

Wirtschaftsinformatik 1
LE 06 – Felder, Mengen und zusammengesetzte
Datentypen

Prof. Dr. Thomas Off

<http://www.ThomasOff.de/lehre/beuth/wi1>

1

Einordnung



LE 06 - Felder, Maps und zusammengesetzte Datentypen

2

2

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps

Abschluss und Ausblick

LE 06 - Felder, Maps und zusammengesetzte Datentypen

3

3

Inhalt

Rückblick

Ausgangspunkt


Zusammengesetzte Datentypen

Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps

Abschluss und Ausblick



LE 06 - Felder, Maps und zusammengesetzte Datentypen

4

4

Rückblick







LE 06 - Felder, Maps und zusammengesetzte Datentypen

5

5

Rückblick





Vorprüfende/Kopfgesteuerte Schleife

- Einsatzzweck
 - erst Bedingung prüfen
 - dann ggf. Anweisung ausführen
 - anschließend Wiederholung der Prüfung usw.
- Generelle Syntax

```
Do While <Bedingung>  
<Anweisung(en)>  
Loop
```

```
Do Until <Bedingung>  
<Anweisung(en)>  
Loop
```
- Beispiel


```
Dim i As Integer  
Let i = 0  
Do While i < 5  
  Let i = i + 1  
  Debug.Print i  
Loop
```

```
Dim j As Byte  
Let j = 0  
Do Until j > 4  
  Let j = j + 1  
  Debug.Print j  
Loop
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

6

6

Rückblick  BHT

Nachprüfende/Fußgesteuerte Schleife

- Einsatzzweck
 - erst Anweisung ausgeführt
 - dann Bedingung prüfen
 - anschließend Wiederholung der Anweisungsausführung usw.
- Generelle Syntax


```
Do
  <Anweisung(en)>
Loop While <Beding.>
```


```
Do
  <Anweisung(en)>
Loop Until <Beding.>
```
- Beispiel


```
Dim i As Integer
Let i = 0
Do
  Let i = i + 1
  Debug.Print i
Loop While i < 5
```

```
Dim j As Byte
Let j = 0
Do
  Let j = j + 1
  Debug.Print j
Loop Until j > 4
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 7

