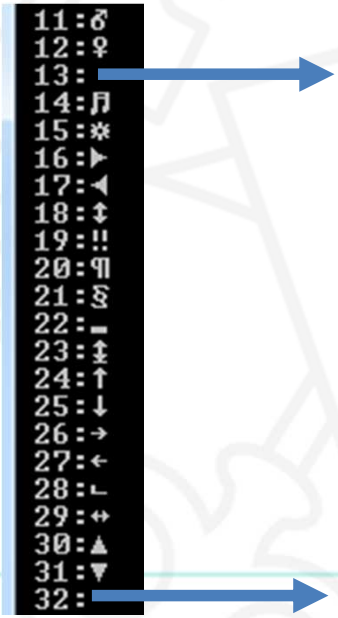
O código ASCII padrão permite a utilização de 128 símbolos diferentes (representados pelos códigos decimais de 0 até 127).

Nesse conjunto de símbolos estão previstos o uso de 96 caracteres imprimíveis, como números, letras minúsculas, letras maiúsculas, símbolos de pontuação e caracteres não imprimíveis, como retorno do carro de impressão (carriege return - tecla <Enter>), retrocesso (backspace), salto de linha (line feed), entre outros.

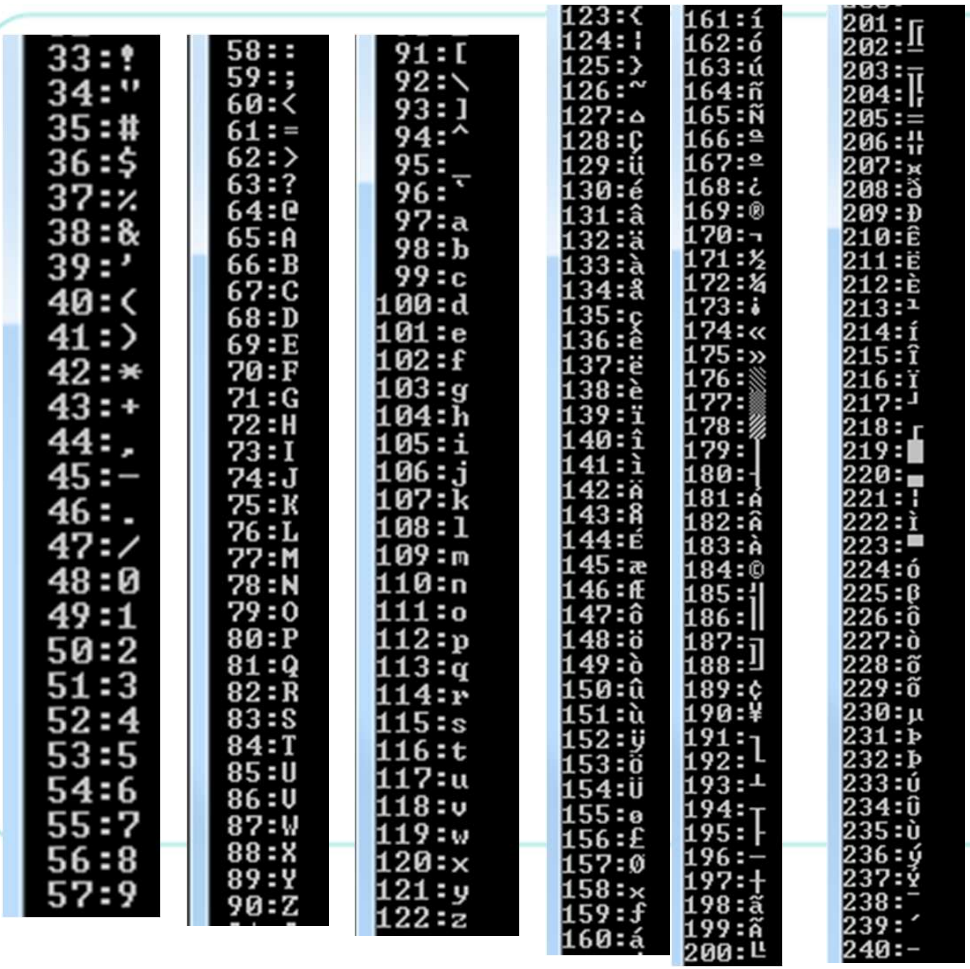
O programa a seguir irá imprimir a tabela ASCII.

