Universidade Braz Cubas

Projeto e Análise de Algoritmos

Carlos Alberto Gonçalves Abreu – 126749 – 11º EN

Prof. Ângelo Pássaro

Introdução

A comparação dos tempos de execução de listas fica melhor demonstrada quando se aplica os conceitos na prática, podendo ter claramente a idéia de como a escolha do algoritmo pode influenciar toda a execução do aplicativo.

Resultados do Problema 4

A implementação dos aplicativos foi feita em Delphi® 3, gerado como aplicativo console. A escolha pelo Delphi® advém de problemas encontrados com o Turbo Pascal®, onde foi inicialmente implementados os algoritmos, mas devido à limitação de posições de memória imposta pelo Turbo Pascal® (32760).

O equipamento utilizado para testes foi um Intel Pentium® 200MMX com 64MB de RAM, e na implementação, foram utilizados o tipo longint e massa de dados variando de 5000 à 100000 posições de memória.

Os tempos obtidos estão em segundos, e estão demostrados através das tabelas abaixo.

Listas Ordenadas

	5000	10000	25000	50000	100000
Aleatório	1	2	11	48	107
Ordenado	0	2	13	54	120
Decrescent	0	1	9	44	99
Remoção	1	2	11	61	86

Tabela 1: Tempo em seg.

Listas Ordenadas

	5000	10000	25000	50000	100000
Aleatório	1	3	23	110	482
Ordenado	1	4	24	110	479
Decrescent	0	3	24	108	480
Remoção	1	7	52	237	629

Tabela 2: Tempo em seg.

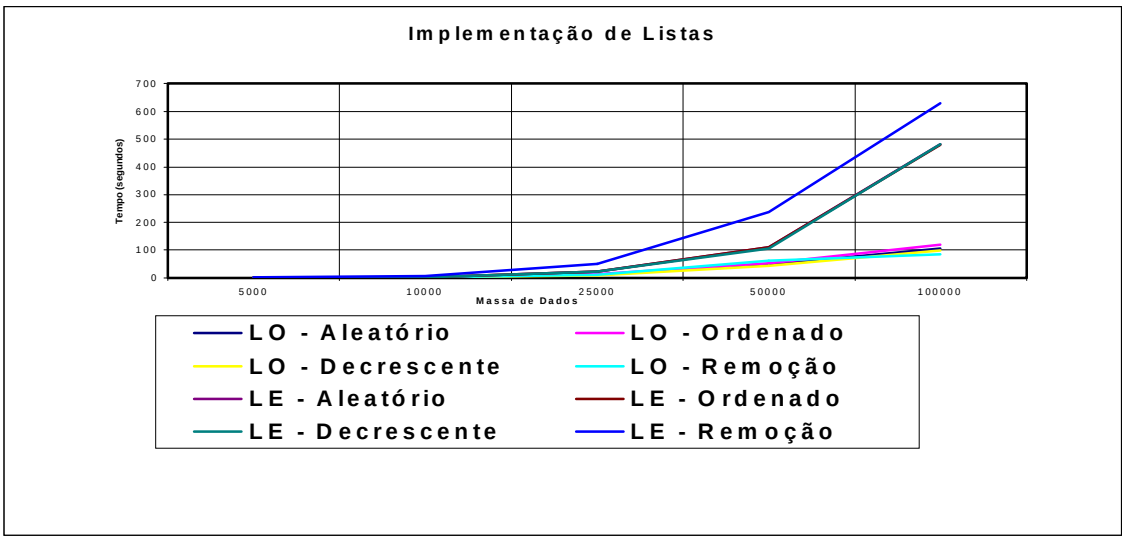

Fig. 1: Gráfico dos tempos obtidos.

Conclusão

Com a implementação podemos verificar o quão importante é a escolha do algoritmo, tendo como parâmetros a massa de dados e como ela se comporta diante da aplicação. Em casos reais, tais como um banco ou uma indústria, a eficiência e a estabilidade são itens indispensáveis, e como visto, uma escolha mal feita pode trazer sérias complicações para nós, profissionais de informática, que desejamos trabalhar em desenvolvimento de aplicativos.