7

Rückblick  BHT

Zählerschleifen


- Einsatz
 - Vorher bekannte Anzahl von Wiederholungen
 - Anzahl gesteuert über Start und Ende
 - Ausführung der Anweisung solange Anzahl Wiederholungen das Ende noch überschritten hat
- Generelle Syntax


```
For <Var> = <Wert/Ausdr> To <Ausdr> Step <Schrittw>
  <Anweisung(en)>
Next
```
- Beispiel


```
Dim i As Integer
For i = 1 To 10 Step 2
  Debug.Print i
Next
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 8

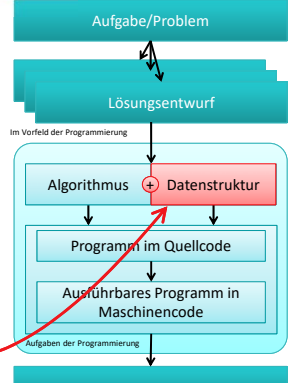
8

Rückblick  BHT

Programmierung als stufenweise Umsetzung

- von Algorithmus und Datenstruktur
- in Quellcode einer Programmiersprache und
- der Überführung in ein ausführbares Programm im Maschinencode


Heute: Datenstrukturen für Felder, Mengen und zusammengesetzte Typen



LE 06 - Felder, Maps und zusammengesetzte Datentypen 9

9

Rückblick (LE02)



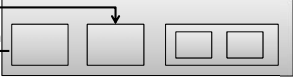
BHT


Komplexe Datenelemente

- Aufbau
 - fassen mehrere einfache Datenelemente oder andere komplexe Datenelemente zusammen
 - Repräsentieren häufig Dinge und Konzepte der Realität
- Beispiel: komplexes Datenelement "Person" fasst einfache Datenelemente "Name", "Vorname" und komplexes Datenelement "Adresse" zusammen
- Zugriff auf einzelne Elemente des komplexen Datenelementes möglich (Lesen, Schreiben)

schreibe 7

Wert lesen






LE 06 - Felder, Maps und zusammengesetzte Datentypen

10

10


Rückblick (LE02)



BHT

Datenstrukturen

- Sequentielle Liste/Feld (Array)
- Verkettete Liste
- Map
- Stapel (Stack)
- Schlange (Queue)
- Graph
- Baum
- ...




LE 06 - Felder, Maps und zusammengesetzte Datentypen

11

11

Rückblick (LE02)



BHT

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld

i_0

i_1

i_2

...

i_{n-1}

i_n

Index

0


1

2

...

n-1

n



LE 06 - Felder, Maps und zusammengesetzte Datentypen

12

12

Rückblick (LE02)

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld	i_0	i_1	i_2	...	i_{n-1}	i_n
Index	0	1	2	...	n-1	n

- Einfügen

LE 06 - Felder, Maps und zusammengesetzte Datentypen 13

13

Rückblick (LE02)

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld	i_0	i_1	i_2	...	i_{n-1}	i_n
Index	0	1	2	...	n-1	n

- Einfügen

Liste	1	2	3	5	6	7		
-------	---	---	---	---	---	---	--	--

4

LE 06 - Felder, Maps und zusammengesetzte Datentypen 14

14

Rückblick (LE02)

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld	i_0	i_1	i_2	...	i_{n-1}	i_n
Index	0	1	2	...	n-1	n


- Einfügen

Liste	1	2	3	5	6	7		
-------	---	---	---	---	---	---	--	--

4

LE 06 - Felder, Maps und zusammengesetzte Datentypen 15

15

Rückblick (LE02)  **BHT**

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld

i_0	i_1	i_2	...	i_{n-1}	i_n
-------	-------	-------	-----	-----------	-------

Index 0 1 2 ... n-1 n


- Einfügen

Liste

1	2	3		5	6	7		
---	---	---	--	---	---	---	--	--


↑

4



LE 06 - Felder, Maps und zusammengesetzte Datentypen 16

16

Rückblick (LE02)  **BHT**

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld


i_0	i_1	i_2	...	i_{n-1}	i_n
-------	-------	-------	-----	-----------	-------

Index 0 1 2 ... n-1 n

- Einfügen


Liste

1	2	3	4	5	6	7		
---	---	---	---	---	---	---	--	--



LE 06 - Felder, Maps und zusammengesetzte Datentypen 17

17

Rückblick (LE02)  **BHT**

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld

i_0	i_1	i_2	...	i_{n-1}	i_n
-------	-------	-------	-----	-----------	-------

Index 0 1 2 ... n-1 n

- Einfügen

Liste

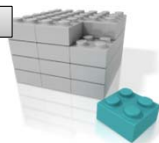
1	2	3	4	5	6	7		
---	---	---	---	---	---	---	--	--

- Entfernen

Liste


1	2	3	4	5	6	7		
---	---	---	---	---	---	---	--	--

↓



LE 06 - Felder, Maps und zusammengesetzte Datentypen 18

18

Rückblick (LE02)  **BHT**

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld

i_0	i_1	i_2	...	i_{n-1}	i_n