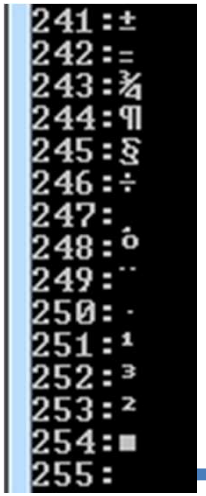
```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int i;
    for (i=0;i<256;i=i+1)
    {
        printf("%3d:%c\n",i,i);
    }
    return 0;
}
```





A tabela ASCII





Reservado

Da possibilidade de uso dos 256 caracteres, faz-se uso padronizado apenas dos 128 primeiros, o que deixa um espaço para outros 128 caracteres estendidos endereçados pelos códigos decimais de 128 até 255. A parte da tabela endereçada de 128 até 255 é reservada para que os fabricantes de computadores e de programas de computador possam definir seus próprios símbolos.

Utilizando a tabela, pode-se então verificar que o A (65) é menor do que o B(66) e assim por diante. Veja que em uma classificação o A(65) é menor do que o a(97)!

Classificação de dados - Métodos

Para classificar dados em um arranjo de uma dimensão (ou mesmo arranjos com mais dimensões) não há necessidade de um programador desenvolver algoritmos próprios, pois já existe um conjunto de algoritmos para essa finalidade. Basta conhecer e escolher aquele que atenda mais adequadamente a uma necessidade especifica. Entre as técnicas (os métodos) de programação para classificação de dados existentes, pode-se destacar as categorias (Manzano, 2016)*:

- Classificação por inserção (método da inserção direta, método da inserção direta com busca binaria, método dos incrementos decrescentes - shellsort).
- Classificação por troca (método da bolha - bubblesort, método da agitação - shakesort, método do pente - combsort, método de partição e troca - quicksort).
- Classificação por seleção (método da seleção direta, método da seleção em arvore - heapsort, método de seleção em arvore amarrada - threadedheapsort).
- Classificação por distribuição de chaves (método de indexação direta - radixsort).
- Classificação por intercalação (método da intercalação simples - mergesort, método de intercalação de sequencias naturais).
- Classificação por calculo de endereços (método das listas de colisão, método da solução postergada das colisões), dentre outros.

MANZANO, José Augusto N. G; OLIVEIRA, Jayr Figueiredo de. **Algoritmos :** lógica para desenvolvimento de programação de computadores. 28. ed. rev. e atual. São Paulo, SP: Érica, 2016. 1 recurso online ISBN 9788536518657 – pag:149

Classificação de dados - Métodos

Dos algoritmos de classificação de dados existentes, o algoritmo que vamos apresentar encontra-se na categoria classificação por troca sendo um método muito simples de classificação de dados.

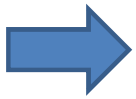
Apesar de eficaz, sua eficiência de velocidade é de certa forma questionável, pois é um método de ordenação lento, sendo útil para ordenar um conjunto pequeno de dados.

No entanto, é um método de fácil entendimento e serve de base para entender a maioria dos métodos existentes.

Imagine a necessidade de colocar em ordem crescente cinco valores numéricos inteiros, representados na tabela de valores.

Inicio

Valores	
0	7
1	4
2	3
3	6
4	2



Fim

Valores	
0	2
1	3
2	4
3	6
4	7

Classificação de dados - Métodos

Inicio

Valores	
0	7
1	4
2	3
3	6
4	2

O 1º elemento deve ser comparado com os demais elementos.

De modo que ao final do primeiro ciclo o 1º elemento contenha o menor valor do conjunto

1º passo

Valores	
0	7
1	4
2	3
3	6
4	2

Se o elemento que estiver na 1ª posição do vetor for maior troca

```
para i de 1 ate 4 passo 1
faça se (Valores[0]>Valores[i])
        então aux ← Valores[0]
                Valores[0] ← Valores[i]
                Valores[i] ←aux
        fimse
fimpara
```

Classificação de dados - Métodos

1º passo
inicio

Valores	
0	7
1	4
2	3
3	6
4	2

Se o elemento que estiver na 1ª posição do vetor for maior troca

```
para i de 1 ate 4 passo 1
faça se (Valores[0]>Valores[i])
    então aux ← Valores[0]
        Valores[0] ← Valores[i]
        Valores[i] ←aux
    fimse
fimpara
```

1º passo
inicio

Valores		
0	7	4
1	4	7
2	3	
3	6	
4	2	

aux	7
i	1

Classificação de dados - Métodos

1º passo
inicio

Valores	
0	7
1	4
2	3
3	6
4	2

Se o elemento que estiver na 1ª posição do vetor for maior troca

```
para i de 1 ate 4 passo 1
faça se (Valores[0]>Valores[i])
    então aux ← Valores[0]
        Valores[0] ← Valores[i]
        Valores[i] ←aux
    fimse
fimpara
```

