

DER DATENTYP SUCHBAUM – SCHNITTSTELLE

```

{ ----- }
{ }
{ Die Unit stellt einen inhaltslosen Datentypen TSuchBaum zur Verfügung. Um einen Suchbaum mit Inhalt zu erhalten, }
{ muss eine hiervon abgeleitete Klasse erstellt werden, z. B.: }
{ }
{ type TIntSuchbaum = class(TSuchbaum) }
{   public }
{     Wurzelinhalt: integer; }
{     constructor Create(w: Integer); }
{     function equal(e: TSuchbaum): boolean; override; }
{     function higher(e: TSuchbaum): boolean; override; }
{     function WurzelinhaltToString: string; override; }
{   end; }
{ }
{ Die Routinen equal (gleich) und higher (größer) dienen zur Einsortierung in den Suchbaum. }
{ }
{ ----- Tool zum Suchbaum, (C) 2001 D. Garmann - }

unit Suchbaum;

interface

uses ExtCtrls, binbaum;

type TSuchBaum=class(TBinBaum)
  private
  function IsoliereGroesstesElement(var grElem: TSuchbaum): TSuchbaum;
    { Liefert den SuchBaum ohne das groesste Element. Dieses steht abgekapselt in grElement }
  function EchtLoeschen: TSuchbaum;
    { Liefert den neuen Suchbaum ohne das zuvor existierende (!!!) Suchelement e }

  public
    { müssen für einen neuen Suchbaum-Typen definiert werden }
  function equal(e: TSuchbaum): boolean; virtual; abstract;
  function higher(e: TSuchbaum): boolean; virtual; abstract;
    { werden aus den beiden oberen Funktionen abgeleitet }
  function notequal(e: TSuchbaum): boolean;
  function higherequal(e: TSuchbaum): boolean;
  function lower(e: TSuchbaum): boolean;
  function lowerequal(e: TSuchbaum): boolean;

  function SuchEinfuegen(e: TSuchbaum; var ExistiertBereits: boolean): TSuchbaum;
    { Liefert den neuen SuchBaum mit dem eingefügten Suchelement }
  function SuchLoeschen(e: TSuchbaum; var Existiert: boolean): TSuchbaum;
    { Liefert den neuen Suchbaum ohne das eventuell vorhandene Suchelement e }
    { war es nicht vorhanden => Existiert = false }
end;

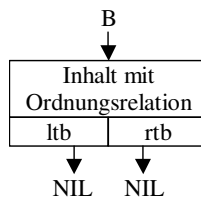
```

Grafisch lässt sich dieser Datentyp wie folgt darstellen: **var** B: TSuchBaum;

leerer Suchbaum



Blatt



beliebiger Suchbaum