-------	-------	-------	-----	-----------	-------

Index 0 1 2 ... n-1 n

- Einfügen


Liste

1	2	3	4	5	6	7		
---	---	---	---	---	---	---	--	--

- Entfernen


Liste

1	2	3	4		6	7		
---	---	---	---	--	---	---	--	--



LE 06 - Felder, Maps und zusammengesetzte Datentypen 19

19

Rückblick (LE02)  **BHT**

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld

i_0	i_1	i_2	...	i_{n-1}	i_n
-------	-------	-------	-----	-----------	-------

Index 0 1 2 ... n-1 n

- Einfügen


Liste

1	2	3	4	5	6	7		
---	---	---	---	---	---	---	--	--

- Entfernen


Liste

1	2	3	4		6	7		
---	---	---	---	--	---	---	--	--



LE 06 - Felder, Maps und zusammengesetzte Datentypen 20

20

Rückblick (LE02)  **BHT**

Sequentielle Liste/Feld (Array)

- Lesender Zugriff über Index frei möglich

Liste/Feld

i_0	i_1	i_2	...	i_{n-1}	i_n
-------	-------	-------	-----	-----------	-------

Index 0 1 2 ... n-1 n

- Einfügen


Liste

1	2	3	4	5	6	7		
---	---	---	---	---	---	---	--	--

- Entfernen

Liste


1	2	3	4	6	7			
---	---	---	---	---	---	--	--	--



LE 06 - Felder, Maps und zusammengesetzte Datentypen 21

21


Rückblick (LE02)



BHT

Datenstrukturen


- Sequentielle Liste/Feld (Array)
- Verkettete Liste
- Map
- Stapel (Stack)
- Schlange (Queue)
- Graph
- Baum
- ...



LE 06 - Felder, Maps und zusammengesetzte Datentypen22

22

Rückblick (LE02)



BHT

Map

- Lesender Zugriff
 - über Schlüssel (Key) frei möglich,
 - teilweise zusätzlich Zugriff über einen Index möglich (z.B. in VBA)

Index	1	2	3	...	n-1	n
Map	k_0	k_1	k_2	...	k_{n-1}	k_n

e_1

e_2


e_3

...

e_{n-1}

e_n


- Einfügen
 - mit Angabe eines Schlüssels (der in VBA noch nicht vergeben sein darf)
 - Vergrößerung erfolgt bei Bedarf automatisch
- Entfernen über Schlüssel bzw. Index



LE 06 - Felder, Maps und zusammengesetzte Datentypen23

23


Rückblick (LE02)



BHT

Datenstrukturen

- Sequentielle Liste/Feld (Array)
- Verkettete Liste
- Map
- Stapel (Stack)
- Schlange (Queue)
- Graph
- Baum
- ...




LE 06 - Felder, Maps und zusammengesetzte Datentypen24

24

Inhalt
Rückblick
Ausgangspunkt
Zusammengesetzte Datentypen
Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps
Abschluss und Ausblick



BHT

LE 06 - Felder, Maps und zusammengesetzte Datentypen25

25

Inhalt
Rückblick
Ausgangspunkt
Zusammengesetzte Datentypen
Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps
Abschluss und Ausblick

BHT

LE 06 - Felder, Maps und zusammengesetzte Datentypen26

26

Inhalt
Rückblick
Ausgangspunkt
Zusammengesetzte Datentypen
Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps
Abschluss und Ausblick

BHT

LE 06 - Felder, Maps und zusammengesetzte Datentypen27

27

Ausgangspunkt

- Variable hat Ähnlichkeiten mit Schubladen in Möbeln
 - speichern Wert eines bestimmten Typs
 - Wert oder Ausdruck kann ihr zugewiesen werden
 - bietet lesenden, schreibenden und ändernden Zugriff
 - hat Bezeichner

LE 06 - Felder, Maps und zusammengesetzte Datentypen 28

28

Ausgangspunkt

... aber

- in eine Schublade kann ich mehrere Dinge des gleichen Typs legen
- in einer Variable kann ich jedoch nur einen Wert eines bestimmten Typs speichern

LE 06 - Felder, Maps und zusammengesetzte Datentypen 29

29

Ausgangspunkt

... und außerdem

- hat die Besteckschublade einen Einsatz, damit z.B. 6 Löffel, 6 Messer, 6 Teelöffel und 6 Gabeln nicht durcheinander kommen
- eine Variable, selbst wenn sie mehrere Werte speichern könnte, würde mir aber keine Struktur anbieten

LE 06 - Felder, Maps und zusammengesetzte Datentypen 30

30

Ausgangspunkt BHT
... oder geht das doch?



LE 06 - Felder, Maps und zusammengesetzte Datentypen 31

31

Ausgangspunkt BHT
Angenommen,
– wir haben drei Kunden
– wir wollen die Kunden mit
• Name
• Vorname
• Straße, HausNr
• Plz und Ort
– in Variablen speichern und im Direktbereich ausgeben.
Wie ginge das?