Bibliografia

SEVERINO, Antônio Joaquim. Metodologia do Trabalho Científico. 21. ed. São Paulo: Cortez, 2000. 279p. Bibliografia: p. 86-132. ISBN 85-249-0050-4.

RUIZ, João Álvaro. Metodologia Científica. 4. ed. São Paulo: Atlas, 1996. 177p. Bibliografia: p. 74-86. ISBN 85-224-1465-3.

MARTINS, Gilberto de Andrade. Manual para Elaboração de Monografias e Dissertações. 2. ed. São Paulo: Atlas. 115p. Bibliografia: p. 49-62. ISBN 85-224-1087-9

Anexo 1 – Código-Fonte do Algoritmo

Este código está escrito em object-pascal do Delphi.

```
{$APPTYPE CONSOLE}

program exe4lo;

uses
  Windows,comctrls,SysUtils;
{
Bibliografia
  Pereira, Silvio do Lago - "Estruturas de Dados Fundamentais",
    SÆo Paulo: □rica,1996, pag. 109-117.
  Aho, Alfred V., Hopcroft, John E., Ullman, Jeffrey D. - "Data Structures
    and Algorithms", Massachussets: Addison-Wesley, pag. 37-43.

       Programa elaborado por Carlos Abreu 126749 11o EN

       Demonstra os tempos de inser‡Æo e remo‡Æo de items em
       Lista Ordenadas
}


{ Declara o tamanho maximo do vetor }
const
    MAXIMO = 100000000; //100 milhões (10^8)

type
   tipo = longint;

   { Cria tipo Lista Ordenada }
   LO = record
     info:array [1..MAXIMO] of tipo;
     ultimo:word;
   end;

   { Cria o Tipo Para estatistica }
   St = record
     S:string[3];
   end;
var
  key:char;
  ListOrd:LO;
  MAX:longint;


{ Verifica se a lista está  completa, se sim, retorna verdadeiro }
function Cheia(var Lista:LO):boolean;
begin
    if lista.ultimo>=MAX then
       Cheia:=true
    else
        cheia:=false;
end;

{ Verifica se a lista está  vazia, se sim, retorna verdadeiro }
function Vazia(Var Lista:LO):boolean;
begin
    if lista.ultimo<=0 then
       vazia:=true
    else
        vazia:=false;
```

```
end;

{ Busca Elemento em uma Lista retornando a posição relativa do elemento }
function busca(Var Lista:LO;Elemento:Tipo):tipo;
var
  i:longint;
begin
    if vazia(Lista) then
      busca:=0
    else
       for i:=1 to lista.ultimo do
          if lista.info[i]>=Elemento then
            begin
               busca:=i;
               exit;
            end
          else
             busca:=0;
end;

{ Insere um elemento a Lista, ordenando os valores de forma crescente }
procedure inserir(Var Lista:LO;Elemento:tipo);
var
  pos,i:longint;
begin
    if cheia(Lista) then
      writeln('Lista Cheia')
    else
       begin
          pos:=busca(Lista,Elemento);
          if pos=0 then
            pos:=lista.ultimo+1
          else
             for i:=Lista.ultimo+1 downto pos do
                lista.info[i]:=lista.info[i-1];
       end;
    Lista.info[pos]:=elemento;
    lista.ultimo:=lista.ultimo+1;
end;

{ Remove um Elemento da Lista, se o elemento não existir, exibe uma mensagem }
procedure remover(var Lista:LO;Elemento:tipo);
var
  pos,i:longint;
begin
   if Vazia(Lista) then
      exit {writeln('Lista Vazia');}
   else
      begin
         pos:=busca(Lista,Elemento);
         if pos=0 then
           exit {writeln('Elemento nÆo encontrado')}
         else
            begin
               for i:=pos to lista.ultimo-1 do
