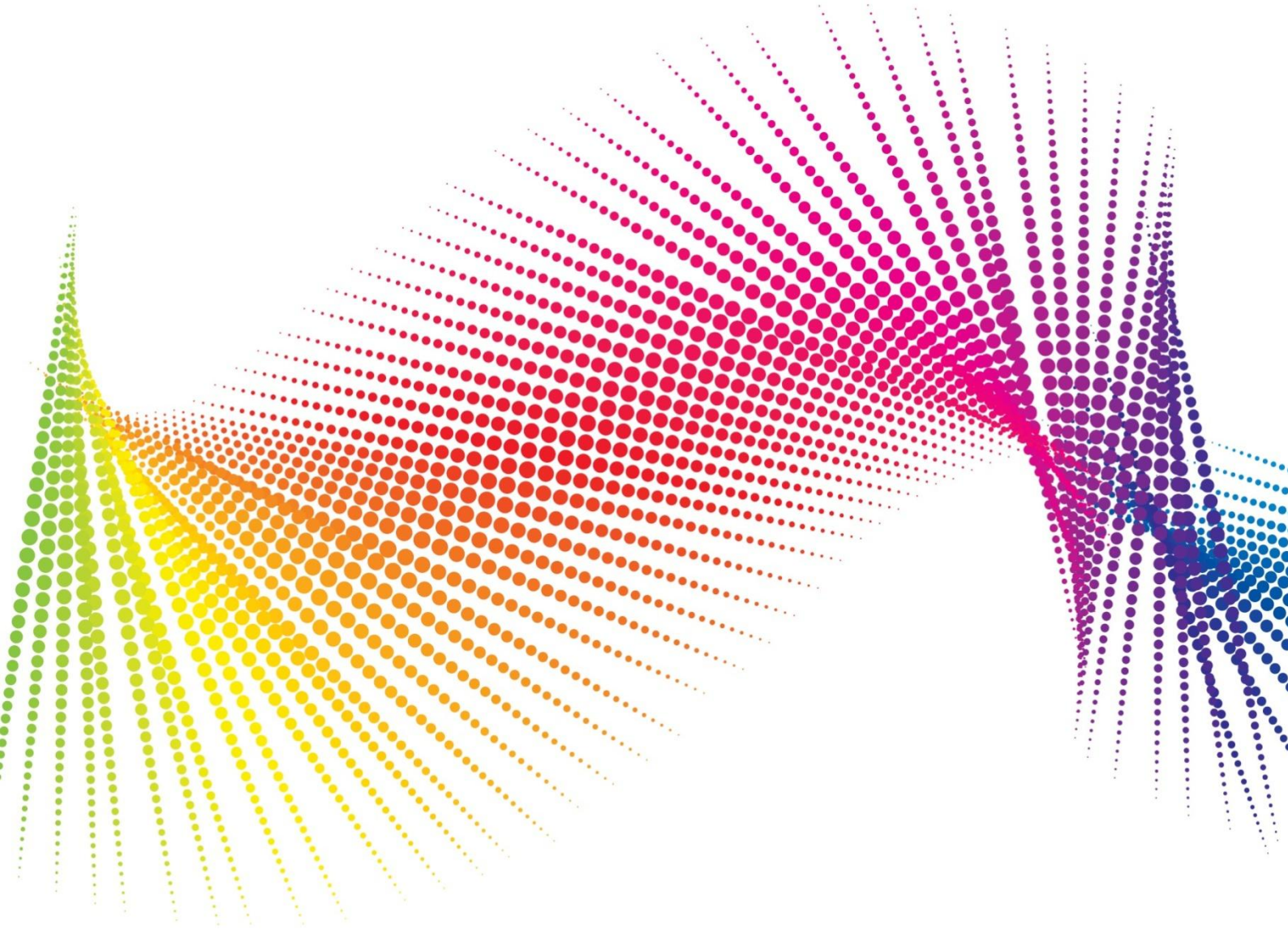
UNINOVE
Universidade Nove de Julho

Estruturas de Dados

Aula 11



Este material é parte integrante da disciplina oferecida pela UNINOVE.
O acesso às atividades, conteúdos multimídia e interativo, encontros virtuais, fóruns de discussão e a comunicação com o professor devem ser feitos diretamente no ambiente virtual de aprendizagem UNINOVE.

Uso consciente do papel.
Cause boa impressão, imprima menos.

Aula 11: Ordenação de vetores pelo método da Bolha

Objetivo: Nesta aula vamos estudar os métodos de ordenação de vetores, em particular, o método da Bolha (*Bubble Sort*).

