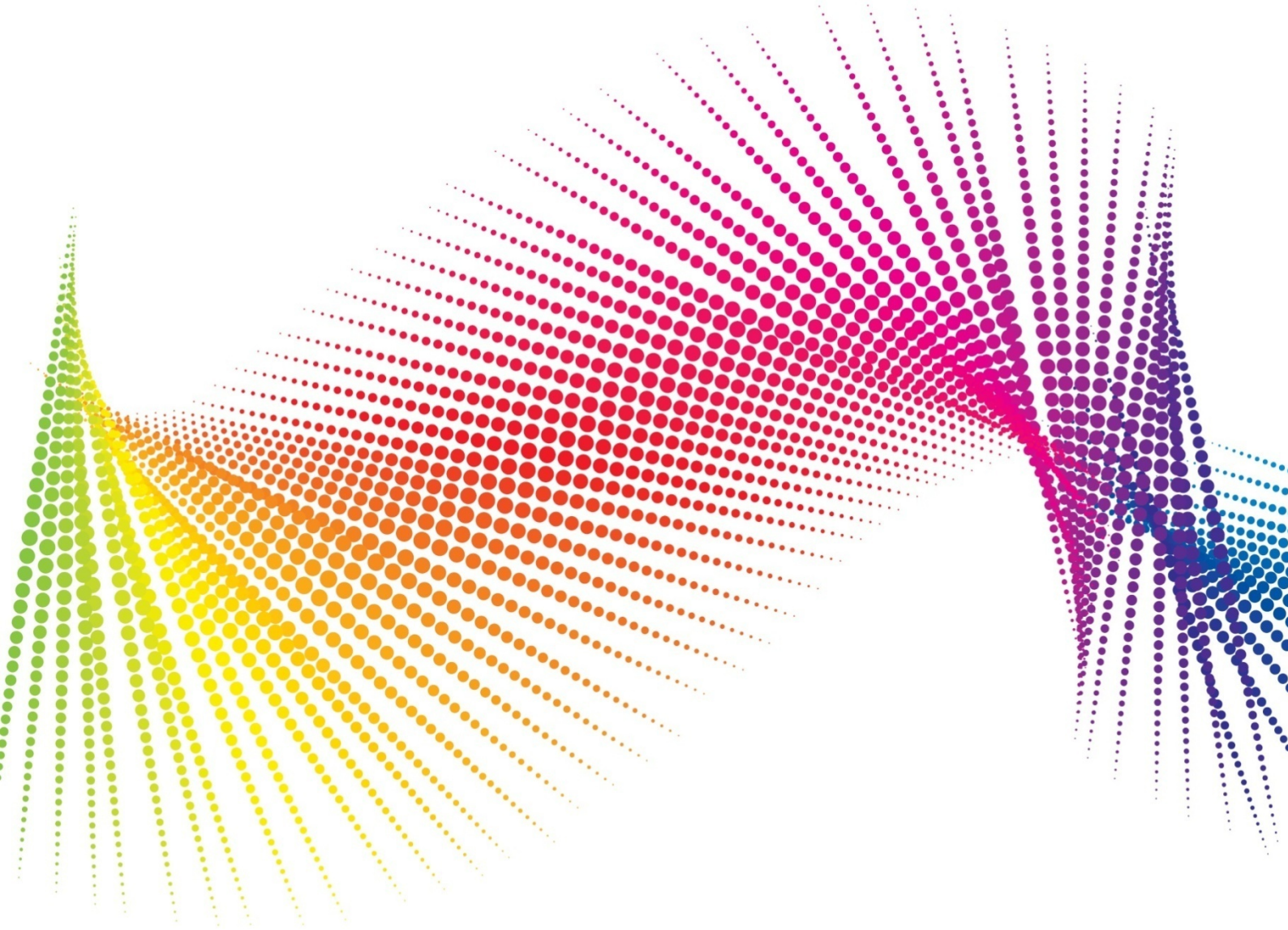
UNINOVE
Universidade Nove de Julho

Estrutura de Dados

Aula 12



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 12: Pesquisa em vetor – Pesquisa binária

Objetivo: Estudar a localização de elementos num vetor e conhecer os métodos de pesquisa sequencial e de pesquisa binária.