                  lista.info[i]:=lista.info[i+1];
               lista.ultimo:=lista.ultimo-1;
            end;
      end;
end;

{ Inicia a lista com valores nulos(0) }
procedure LimpaLista(Var Lista:LO);
var
  i:longint;
begin
```

```
    for i:=1 to MAX do
       lista.info[i]:=0;
    lista.ultimo:=0;
end;

{ Estatistica }
procedure stat;
var
  S:array[1..4] of ST;
  v,m1,m2,s1,s2:string;
  min,sec:word;
  x,y:longint;
begin
    randomize;
    repeat
        Write('Digite o tamanho da lista, com o minimo de 1 e no m ximo de 100000000 :');
        readln(v);
        Val(v,MAX,y);
    until (MAX<=100000000) and (MAX>=1);

    writeln('Criando Estatistica, pode demorar algum tempo...');

{ Lista Ordenada }
{ Insere Aleatório }
    LimpaLista(ListOrd);
    writeln('Inserindo dados aleatóriamente');
    s1:=timetostr(time);
    for y:=1 to Max do
       Inserir(ListOrd,Random(MAXIMO));
    s2:=timetostr(time);
    m1:=copy(s1,4,2);
    m2:=copy(s2,4,2);
    s1:=copy(s1,7,2);
    s2:=copy(s2,7,2);
    min:=strtoint(m2)-strtoint(m1);
    sec:=strtoint(s2)-strtoint(s1);
    sec:=sec+(min*60);
    S[1].s:=inttostr(sec);

{ Insere Ordenado }
    LimpaLista(ListOrd);
    writeln('Inserindo dados Ordenadamente');
    s1:=timetostr(time);
    for y:=1 to Max do
       Inserir(ListOrd,y);
    s2:=timetostr(time);
    m1:=copy(s1,4,2);
    m2:=copy(s2,4,2);
    s1:=copy(s1,7,2);
    s2:=copy(s2,7,2);
    min:=strtoint(m2)-strtoint(m1);
    sec:=strtoint(s2)-strtoint(s1);
    sec:=sec+(min*60);
    S[2].s:=inttostr(sec);

{ Insere Decrescente }
    LimpaLista(ListOrd);
    x:=MAX;
    writeln('Inserindo dados em ordem decrescente');
    s1:=timetostr(time);
    for y:=1 to Max do
       begin
           Inserir(ListOrd,x);
           x:=x-1;
       end;
    s2:=timetostr(time);
```

```
    m1:=copy(s1,4,2);
    m2:=copy(s2,4,2);
    s1:=copy(s1,7,2);
    s2:=copy(s2,7,2);
    min:=strtoint(m2)-strtoint(m1);
    sec:=strtoint(s2)-strtoint(s1);
    sec:=sec+(min*60);
    S[3].s:=inttostr(sec);

{ Remoção }
    s1:=timetostr(time);
    writeln('Removendo dados aleatóriamente');
    //for y:=1 to Max do
    while not Vazia(ListOrd) do
       Remover(ListOrd,Random(MAX));
    s2:=timetostr(time);
    m1:=copy(s1,4,2);
    m2:=copy(s2,4,2);
    s1:=copy(s1,7,2);
    s2:=copy(s2,7,2);
    min:=strtoint(m2)-strtoint(m1);
    sec:=strtoint(s2)-strtoint(s1);
    sec:=sec+(min*60);
    S[4].s:=inttostr(sec);
{ Fim Lista Ordenada }

{ Resultados }
    str(Max,v);
    writeln('Estat¡stica: Listas com '+v+' tamanho.');
              writeln('Os dados estÆo no formato: Segundos');
    writeln('');
    writeln('Inser‡Æo de Dados Aleat¢rios (Lista Ordenada):   '+s[1].s+' segundos');
    writeln('Inser‡Æo de Dados Ordenados (Lista Ordenada):    '+s[2].s+' segundos');
    writeln('Inser‡Æo de Dados Decrescente (Lista Ordenada):  '+s[3].s+' segundos');
    writeln('Remo‡Æo de Dados (Lista Ordenada):               '+s[4].s+' segundos');
    writeln('');
    write('Tecle [enter]');
    readln;
end;

begin

    repeat
        writeln('1 - Inicia a estat¡stica de Listas Ordenadas');
        writeln('0 - Encerrar Programa');
