Lösungen zu den Übungsaufgaben zu Feldern

Aufgabe 1: Punktberechnung

```
procedure TForm1.Button1Click(Sender: TObject);
type punkt = array[1..3] of double;
var
  d, p, q: punkt;
  i: integer;
  abstand: double;
begin
  // Eingabe
  p[1]:= StrToFloat(Edit1.Text);
  p[2]:= StrToFloat(Edit2.Text);
  p[3]:= StrToFloat(Edit3.Text);
  q[1]:= StrToFloat(Edit4.Text);
  q[2]:= StrToFloat(Edit5.Text);
  q[3]:= StrToFloat(Edit6.Text);
  // Berechnung der Differenz
  for i := 1 to 3 do d[i] := q[i]- p[i];
  // Berechnung des Abstandes der Punkte
  abstand := sqrt(d[1]*d[1]+d[2]*d[2]+d[3]*d[3]);
  // Ausgabe
  Edit7.Text := FloatToStrF(d[1],ffFixed,6,2);
  Edit8.Text := FloatToStrF(d[2],ffFixed,6,2);
  Edit9.Text := FloatToStrF(d[3],ffFixed,6,2);
  Edit10.Text := FloatToStrF(abstand,ffFixed,6,2);
end;
```

Aufgabe 2: Vergleich zweier Felder

```
type
  feld = array[1..4] of integer;

const
  n = 4;

var
  feld1, feld2: feld;


procedure TForm1.EinlesenClick(Sender: TObject);
begin
  feld1[1] := StrToInt(Edit1.Text);
  feld1[2] := StrToInt(Edit2.Text);
  feld1[3] := StrToInt(Edit3.Text);
  feld1[4] := StrToInt(Edit4.Text);
  feld2[1] := StrToInt(Edit5.Text);
  feld2[2] := StrToInt(Edit6.Text);
  feld2[3] := StrToInt(Edit7.Text);
  feld2[4] := StrToInt(Edit8.Text);
  ShowMessage('Die Daten wurden erfolgreich eingelesen!');
end;
```

```
function vergleiche(feld1, feld2: feld): boolean;
var
  i: integer;
begin
  for i := 1 to n do begin
    if feld1[i] = feld2[i] then vergleiche := true
    else begin
      vergleiche := false;
      exit;
    end;
  end;
end;


procedure TForm1.VergleichenClick(Sender: TObject);
var
  gleich: Boolean;
begin
  gleich := vergleiche(feld1,feld2);
  if gleich = true then ShowMessage('Beide Felder sind identisch!')
    else ShowMessage('Die Felder sind nicht identisch!');
end;
```

Aufgabe 3: Vergleich eines Feldes mit einer Zahl

```
type
  feld = array[1..10] of integer;
var
  feld1: feld;
// Füllen des Feldes mit Zufallszahlen
procedure TForm1.Button1Click(Sender: TObject);
var
  i, zufallszahl: integer;
begin
  with StringGrid1 do for i := 1 to ColCount do begin
    zufallszahl := random(100)+1;
    Cells[i-1,0] := IntToStr(zufallszahl);
    feld1[i] := zufallszahl;
  end;
end;
// Vergleich der Feldelemente mit der Zahl
procedure TForm1.Button4Click(Sender: TObject);
var
 i, vergleichszahl: integer;
begin
  Label6.Visible := true;
  Label6.Caption := Edit1.Text;
  Label8.Visible := true;
  Label8.Caption := Edit1.Text;
  Label10.Visible := true;
  Label10.Caption := Edit1.Text;
  vergleichszahl := StrToInt(Edit1.Text);
  for i := 1 to 10 do begin
    if vergleichszahl > feld1[i] then Memo1.Lines.Add(IntToStr(feld1[i]));
    if vergleichszahl = feld1[i] then Memo2.Lines.Add(IntToStr(feld1[i]));
    if vergleichszahl < feld1[i] then Memo3.Lines.Add(IntToStr(feld1[i]));
```

```
  end;
end;
```

Aufgabe 4: Textumkehr

```
procedure TForm1.UmkehrenClick(Sender: TObject);
const
  n = 100;
type
  text = array[1..n] of char;
var
  t: text;
  laenge, i: integer;
  str1, str2: string;
begin
  str1 := Edit1.Text;
  str2 := ' ';
  laenge := length(str1);
  for i := 1 to laenge do t[i] := str1[i];
  for i := laenge downto 1 do  str2 := str2 + t[i];
  Edit2.Text := str2;
end;
```