```
Sub Bsp60()  
  
Dim strKundeName1 As String  
Dim strKundeVorname1 As String  
Dim strKundeStrasseNr1 As String  
Dim strKundePlzOrt1 As String  
  
Dim strKundeName2 As String  
Dim strKundeVorname2 As String  
Dim strKundeStrasseNr2 As String  
Dim strKundePlzOrt2 As String  
  
Dim strKundeName3 As String  
Dim strKundeVorname3 As String  
Dim strKundeStrasseNr3 As String  
Dim strKundePlzOrt3 As String  
  
' ...  
  
End Sub
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 32

32

Ausgangspunkt BHT
Angenommen,
– wir haben drei Kunden
– wir wollen die Kunden mit
• Name
• Vorname
• Straße, HausNr
• Plz und Ort
– in Variablen speichern und im Direktbereich ausgeben.
Wie ginge das?

```
Sub Bsp60()  
  
' ...  
  
Let strKundeName1 = "Yilmaz"  
Let strKundeVorname1 = "Samir"  
Let strKundeStrasseNr1 = "Straße 2"  
Let strKundePlzOrt1 = "12345 Stadt"  
  
Let strKundeVorname2 = "Michael"  
Let strKundeName2 = "Müller"  
Let strKundePlzOrt2 = "12346 Stadt"  
Let strKundeStrasseNr2 = "Weg 14"  
  
Let strKundeVorname3 = "Barbara"  
Let strKundeName3 = "Müller"  
Let strKundePlzOrt3 = "12346 Stadt"  
Let strKundeStrasseNr3 = "Weg 14"  
  
' ...  
  
End Sub
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 33

33

Ausgangspunkt

Angenommen,

- wir haben drei Kunden
- wir wollen die Kunden mit
 - Name
 - Vorname
 - Straße, HausNr
 - Plz und Ort
- in Variablen speichern und im Direktbereich ausgeben.

Wie ginge das?

```
Sub Bsp60()  
    ...  
  
    Debug.Print strKundeVorname1 & _  
        " " & strKundeName1 & _  
        " " & strKundePlzOrt1 & _  
        " " & strKundeStrasseNr1  
  
    Debug.Print strKundeVorname2 & _  
        " " & strKundeName2 & _  
        " " & strKundePlzOrt2 & _  
        " " & strKundeStrasseNr2  
  
    Debug.Print strKundeVorname3 & _  
        " " & strKundeName3 & _  
        " " & strKundePlzOrt3 & _  
        " " & strKundeStrasseNr3  
  
End Sub
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

34

34

Ausgangspunkt

Angenommen,

- wir haben drei Kunden
- wir wollen die Kunden mit
 - Name
 - Vorname
 - Straße, HausNr
 - Plz und Ort
- in Variablen speichern und im Direktbereich ausgeben.

Wie ginge das?

```
Sub Bsp60()  
    ...  
  
    Debug.Print strKundeVorname1 & _  
        " " & strKundeName1 & _  
        " " & strKundePlzOrt1 & _  
        " " & strKundeStrasseNr1  
  
    Debug.Print strKundeVorname2 & _  
        " " & strKundeName2 & _  
        " " & strKundePlzOrt2 & _  
        " " & strKundeStrasseNr2  
  
    Debug.Print strKundeVorname3 & _  
        " " & strKundeName3 & _  
        " " & strKundePlzOrt3 & _  
        " " & strKundeStrasseNr3  
  
End Sub
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

35

35

Ausgangspunkt

Angenommen,

- wir haben drei Kunden
- wir wollen die Kunden mit
 - Name
 - Vorname
 - Straße, HausNr
 - Plz und Ort
- in Variablen speichern und im Direktbereich ausgeben.

Es ginge aber, ...

- nicht mit vielen Kunden und
- die zusammenhängende Struktur fehlt.




LE 06 - Felder, Maps und zusammengesetzte Datentypen

36

36

Inhalt



Rückblick

Ausgangspunkt


Zusammengesetzte Datentypen

Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps

Abschluss und Ausblick




LE 06 - Felder, Maps und zusammengesetzte Datentypen

37

37

Inhalt



Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps


Abschluss und Ausblick

LE 06 - Felder, Maps und zusammengesetzte Datentypen

38

38

Inhalt



Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps

Abschluss und Ausblick

LE 06 - Felder, Maps und zusammengesetzte Datentypen

39

39

Zusammengesetzte Datentypen

BHT

Strukturdefinition

- Definition des Typs
- einzelne Eigenschaften des Typs basierend auf vorhanden Datentypen
- innerhalb des Moduls



```
' Generelle Syntax
Type <Typbezeichner>
  <Eigenschaft1> As <Datentyp>
  <Eigenschaft2> As <Datentyp>
  ' ...
End Type
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