        readln(key);
        case key of
            '1':stat;

        end;
    until key='0';

end.
```

```
{$APPTYPE CONSOLE}
```

```
program exe4le;

uses
  Windows,comctrls,SysUtils;



{
Bibliografia
  Pereira, Silvio do Lago - "Estruturas de Dados Fundamentais",
     SÆo Paulo: □rica,1996, pag. 109-117.
  Aho, Alfred V., Hopcroft, John E., Ullman, Jeffrey D. - "Data Structures
     and Algorithms", Massachussets: Addison-Wesley, pag. 37-43.

        Programa elaborado por Carlos Abreu 126749 11o EN

        Demonstra os tempos de inser‡Æo e remo‡Æo de items em
        Lista Ordenadas
}

{ Declara o tamanho maximo do vetor }
const
     MAXIMO = 100000000;  //100 milhões (10^8);

type
    tipo = longint;

    { Cria o tipo Lista Encadeada }

     LEnc=^le;
     LE = record
       info: tipo;
       prox:LEnc;
    end;

    { Cria o Tipo Para estatistica }
    St = record
       S:string[3];
    end;
var
   key:char;
   ListEnc:Lenc;
   MAX:longint;

procedure Mostra(var L:Lenc);
var
   list:Lenc;
   st:string;
begin
     list:=L;
     while (list^.prox<>nil) do
          begin
               st:=inttostr(list^.info);
               write(st+' ');
               List:=List^.prox;
          end;
end;

{ Cria uma Lista Encadeada Vazia }
procedure Create(var L:Lenc);
begin
     L:=nil;
end;

{ Verifica se a Lista Encadeada est  vazia }
```

```
function Null(L:LEnc):boolean;
begin
    Null:=(L=nil);
end;

{ Insere Elemento  na Lista Encadeada }
procedure insert(var L:LEnc;Elemento:tipo);
var
  N,P:Lenc;
begin
    new(N);
    N^.info:=Elemento;
    if Null(L) {or (elemento<L^.info)} then
      begin
          N^.prox:=L;
          L:=N;
      end
    else
       begin
           P:=L;
           while (p^.prox<>nil) {and (Elemento>p^.prox^.info)} do
                p:=p^.prox;
           N^.prox:=P^.prox;
           P^.prox:=N;
       end;
end;

{ Remove o Elemento da Lista Encadeada }
function Del(var L:Lenc;Elemento:tipo):boolean;
var
  P,Q:Lenc;
begin
    if Null(L) {or (Elemento<L^.info)} then
      Del:=false
    else
       if Elemento=L^.info then
         begin
             P:=l;
             L:=L^.prox;
             dispose(p);
             del:=true;
         end
       else
          begin
              P:=l;
              while (p^.prox<>nil) {and (Elemento>p^.prox^.info)} do
                   P:=p^.prox;
              if (p^.prox<>nil) {and (elemento=p^.prox^.info)} then
                 begin
                     q:=p^.prox;
                     p^.prox:=q^.prox;
                     dispose(q);
                     del:=true;
                 end
              else
                  del:=false;
          end;
end;

{ Procura Elemento em Lista Encadeada }
function procura(var L:Lenc;Elemento:tipo):Lenc;
var
  p:lenc;
begin
    p:=l;
    while (p<>nil) and (elemento>p^.info) do
```

```
        P:=p^.prox;
    if (p<>nil) and (elemento=p^.info) then
      procura:=p
    else
       procura:=nil;
end;

{ Destroi a lista encadeada }
procedure kill(var L:Lenc);
var
  p:Lenc;
begin
    while L<>nil do
        begin
            P:=l;