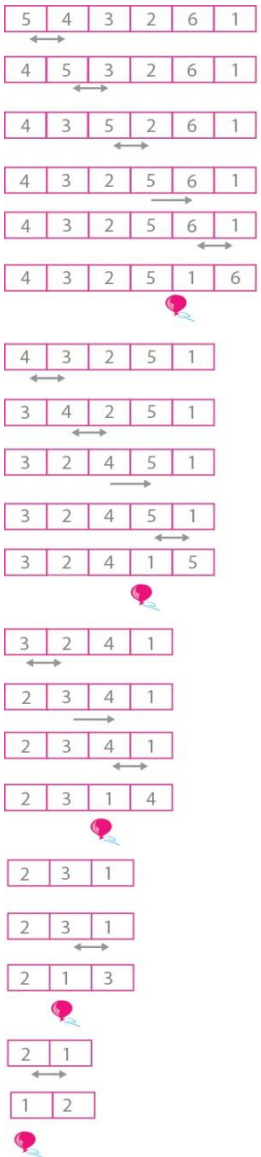
11.1. Métodos de ordenação de vetores

Caro aluno, antes de iniciar os estudos sobre o método da bolha, convém ler o material complementar anexo que lhe dará informações gerais sobre os métodos de ordenação e as maneiras de medir o esforço dos diferentes métodos de ordenação de vetores.

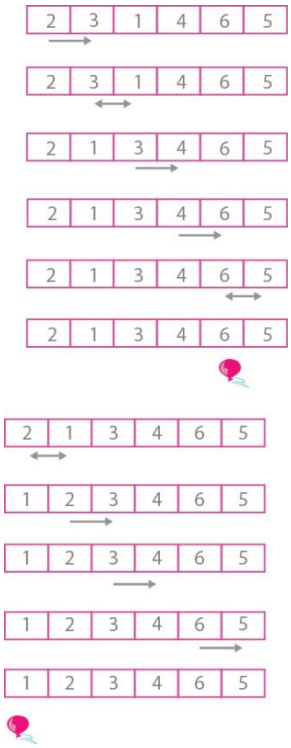
11.2. O método da Bolha (*bubble sort*)

O método da bolha - ***bubble sort***, ou ordenação por flutuação, é um algoritmo de ordenação de vetores dos mais simples. A ideia é percorrer o vetor diversas vezes e, a cada passagem, ir fazendo o maior elemento "flutuar" para o final do vetor, onde o maior elemento da sequência deve estar.

Neste método, percorre-se o vetor comparando-se os elementos dois a dois, começando-se das 2 primeiras posições. Se o elemento anterior for maior que seu posterior, trocam-se suas posições. No final do primeiro passo, o maior elemento do vetor é colocado em seu lugar, ou seja, na última posição. No próximo passo, percorre-se novamente o vetor, fazendo as comparações e as trocas, ignorando-se as posições finais já ordenadas. Os próximos passos repetem o mesmo processo até que o vetor esteja totalmente ordenado. Veja um exemplo de aplicação do método na figura ao lado.

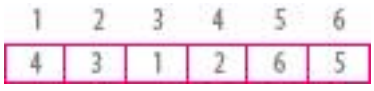


No método da bolha, existe a variável **bolha** que é usada para marcar a última posição onde ocorreu uma troca no passo J. No próximo passo, o vetor é percorrido da posição 1 até a posição marcada pela variável bolha. Isto pode conferir uma certa agilidade ao método, dependendo de como os elementos do vetor estejam posicionados inicialmente. Veja o comportamento do método com uma nova configuração inicial mostrada na figura abaixo. Em 2 passos o vetor já ficou ordenado!

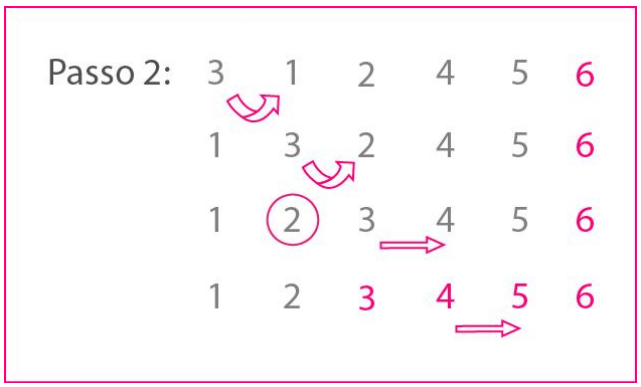


11.3. Exemplo passo a passo:

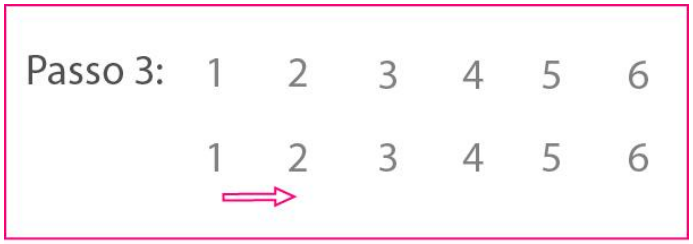
Seja o vetor inicial:



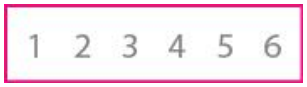
No **passo 1** as comparações começam na posição 1 e vão até o final do vetor. Cada vez que um elemento maior está antes de um elemento menor, eles são trocados. No final do passo 1 o maior elemento está na posição 6 do vetor. Foram feitas 5 comparações (conte o número de setas) e 4 trocas no passo 1 (cada seta curva representa uma troca). A variável bolha (círculo) marca a posição onde ocorreu a última troca no passo.