1º passo
inicio

Valores		
0	4	3
1	7	
2	3	4
3	6	
4	2	

aux	7	4
i	1	2

Classificação de dados - Métodos

1º passo
inicio

Valores	
0	7
1	4
2	3
3	6
4	2

Se o elemento que estiver na 1ª posição do vetor for maior troca

```
para i de 1 ate 4 passo 1
faça se (Valores[0]>Valores[i])
        então aux ← Valores[0]
              Valores[0] ← Valores[i]
              Valores[i] ←aux
    fimse
fimpara
```

1º passo
inicio

Valores	
0	3
1	7
2	4
3	6
4	2

aux	4
i	2 3

Classificação de dados - Métodos

1º passo
inicio

Valores	
0	7
1	4
2	3
3	6
4	2

Se o elemento que estiver na 1ª posição do vetor for maior troca

para i de 1 ate 4 passo 1
faça se (Valores[0]>Valores[i])
então aux ← Valores[0]
Valores[0] ← Valores[i]
Valores[i] ←aux
fimse
fimpara

1º passo
inicio

Valores		
0	3	2
1	7	
2	4	
3	6	
4	2	3

aux	4	3
i	3	4

1º passo
Fim

Valores	
0	2
1	7
2	4
3	6
4	3

Classificação de dados - Métodos

1º passo
Fim

Valores	
0	2
1	7
2	4
3	6
4	3

Note que o 1º elemento após o primeiro passo contém o menor elemento do conjunto.

Precisamos agora repetir o processo para o 2º elemento de forma que ao final o 2º elemento possua o menor valor dentre os demais elementos do vetor.
Como se o vetor possuísse apenas 4 elementos!

Classificação de dados - Métodos

2º passo
inicio

Valores	
0	2
1	7
2	4
3	6
4	3

Se o elemento que estiver na 2ª posição do vetor for maior troca

```
para i de 2 ate 4 passo 1
faça se (Valores[1]>Valores[i])
    então aux ← Valores[1]
        Valores[1] ← Valores[i]
        Valores[i] ←aux
    fimse
fimpara
```

1º passo
inicio

Valores	
0	2
1	7
2	4
3	6
4	3

aux
i

4 3
7
4
7 4
2 3 4

2º passo
Fim

Valores	
0	2
1	3
2	7
3	6
4	4

Classificação de dados - Métodos

2º passo
Fim

Valores	
0	2
1	3
2	7
3	6
4	4

O 2º elemento após o segundo passo contém o menor elemento do conjunto restante.

Precisamos agora repetir o processo para o 3º elemento de forma que ao final o 3º elemento possua o menor valor dentre os demais elementos do vetor.
Como se o vetor possuísse apenas 3 elementos!

Classificação de dados - Métodos

3º passo
inicio

Valores	
0	2
1	3
2	7
3	6
4	4

Se o elemento que estiver na 3ª posição do vetor for maior troca

para i de 3 ate 4 passo 1
faça se (Valores[2]>Valores[i])
então aux ← Valores[2]
Valores[2] ← Valores[i]
Valores[i] ←aux
fimse
fimpara

1º passo
inicio

Valores	
0	2
1	3
2	7
3	6
4	4

aux
i

6 4
7
6
7 6
3 4

3º passo
Fim

Valores	
0	2
1	3
2	4
3	7
4	6

Classificação de dados - Métodos

3º passo
Fim

Valores	
0	2
1	3
2	4
3	7
4	6

Agora o 3º elemento contém o menor elemento do conjunto restante.

} Precisamos agora repetir o processo para o 4º elemento de forma que ao final o 4º elemento possua o menor valor dentre os demais elementos do vetor.
Como se o vetor possuísse apenas 2 elementos!

Classificação de dados - Métodos

4º passo
inicio

Valores	
0	2
1	3
2	4
3	7
4	6

Se o elemento que estiver na 4ª posição do vetor for maior troca

```
para i de 4 ate 4 passo 1
faça se (Valores[3]>Valores[i])
    então aux ← Valores[3]
        Valores[3] ← Valores[i]
        Valores[i] ←aux
    fimse
fimpara
}
```

1º passo
inicio

Valores	
0	2
1	3
2	4
3	7
4	6

6
7

aux	7
i	4

4º passo
Fim

Valores	
0	2
1	3
2	4
3	6
4	7

Vetor ordenado!!!

Exercícios de fixação

5 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Ordene e imprima o vetor A.

Procedimento para ler vetor

Procedimento classificar o vetor

Procedimento para imprimir vetor

5 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Ordene e imprima o vetor A.