            L:=L^.prox;
            dispose(p);
        end;
end;

{ Estatistica }
procedure stat;
var
  S:array[1..4] of ST;
  v,m1,m2,s1,s2:string;
  min,sec:word;
  x,y:longint;
begin
    randomize;
    repeat
        Write('Digite o tamanho da lista, com o minimo de 1 e no m ximo de 100000000 :');
        readln(v);
        Val(v,MAX,y);
    until (MAX<=100000000) and (MAX>=1);

    writeln('Criando Estatistica, pode demorar algum tempo...');

{ Lista Encadeada }
{ Insere Aleatório }
    Create(ListEnc);
    writeln('Inserindo dados aleatóriamente');
    s1:=timetostr(time);
    for y:=1 to Max do
       Insert(ListEnc,Random(MAXIMO));
    s2:=timetostr(time);

    Kill(ListEnc);
    m1:=copy(s1,4,2);
    m2:=copy(s2,4,2);
    s1:=copy(s1,7,2);
    s2:=copy(s2,7,2);
    min:=strtoint(m2)-strtoint(m1);
    sec:=strtoint(s2)-strtoint(s1);
    sec:=sec+(min*60);
    S[1].s:=inttostr(sec);

{ Insere Ordenado }
    Create(ListEnc);
    writeln('Inserindo dados ordenadamente');
    s1:=timetostr(time);
    for y:=1 to Max do
       Insert(ListEnc,y);
    s2:=timetostr(time);


    Kill(ListEnc);
```

```
    m1:=copy(s1,4,2);
    m2:=copy(s2,4,2);
    s1:=copy(s1,7,2);
    s2:=copy(s2,7,2);
    min:=strtoint(m2)-strtoint(m1);
    sec:=strtoint(s2)-strtoint(s1);
    sec:=sec+(min*60);
    S[2].s:=inttostr(sec);

{ Insere Decrescente }
    Create(ListEnc);
    x:=MAX;
    writeln('Inserindo dados em ordem decrescente');
    s1:=timetostr(time);
    for y:=1 to Max do
       begin
          Insert(ListEnc,x);
          x:=x-1;
       end;
    s2:=timetostr(time);

    //Kill(ListEnc);
    m1:=copy(s1,4,2);
    m2:=copy(s2,4,2);
    s1:=copy(s1,7,2);
    s2:=copy(s2,7,2);
    min:=strtoint(m2)-strtoint(m1);
    sec:=strtoint(s2)-strtoint(s1);
    sec:=sec+(min*60);
    S[3].s:=inttostr(sec);

{ Remoção }
    //Create(ListEnc);
    writeln('Removendo dados aleatóriamente');
    s1:=timetostr(time);
    for y:=1 to Max do
    //while not Null(ListEnc) do
       Del(ListEnc,Random(MAX));
    s2:=timetostr(time);

    Kill(ListEnc);
    m1:=copy(s1,4,2);
    m2:=copy(s2,4,2);
    s1:=copy(s1,7,2);
    s2:=copy(s2,7,2);
    min:=strtoint(m2)-strtoint(m1);
    sec:=strtoint(s2)-strtoint(s1);
    sec:=sec+(min*60);
    S[4].s:=inttostr(sec);
{ Fim Lista Ordenada }

{ Resultados }
    str(Max,v);
    writeln('Estat¡stica: Listas com '+v+' tamanho.');
              writeln('Os dados estÆo no formato: Segundos');
    writeln('');
    writeln('Inser‡Æo de Dados Aleat¢rios (Lista Encadeada):   '+s[1].s+' segundos');
    writeln('Inser‡Æo de Dados Ordenados (Lista Encadeada):    '+s[2].s+' segundos');
    writeln('Inser‡Æo de Dados Decrescente (Lista Encadeada):  '+s[3].s+' segundos');
    writeln('Remo‡Æo de Dados (Lista Encadeada):               '+s[4].s+' segundos');
    writeln('');
    write('Tecle [enter]');
    readln;
end;

begin
```

```
    repeat
        writeln('1 - Inicia a estat¡stica de Listas Encadeadas');
        writeln('0 - Encerrar Programa');
        readln(key);
        case key of
            '1':stat;

        end;
    until key='0';

end.
```