40

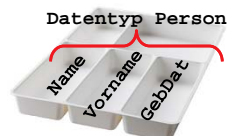
40

Zusammengesetzte Datentypen

BHT

Strukturdefinition

- Definition des Typs
 - Schlüsselwort Type
 - Bezeichner, per Konvention beginnend mit großem T
 - Schlüsselwort End Type
- einzelne Eigenschaften des Typs basierend auf vorhanden Datentypen
- innerhalb des Moduls



```
' Generelle Syntax
Type <Typbezeichner>
  <Eigenschaft1> As <Datentyp>
  <Eigenschaft2> As <Datentyp>
  ' ...
End Type
```

```
' Beispiel
Type TPerson
  strName As String
  strVorname As String
  datGebDat As Date
End Type
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

41

41

Zusammengesetzte Datentypen

BHT

Strukturdefinition

- Definition des Typs
- einzelne Eigenschaften des Typs basierend auf vorhanden Datentypen, z.B.
 - Vorname, Name als String
 - GebDat als Datum
 - Weitere als Typen (auch benutzerdefinierte)
- innerhalb des Moduls



```
' Generelle Syntax
Type <Typbezeichner>
  <Eigenschaft1> As <Datentyp>
  <Eigenschaft2> As <Datentyp>
  ' ...
End Type
```

```
' Beispiel
Type TPerson
  strName As String
  strVorname As String
  datGebDat As Date
End Type
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

42


42

Zusammengesetzte Datentypen

Strukturdefinition

- Definition des Typs
- einzelne Eigenschaften des Typs basierend auf vorhanden Datentypen
- innerhalb des Moduls
 - außerhalb unseres "Programms", d.h. der Prozedur

Datentyp Person



```
' Generelle Syntax
Type <Typbezeichner>
<Eigenschaft1> As <Datentyp>
<Eigenschaft2> As <Datentyp>
' ...
End Type

' Beispiel
Type TPerson
strName As String
strVorname As String
datGebDat As Date
End Type
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

43


43

Zusammengesetzte Datentypen

Strukturdefinition

- Definition des Typs
- einzelne Eigenschaften des Typs basierend auf vorhanden Datentypen
- innerhalb des Moduls

Datentyp Person



```
' Generelle Syntax
Type <Typbezeichner>
<Eigenschaft1> As <Datentyp>
<Eigenschaft2> As <Datentyp>
' ...
End Type

' Beispiel
Type TPerson
strName As String
strVorname As String
datGebDat As Date
End Type
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

44

44

Zusammengesetzte Datentypen

Strukturdefinition

- Definition des Typs
- einzelne Eigenschaften des Typs basierend auf vorhanden Datentypen
- innerhalb des Moduls

Nutzung

- zur Deklaration von Variablen
 - Konvention: Präfix wie Typ
- Zugriff auf die Eigenschaften des Typs mit der Punkt-Notation
- Abkürzung mit **With** und **End With** möglich

```
Option Compare Database
Option Explicit

Type TPerson
strName As String
strVorname As String
datGebDat As Date
End Type

Sub xyz()
Dim perErste As TPerson
Dim perZweite As TPerson

Let perErste.strName="Meier"
Let perErste.strVorname="Karl"
Let perErste.datGebDat=#11/11/2011#

With perZweite
Let .strName="Meier"
Let .strVorname="Kurt"
Let .datGebDat=#11/11/2011#
End With

' ...
End Sub
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

45

45

Zusammengesetzte Datentypen: Beispiel 06.01

Ziel

- Definition und Nutzung eines zusammengesetzten Datentypen

Aufgabe

- Definieren Sie einen Datentyp für Kunden mit
 - Kundennummer
 - Name, Vorname
 - Straße und HausNr.
 - Plz und Ort
- Nutzen Sie den Datentyps zur Deklaration von drei Variablen des Typs Kunde
- Initialisieren Sie die Variablen mit Werten
- Geben Sie die Variablen im Direktbereich aus



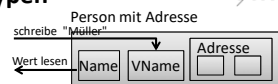
LE 06 - Felder, Maps und zusammengesetzte Datentypen

46

46

Zusammengesetzte Datentypen

- fassen mehrere Eigenschaften definierter Datentypen zusammen
- Repräsentieren häufig Dinge der Realität, z.B. "Person" mit Eigenschaften "Name", "Vorname" und "Adresse"
- Werden als Type definiert und zur Deklaration von Variablen benutzt
- Zugriff auf einzelne Elemente der Variable des zusammengesetzten Datentypen über Punkt-Notation möglich (Lesen, Schreiben)