Definição de pesquisa ou busca

Buscar dados que foram armazenados no computador é algo obviamente necessário e existem técnicas para isto, porque, embora não pareça, até o computador vai demorar a buscar algo se o algoritmo não for eficiente.

Aqui vamos estudar 2 métodos simples de pesquisa bastante difundidos: **pesquisa sequencial e pesquisa binária**.

Método de pesquisa sequencial

Esta técnica consiste em comparar, sequencialmente, cada elemento (ou chave) do vetor com certo valor procurado até que este seja encontrado, ou seja, atingido o final do vetor, sem sucesso.

Esforço do método de pesquisa sequencial

A ordem de complexidade deste método é linear. O número de comparações para encontrar o elemento que está na posição J do vetor é J. O número máximo de comparações é N, ou seja, é o tamanho do vetor. No caso médio, realizam-se (N+1)/2 comparações.

Melhor caso: 1 comparação
Caso médio: (N+1)/2 comparações
Pior caso: N comparações

Algoritmo do método de pesquisa sequencial

```
//pesquisaseq.c: implementa o metodo de pesquisa sequencial em vetor

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

#define FALSO 0
#define VERDADEIRO 1
#define N 10

int vet[N+1];
int i, k, achou;

int main()
{
   printf("\nMetodo de Pesquisa Sequencial\n");
   printf("\nForneca os elementos do vetor a ser pesquisado \n");
   for (i=1; i<=N; i++)
   { printf("vet[%d]= ",i);
     scanf("%d",&vet[i]);
   }

   while (VERDADEIRO)
   {
     printf("\nChave a ser procurada  (digite -1 para terminar): ");
     scanf("%d",&k);

     if (k==-1)
       break;

     achou = FALSO;
     i = 1;
     while ( achou==FALSO && i<=N )
     { if (vet[i] == k)
       { printf("\nA chave %d esta na posicao %d do vetor \n",k,i);
         achou = VERDADEIRO;
       }
       else    ++i;
     }
     if (achou==FALSO)
          printf("\nA chave %d nao se encontra no vetor \n",k);

   }

   printf("\n\n\nFim do programa");
   getch();
   return 0;

}
```

Método de Pesquisa binária: vetores ordenados

Quando os elementos de um vetor estão previamente ordenados segundo algum critério (por exemplo, ordem crescente ou decrescente), então técnicas mais eficientes de pesquisa podem ser empregadas. Entre elas, destaca-se o método de **pesquisa binária.**

Seu funcionamento baseia-se no princípio de reduzir à metade, sucessivamente, o "universo de busca". Desse princípio resulta sua eficiência, o qual é conhecido como "dividir para conquistar".

Para se realizar a pesquisa binária, devem-se seguir os passos:

[1] Determinar o elemento que está no `meio` do vetor e compará-lo com o valor procurado (`K`). O elemento do `meio` está na posição `meio=(comeco+fim)/2.`

[2] Se o elemento central for igual a `K`, a pesquisa termina.

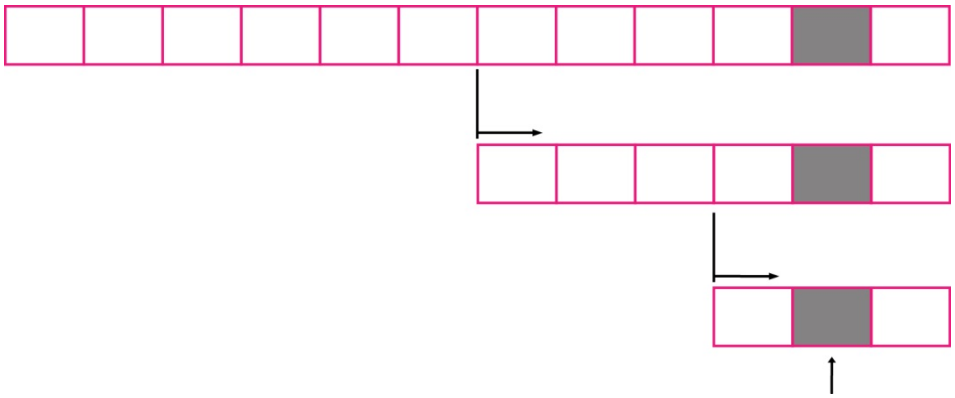
[3] Se o elemento central for menor que `K`, a pesquisa continuará na metade superior (a inferior será descartada), ou seja, o `fim` será mantido e o `comeco` ajustado para a posição `meio +1.`