Aufgabe 5: Palindromtest 1

```
var
  wort, kehrwort: string;

procedure kehre_um (wort: string; var kehrwort: string);
type text = array[1..100] of char;
var
  laenge, i: integer;
  t: text;
begin
  laenge := length(wort);
  kehrwort := '';
  for i := 1 to laenge do t[i] := wort[i];
  for i := laenge downto 1 do kehrwort:= kehrwort + t[i];
end;

function palindromtest (wort, kehrwort: string): boolean;
begin
  if wort = kehrwort then palindromtest := true
  else palindromtest := false;
end;

procedure TForm1.Button1Click(Sender: TObject);
var palindrom: boolean;
begin
  wort := Edit1.Text;
  kehre_um(wort,kehrwort);
  palindrom := palindromtest(wort,kehrwort);
  if palindrom = true then begin
    Label4.Visible := true;
```

```
    Label4.Caption := wort + ' ist ein Palindrom.';
  end
  else begin
    Label4.Visible := true;
    Label4.Caption := wort + ' ist kein Palindrom.';
  end;
end;
```

Zusatzaufgabe zur Aufgabe 5: Palindromtest 2

```
function Palindromtest (Wort: string): Boolean;
var
  Laenge,I: Integer;
begin
  Laenge := Length(Wort);
  for I := 1 to Trunc(Laenge/2) do
  if Wort[I] =  Wort[Laenge+1-I] then Palindromtest := True
  else begin
    Palindromtest := False;
    Exit;
  end;
end;

procedure Tform1.Button1Click(Sender: TObject);
var
  Palindrom: Boolean;
  Wort: string;
begin
  Wort := Edit1.Text;
  Palindrom := Palindromtest(Wort);
  if Palindrom = True then begin
    Label4.Visible := True;
    Label4.Caption := Wort + ' ist ein Palindrom.';
  end
  else begin
    Label4.Visible := True;
    Label4.Caption := Wort + ' ist kein Palindrom.';
  end;
end;
```

Aufgabe 6: Ein kleines Packprogramm

```
type
  feld = array[1..50] of string;
  {Feld großzügig dimensionieren, da bei leeren Elementen
   entsprechend nachgerückt weren muß. Ansonsten gibt es
   eine Fehlermeldung!}
var
  feld1, feld2: feld;
  zaehler1: integer;     // zählt die leeren Felder

procedure TForm1.EinlesenClick(Sender: TObject);
var
  i: integer;
begin
  for i := 1 to 50 do feld1[i] := '-';
  // füllt Feld1 zunächst vollständig mit vordefinierten Zeichen
```

```
  feld1[1] := Edit2.Text;
  feld1[2] := Edit3.Text;
  feld1[3] := Edit4.Text;
  feld1[4] := Edit5.Text;
  feld1[5] := Edit6.Text;
  feld1[6] := Edit7.Text;
  feld1[7] := Edit8.Text;
  feld1[8] := Edit9.Text;
  feld1[9] := Edit10.Text;
  feld1[10] := Edit11.Text;
  ShowMessage('Die Daten wurden erfolgreich eingelesen!');
end;
function leere_Felder_Test(feld1: feld; zeiger: integer): integer;
{Funktion zum Testen der Anzahl der hintereinander(!) liegenden leeren
Elemente}
var
  zaehler: integer;
begin
  zaehler := 0;
  while feld1[zeiger] = '' do begin
    inc(zaehler);  {Zähler für die Anzahl der nachzurückenden Elemente}
    inc(zaehler1); {globaler Zähler für die Anzahl der leeren Elemente}
    inc(zeiger);
  end;
  result := zaehler;
end;

procedure TForm1.PackenClick(Sender: TObject);
var
  i, j,AnzLeereFelder: integer;
begin
  zaehler1 := 0;   {globalen Zähler für die leeren Elemente initialisieren}
  for i := 1 to 10 do begin
    if feld1[i] = '' then begin
      AnzLeereFelder := leere_Felder_Test(feld1,i); {auf Anzahl der leeren
Elemente testen}
      for j := i to 10 do feld1[j] := feld1[j+AnzLeereFelder];
{nachrücken}
      feld2[i] := feld1[i];
    end
    else feld2[i] := feld1[i];
  end;
  for i := 1 to 50 do if feld2[i] = '-' then feld2[i]:= '';
  {nicht benötigte Elemente freigeben}
  Edit12.Text := feld2[1];
  Edit13.Text := feld2[2];
  Edit14.Text := feld2[3];
  Edit15.Text := feld2[4];
  Edit16.Text := feld2[5];
  Edit17.Text := feld2[6];
  Edit18.Text := feld2[7];
  Edit19.Text := feld2[8];
  Edit20.Text := feld2[9];
  Edit21.Text := feld2[10];
  Edit1.Text := IntToStr(zaehler1);  {Ausgabe des globalen Zählers}
end;
```