```
' Generelle Syntax
Type <Typbezeichner>
  <Eigenschaft1> As <Datentyp>
  <Eigenschaft2> As <Datentyp>
  ' ...
End Type
```

```
' Definition
Type TPerson
  strName As String
  adrWohnanschrift As TAdresse
End Type
```

```
' Deklaration und Nutzung
Dim perTom As TPerson
Let perTom.strName = "Tom"
Debug.Print perTom.strName
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

50

50

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps

Abschluss und Ausblick



LE 06 - Felder, Maps und zusammengesetzte Datentypen

51

51

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

Maps

Abschluss und Ausblick

BHT

LE 06 - Felder, Maps und zusammengesetzte Datentypen

52

52

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

Maps

Abschluss und Ausblick

BHT

LE 06 - Felder, Maps und zusammengesetzte Datentypen

53

53

Einfache Felder

Felder (Array) können benutzt werden, um

– mehrere Werte des gleichen Datentyps

– unter einem gemeinsamen Namen (Bezeichner) zu speichern

– jeden Wert einzeln anzusprechen

– innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

IKEA

ASPVIK


Akten-

schränk

Kunden A-Z

Projekt XY

Projekt AB



BHT

LE 06 - Felder, Maps und zusammengesetzte Datentypen

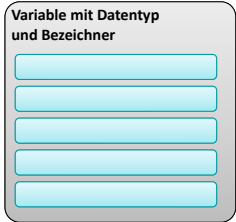

54

54

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

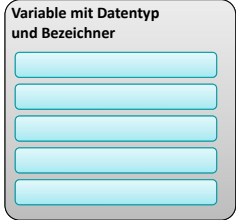


LE 06 - Felder, Maps und zusammengesetzte Datentypen55

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

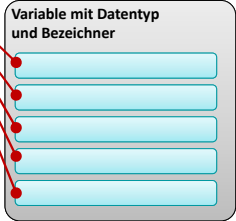


LE 06 - Felder, Maps und zusammengesetzte Datentypen56

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze



LE 06 - Felder, Maps und zusammengesetzte Datentypen57

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

Variable mit Datentyp und Bezeichner

LE 06 - Felder, Maps und zusammengesetzte Datentypen

58

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

Variable mit Datentyp und Bezeichner

LE 06 - Felder, Maps und zusammengesetzte Datentypen

59

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

Variable mit Datentyp und Bezeichner

LE 06 - Felder, Maps und zusammengesetzte Datentypen

60

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

Variable mit Datentyp und Bezeichner

LE 06 - Felder, Maps und zusammengesetzte Datentypen61

61

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

Nachnamen vom Typ String

LE 06 - Felder, Maps und zusammengesetzte Datentypen62

62

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

```
Sub Bsp61a()  
    Dim strNamen(4) As String  
    Let strNamen(0) = "Yilmaz"  
    Let strNamen(1) = "Müller"  
    Let strNamen(2) = "Meier"  
    Let strNamen(3) = "Muster"  
    Let strNamen(4) = "Berg"  
    Debug.Print strNamen(0)  
    Debug.Print strNamen(1)  
    Debug.Print strNamen(2)  
End Sub
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen63

63

Einfache Felder

BHT

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze

```
Sub Bsp61a()  
    Dim strNamen(4) As String  
  
    Let strNamen(0) = "Yilmaz"  
    Let strNamen(1) = "Müller"  
    Let strNamen(2) = "Meier"  
    Let strNamen(3) = "Meier"  
    Let strNamen(4) = "Meier"  
  
    Debug.Print strNamen(0)  
    Debug.Print strNamen(1)  
    Debug.Print strNamen(2)  
End Sub
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

64

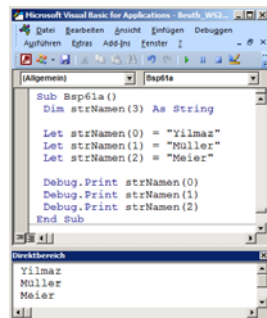
64

Einfache Felder

BHT

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze



LE 06 - Felder, Maps und zusammengesetzte Datentypen

65

65

Einfache Felder: Beispiel 06.02.