No **passo 2**, as comparações começam na posição 1 e vão até a posição marcada pela variável bolha no passo 1, que neste caso é a posição 5. Cada vez que um elemento maior está antes de um elemento menor, eles são trocados. No final do passo 2 todos os maiores elementos a partir da variável bolha estão nos seus lugares. Foram feitas 4 comparações (conte o número de setas) e 2 trocas no passo 2 (cada seta curva representa uma troca). A variável bolha (círculo) marca a posição onde ocorreu a última troca no passo.



No **passo 3**, as comparações começam na posição 1 e vão até a posição marcada pela variável bolha no passo 2, que neste caso é a posição 2. No final do passo 3 foi feita apenas 1 comparação e não ocorreu nenhuma troca, então a variável bolha ficou com um valor inicial igual a zero, indicando que o vetor já está ordenado e o método finaliza.



Resultado do método Bolha: Foram feitas 10 comparações e 6 trocas

11.4. Esforço do Método da Bolha

Consideremos que N corresponde ao número de elementos do vetor. O esforço do método contabiliza o número de comparações que ele realiza para ordenar o vetor.

No **melhor caso**, que é quando o vetor já está ordenado, o método Bolha realiza apenas (N – 1) trocas. Isso significa que o Bolha, no melhor caso, é O (N) ou possui complexidade linear.

O Bolha é O(N) no melhor caso.

No **caso médio**, em que os elementos do vetor estão em ordem aleatória, teríamos que realizar a análise do número médio de comparações, que é muito complicada e pode ser vista em [Knuth, “The art of computer programming”, vol.3]. O resultado é:

$(N^2 /2) - (3N / 4)$

Isso significa que o Bolha, no caso médio, é O (N^2) ou possui complexidade quadrática.

O Bolha é O(N^2) no caso médio.

Já no **pior caso**, o Bolha "empata" com o método de Seleção e realiza o seguinte número de comparações:

(N^2 – N)/ 2

Isso significa que o Bolha, no pior caso, é O (N^2) ou possui complexidade quadrática.

O Bolha é O(N^2) no pior caso.

Na análise final do método da Bolha devemos considerar o seu pior caso, então podemos dizer que este método de ordenação é de ordem de complexidade quadrática ou O (N^2), sendo N o número de elementos do vetor.

O Bolha pode ter no máximo N-1 passos. Em cada passo J, são realizadas da ordem de J-1 comparações, mas o número de passos pode variar, dependendo de como o vetor está inicializado.

A variável bolha armazena o valor da posição do vetor em que ocorreu a última troca no passo J. No próximo passo, o vetor é percorrido da posição 1 até a posição marcada pela variável Bolha. Caso Bolha esteja sinalizada com 1 ou 0, o método finaliza, pois o vetor já está ordenado.

11.5. Algoritmo do Método da Bolha

```
/* bolha.c: implementa o metodo da bolha - bubble sort- para
ordenacao de vetor */

#include <stdio.h>
#include <conio.h>

#define N 10

int vet[N+1];

int    aux,  bolha,  lsup, j, i;
int    cont=0;  // conta o numero de comparacoes
```

```
 int main()
 {
   printf("\nMetodo da Bolha");

   // Le os dados do vetor
   printf("\n\nForneca os elementos do vetor a ser ordenado");
   for (i=1; i<=N; i++)
   { printf("\nvet[%i]= ",i);
     scanf("%i",&vet[i]);
   }

    // este trecho implementa o metodo da bolha
     lsup = N;
     while (lsup > 1)
     {   bolha = 0;
         for (j=1;  j <= (lsup-1); j++)
           {   if (vet[j] > vet[j+1])
               {    aux = vet[j];
                    vet[j] = vet[j+1];
                    vet[j+1] = aux;
                    bolha = j;
               //bolha guarda a posicao onde ocorreu a ultima troca
               }
              ++cont;
         }
         lsup = bolha;
     }


  // mostra o vetor ordenado
  printf("\n Vetor ordenado pelo metodo Bolha: ");
  for (j=1; j<=N; j++)
      printf("\n vet[%i] = %i ", j,vet[j]);

  printf("\n Numero de comparacoes do BOLHA: %i ", cont);
  printf("\n\n\nFim do programa");
  getch();
  return 0;
}
```

Agora, caro aluno, vamos praticar resolvendo os exercícios propostos. **Leia a lista, resolva os exercícios e verifique seu conhecimento**. Caso fique alguma dúvida, leve a questão ao Fórum e divida com seus colegas e professor (a).

REFERÊNCIAS

KNUTH, Donald E. ***The Art of Computer Programming***. Addison-Wesley, 1990.

SCHILDT, H. **C Completo e Total**. São Paulo: Makron Books, 1997.

TENEMBAUM, Aaron M., et al. **Estruturas de Dados usando C**. São Paulo: Pearson Makron Books, 1995.