Procedimento para ler vetor

Em C os vetores são sempre por referência

```
procedimento leVetor (var v:vetor[0..N-1] de inteiro,N:inteiro)
var
  i:inteiro
inicio
   para i de 0 ate N-1 passo 1
      faça  leia(v[i])
    fimpara
fimprocedimento
```

```
28  void leVetor (int *v, int tam)
29  {
30      int i;
31      for (i=0;i<tam; i=i+1)
32      {
33          printf("Digite o %d termo do vetor:", i);
34          scanf("%d",&v[i]);
35      }
36  }
```

5 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Ordene e imprima o vetor A.

Procedimento classificar o vetor

```
procedimento classificaVetor (v:vetor[0..tam-1] de inteiro,tam:inteiro
var
  i,j:inteiro
inicio
   para i de 0 ate tam-1 passo 1
     faça para j=i ate tam-1 passo 1
            faça se (v[i]>v[j]
                  então aux=v[i]
                        v[i]=v[j]
                        v[j]=aux
                fimse
          fímpara
  fimpara
fimprocedimento
```

```
41  void classificaVetor (int *v,int tam)
42  {
43      int i,j,aux;
44      for (i=0;i<tam;i=i+1)
45      {
46          for (j=i;j<tam;j=j+1)
47          {
48              if (v[i]>v[j])
49              {
50                  aux=v[i];
51                  v[i]=v[j];
52                  v[j]=aux;
53              }
54          }
55      }
56  }
```

5 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Ordene e imprima o vetor A.

Procedimento para imprimir vetor

```
procedimento imprimeVetor (v:vetor[0..N-1] de inteiro,N:inteiro)
var
  i:inteiro
inicio
   para i de 0 ate N-1 passo 1
     faça  escreva(v[i])
    fimpara
fimprocedimento
```

Em C os vetores são sempre por referência

```
37  void imprimeVetor(int *v, int tam, char *nomeVetor)
38  {
39      int i;
40      for (i=0;i<tam;i=i+1)
41      {
42          printf("%s[%d]=%d\n",nomeVetor,i,v[i]);
43      }
44  }
```

Mostrar o nome do vetor a ser impresso

5 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Ordene e imprima o vetor A.

.

```
Programa principal
var
A: vetor[0..9] de inteiro
tam,i: inteiro
Inicio
 tam← 10
  leVetor(A,tam)
  imprimeVetor(A,tam)
  classificaVetor(A,tam)
  imprimeVetor(A,tam)
fimalgoritmo
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
void leVetor (int *, int );
void imprimeVetor(int *, int,  char *);
void classificaVetor (int *,int );
/*
5 - Elaborar um programa que leia 10 elementos do
tipo inteiro em um vetor A. Ordene e imprima o vetor A.
*/
int main()
{
    setlocale(LC_ALL, "portuguese");
    int A[10],tam=10,i;
    leVetor(A,tam);
    printf("impressão antes da classificação \n");
    imprimeVetor(A,tam,"A");
    classificaVetor(A,tam);
    printf("impressão após a classificação \n");
    imprimeVetor(A,tam,"A");
    return 0;
}
```

5 - Elaborar um programa que leia 10 elementos do tipo inteiro em um vetor A. Ordene e imprima o vetor A.

.

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
void leVetor (int *, int );
void imprimeVetor(int *, int,  char *);
void classificaVetor (int *,int );
/*
5 - Elaborar um programa que leia 10 elementos do
tipo inteiro em um vetor A. Ordene e imprima o vetor A.
*/
int main()
{
    setlocale(LC_ALL, "portuguese");
    int A[10],tam=10,i;
    leVetor(A,tam);
    printf("impressão antes da classificação \n");
    imprimeVetor(A,tam,"A");
    classificaVetor(A,tam);
    printf("impressão após a classificação \n");
    imprimeVetor(A,tam,"A");
    return 0;
}
```

```
Digite o 0 termo do vetor:10
Digite o 1 termo do vetor:9
Digite o 2 termo do vetor:8
Digite o 3 termo do vetor:7
Digite o 4 termo do vetor:6
Digite o 5 termo do vetor:5
Digite o 6 termo do vetor:4
Digite o 7 termo do vetor:3
Digite o 8 termo do vetor:2
Digite o 9 termo do vetor:1
impressão antes da classificação
A[0]=10
A[1]=9
A[2]=8
A[3]=7
A[4]=6
A[5]=5
A[6]=4
A[7]=3
A[8]=2
A[9]=1
impressão após a classificação
A[0]=1
A[1]=2
A[2]=3
A[3]=4
A[4]=5
A[5]=6
A[6]=7
A[7]=8
A[8]=9
A[9]=10
```

PUC Minas
Virtual