BHT

Ziel

- einfache Felder verwenden, um mehrere Werte zu speichern und zu lesen

Aufgabe

- Schreiben Sie ein Programm in dem Sie die Tageshöchst- und Tiefstwerte der Temperaturen des heutigen und der folgenden 6 Tage verarbeiten
 - Deklarieren Sie hierfür zwei Felder mit geeigneten Datentypen
 - Weisen Sie den Felder selbst gewählte Werte zu
 - Geben Sie alle Werte der Felder aus, nutzen Sie hierfür eine Zählerschleife.
 - Erweitern Sie die Ausgabe um die Differenz zwischen Höchst- und Tiefsttemperatur



LE 06 - Felder, Maps und zusammengesetzte Datentypen

66

66

Einfache Felder

Felder (Array) können benutzt werden, um

- mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln über einen Index anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze Bei der Deklaration wird die Anzahl Elemente angegeben

Liste/Feld

i ₀	i ₁	i ₂	...	i _{n-1}	i _n	
Index	0	1	2	...	n-1	n

' Generelle Syntax

```
Dim <Bez>(<n>) As <DTyp>
```

Obergrenze, nicht Anzahl der Elemente

```
Let <Bez>(0) = <WertAusd>
Let <Bez>(1) = <WertAusd>
'...
```

' Beispiel

```
Dim strFeld(2) As String
```

3 Elemente

```
Let strFeld(0) = "Wert 1"
Let strFeld(1) = "Wert 2"
Let strFeld(2) = "Wert 3"

Debug.Print strFeld(1)
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 67

67

Erweiterte Möglichkeiten bei Feldern

Obergrenze und Untergrenze

- Feld verfügt stets über eine Ober- und eine Untergrenze
- in VBA bei Deklaration Angabe beider Grenzen möglich

Liste/Feld

i _n	i _{n+1}	i _{n+2}	...	i _{m-1}	i _m	
Index	n	n+1	n+2	...	m-1	m

(mit n <= m)

– Syntax

```
Dim <Bezeichner>(<n> To <m>) As <Datentyp>
```

– Beispiel

```
Dim strFeld(2 To 4) As String

Let strFeld(3) = "Wert 2"
Debug.Print strFeld(3)
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 68

68

Erweiterte Möglichkeiten bei Feldern

Obergrenze und Untergrenze

- können durch Hilfsfunktionen ermittelt werden

' Beispiel

```
Dim strFeld(2 To 4) As String
```

' Untergrenze (Lower Bound)

```
Debug.Print LBound(strFeld)
```

' Obergrenze (Upper Bound)

```
Debug.Print UBound(strFeld)
```

2

4

LE 06 - Felder, Maps und zusammengesetzte Datentypen 69

69

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder


– Einfache Felder

– Dynamische Felder

– Mehrdimensionale Felder

Maps

Abschluss und Ausblick



LE 06 - Felder, Maps und zusammengesetzte Datentypen

70

70

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

– Einfache Felder

– Dynamische Felder

– Mehrdimensionale Felder

Maps

Abschluss und Ausblick

LE 06 - Felder, Maps und zusammengesetzte Datentypen

71

71

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

– Einfache Felder

– Dynamische Felder

– Mehrdimensionale Felder

Maps

Abschluss und Ausblick

LE 06 - Felder, Maps und zusammengesetzte Datentypen

72

72

Dynamische Felder



Erweiterung des Feldes

- Ober- und Untergrenze legen mögliche Speicherplätze fest
- Erweiterung um zusätzliche Speicherplätze möglich



Syntax

- Vorhandene Inhalte werden bei Vergrößerung gelöscht

```
Dim <Bezeichner>() As <Datentyp>
ReDim <Bezeichner>(<n>)
```

Obergrenze des Feldes n

- Vorhandene Inhalte bleiben bei Vergrößerung mit Preserve erhalten

```
ReDim Preserve <Bezeichner>(<m>)
```

