Physikalisch-Technische Lehranstalt Wedel

Staatsexamen für Technische Assistenten SoSe2002
schriftliche Prüfung im Fach Prozesstechnik
gewählter Vorschlag

Dauer : 180 Minuten **keine externen Hilfsmittel**

Aufgabe :

Den einzelnen Programmen in der Prozessdatenverarbeitung werden zumeist Prozessnummern zugeordnet. Zur einfachen, sortierten und reorganisationsfreien temporären Speicherung derartiger Prozessnummern können binäre Bäume dienen. Entwickeln Sie daher auf Basis des 8086-Prozessors einen Prototypen einer Toolbox für binäre Bäume - das Assembler-Modul *BTOOLS.ASM* - durch möglichst bedeutungstreue Übersetzung der Pascal-Unit *BTOOLS.PAS*.

Vergessen Sie nicht die hinreichende Kommentierung Ihres Assembler-Programms z.B. durch Zuordnung der Pascal-Befehle zu den Assembler-Befehlen.

```
Unit BTools;

Interface

Type PKnoten = ^TKnoten;
     TKnoten = Record
                 Key    : Integer;
                 Links  : PKnoten;
                 Rechts : PKnoten
               End;

Function InsertKey(Var P:PKnoten;K:Integer):Boolean;
Function DeleteKey(Var P:PKnoten;K:Integer):Boolean;
Procedure ListKeys(P:PKnoten);
Procedure ListTree(P:PKnoten);

Implementation

Function InsertKey(Var P:PKnoten;K:Integer):Boolean;

Begin
  If P = Nil Then Begin
    P := New(PKnoten);
    P^.Key := K;
    P^.Links := Nil;
    P^.Rechts := Nil;
    InsertKey := True
  End Else
    If P^.Key = K Then
      InsertKey := False
    Else
      If P^.Key > K Then
        InsertKey := InsertKey(P^.Links,K)
      Else
        InsertKey := InsertKey(P^.Rechts,K)
End;
```

```
Function DeleteKey(Var P:PKnoten;K:Integer):Boolean;

Function _FindMax(P:PKnoten):PKnoten;

Begin
  If P^.Rechts = Nil Then
    _FindMax := P
  Else
    _FindMax := _FindMax(P^.Rechts)
End;

Function _FindMin(P:PKnoten):PKnoten;

Begin
  If P^.Links = Nil Then
    _FindMin := P
  Else
    _FindMin := _FindMin(P^.Links)
End;

Begin
  If P = Nil Then
    DeleteKey := False
  Else
    If P^.Key = K Then Begin
      If (P^.Links = Nil) And (P^.Rechts = Nil) Then Begin
        Dispose(P);
        P := Nil;
        DeleteKey := True
      End Else
        If P^.Links <> Nil Then Begin
          P^.Key := _Findmax(P^.Links)^.Key;
          DeleteKey := DeleteKey(P^.Links,P^.Key)
        End Else Begin
          P^.Key := _FindMin(P^.Rechts)^.Key;
          DeleteKey := DeleteKey(P^.Rechts,P^.Key)
        End
    End Else
      If P^.Key > K Then
        DeleteKey := DeleteKey(P^.Links,K)
      Else
        DeleteKey := DeleteKey(P^.Rechts,K)
End;


Procedure ListKeys(P:PKnoten);

Begin
  If P <> Nil Then Begin
    ListKeys(P^.Links);
    WriteLn(P^.Key);
    ListKeys(P^.Rechts)
  End
End;
```

```
Procedure ListTree(P:PKnoten);

Procedure _ListTree(P:PKnoten;S:Integer);

Begin
  If P <> Nil Then Begin
    WriteLn(P^.Key);
    If P^.Links <> Nil Then Begin
      Write(' ':S*2,'L:');
      _ListTree(P^.Links,S+1)
    End;
    If P^.Rechts <> Nil Then Begin
      Write(' ':S*2,'R:');
      _ListTree(P^.Rechts,S+1)
    End
  End
End;

Begin
  _ListTree(P,0)
End;

End.
```

Zur Realisierung der New-Funktion und der Dispose-Prozedur werden die DOS-Funktionen "Allocate Memory" (Funktionscode 48h) und "Free Memory" (Funktionscode 49h) verwendet. Die Funktion zum Reservieren von Speicherplatz erwartet im BX-Register die Anzahl der benötigten zusammenhängenden 16-Byte Blöcke und liefert als Ergebnis im AX-Register die Segmentadresse der reservierten Blöcke. Der Fall "nicht genügend zur Verfügung stehender Speicher" bleibt unberücksichtigt. Die Funktion zum Freigeben von Speicherplatz erwartet im ES-Register die Segmentadresse der freizugebenden Blöcke, also eine Adresse die von der Funktion "Allocate Memory" zurückgeliefert wurde.

Das PTL-Team wünscht viel Erfolg