[4] Se o elemento central for maior que `K`, continua-se a pesquisa somente na metade inferior do vetor, ou seja, o `comeco` será mantido e o `fim` ajustado para a posição `meio -1.`

[5] Se `comeco` ficar maior que `fim` significa que o elemento `K` não pertence ao vetor e a pesquisa termina. Caso contrário, volta-se ao passo `1` e o processo é repetido.

A pesquisa se encerrará em dois casos: ou quando a chave for encontrada, ou quando não houver mais nenhum componente do vetor a ser pesquisado. **A não localização do elemento procurado ocorre quando** `comeco` **fica maior que** `fim`**.**

O procedimento anteriormente descrito aplica-se a vetores classificados em ordem crescente. Para vetores em ordem decrescente, deve aplicar-se um raciocínio análogo.



Caro aluno, acesse o AVA e assista à animação para conhecer melhor o funcionamento do método de pesquisa binária. Esta animação faz parte da sequência desta aula e, portanto, é essencial para a sua aprendizagem.

Esforço do método de pesquisa binária

A quantidade de comparações que o método de pesquisa binária realiza é aproximadamente igual ao número de vezes que N (número de elementos do vetor) pode ser dividido por 2 até resultar 1, isto é, **$\log_2 N$ mais 1.** A ordem de complexidade desse método é **$O(\log_2 N)$**, ou seja**,** logarítmica, o que significa que é muito eficiente.

Algoritmo do método de pesquisa binária

```
/* pesqbinaria.c: implementa pesquisa binaria num vetor ordenado */

#include <stdio.h>
#include <conio.h>
#define N 10

int vet[N+1];
int comeco, meio, fim, i, k;

void  OrdenaInsercao (int vet[N], int max);



int main()
{  printf ("\n\nMetodo de pesquisa binaria \n");

   printf("\nForneca os elementos do vetor a ser pesquisado \n");
   for (i=1; i<=N; i++)
```

```
      { printf("vet[%d]= ",i);
        scanf("%d",&vet[i]);
       }

   OrdenaInsercao(vet,N);
   printf("\n Vetor ordenado pelo metodo de Insercao: \n");
   for (i=1; i<=N; i++)
      printf("\nvet[%d]= %d \n",i,vet[i]);

      for ( ; ; )
      { printf("\nChave a ser procurada (digite -1 para terminar):");
        scanf("%d",&k);

        if (k==-1)
            break;

        comeco = 1;
        fim = N;
        meio = (comeco+fim)/2;
        while (vet[meio] != k && comeco < fim)
        { if  (k < vet[meio])
                  fim = meio -1;
            else comeco = meio +1;
            meio =  (comeco+fim)/2;
        }
        if (vet[meio] == k)
        printf("\nA chave %d encontra-se na posicao %d do vetor \n",
                                    k, meio);
        else printf("\nA chave %d nao se encontra no vetor\n",k);
      }
   printf ("\n\nFim do metodo de busca binaria ");
   getch();
   return 0;
 }

void OrdenaInsercao(int vet[N], int max)
{      int aux, j, i;

       for (j=2; j<=max; j++)
      {    aux  = vet[j];
           vet[0] = aux;
           i = j-1;
           while (aux < vet[i])
           {  vet[i+1] = vet[i];
               --i;
           }
           vet[i+1] = aux;
       }
}
```

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.

REFERÊNCIAS

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.