Neue Obergrenze des Feldes m

LE 06 - Felder, Maps und zusammengesetzte Datentypen

73

73

Dynamische Felder: Beispiel 06.03



Ziel

- Verwenden von Ober- und Untergrenzen eines Feldes
- Größenänderung des Feldes

Aufgabe

- Schreiben Sie ein Programm mit dem Sie Kunden mit Ihren Kundennummern (im Bereich von 1000 bis 4999) verarbeiten können
- Deklarieren Sie dazu ein dynamisches Feld
 - vom Typ String, in dem Sie die Namen der Kunden speichern können
 - mit Untergrenze 1000 und Obergrenze 4999
- Erfassen Sie Kundennamen für die Kundennummern 1000 bis 1002
- Erweitern Sie die Obergrenze des Feldes auf 9999. Erfassen Sie den Kunden Nr. 9999.
- Geben Sie erneut den Kunden mit der Kundennummer 1002 aus.



LE 06 - Felder, Maps und zusammengesetzte Datentypen

74

74

Dynamische Felder



Größenanpassung des Feldes

- Ober- und Untergrenze legen mögliche Speicherplätze fest
- Erweiterung um zusätzliche Speicherplätze möglich



Syntax

- Vorhandene Inhalte werden bei Vergrößerung gelöscht

```
Dim <Bezeichner>() As <Datentyp>
ReDim <Bezeichner>(<n>)
```

Obergrenze des Feldes n

- Vorhandene Inhalte bleiben bei Vergrößerung erhalten

```
ReDim Preserve <Bezeichner>(<m>)
```


Neue Obergrenze des Feldes m

LE 06 - Felder, Maps und zusammengesetzte Datentypen

75

75

Inhalt
Rückblick
Ausgangspunkt
Zusammengesetzte Datentypen
Felder
– Einfache Felder
– Dynamische Felder
– Mehrdimensionale Felder
Maps
Abschluss und Ausblick



LE 06 - Felder, Maps und zusammengesetzte Datentypen76

76

Inhalt
Rückblick
Ausgangspunkt
Zusammengesetzte Datentypen
Felder
– Einfache Felder
– Dynamische Felder
– Mehrdimensionale Felder
Maps
Abschluss und Ausblick

LE 06 - Felder, Maps und zusammengesetzte Datentypen77


77

Inhalt
Rückblick
Ausgangspunkt
Zusammengesetzte Datentypen
Felder
– Einfache Felder
– Dynamische Felder
– Mehrdimensionale Felder
Maps
Abschluss und Ausblick

LE 06 - Felder, Maps und zusammengesetzte Datentypen78

78

Mehrdimensionale Felder



Einfache Felder als Liste von einzelnen Werten

Liste/Feld	i ₀	i ₁	i ₂	...	i _{n-1}	i _n
Index	0	1	2	...	n-1	n

Mehrdimensionale Felder als Matrix

– Feld mit Zeilen und Spalten, d.h. zwei Dimensionen

Index	0	1	2	...	n-1	n	
Mehrdimensionales Feld	0	i _{0,0}	i _{0,1}	i _{0,2}	...	i _{0,n-1}	i _{0,n}
	1	i _{1,0}	i _{1,1}	i _{1,2}	...	i _{1,n-1}	i _{1,n}


	m	i _{m,0}	i _{m,1}	i _{m,2}	...	i _{m,n-1}	i _{m,n}

– Felder mit mehr als zwei Dimensionen möglich (ab vier schwer vorstellbar)

LE 06 - Felder, Maps und zusammengesetzte Datentypen 79

79

Mehrdimensionale Felder



Syntax

```
' Zweidimensionales Feld
Dim <Bezeichner>(<n>, <m>) As <Datentyp>

' Mehrdimensionales Feld
Dim <Bezeichner>(<n>, <m>, ...) As <Datentyp>
```

Beispiel

```
Dim strTicTacToe(2,2) As String

Let strTicTacToe(0,2) = "X"
Let strTicTacToe(0,0) = "O"
Let strTicTacToe(2,0) = "X"
Let strTicTacToe(1,1) = "O"
Let strTicTacToe(2,2) = "X"
' ...
' Wer gewinnt?
```


strTicTacToe(zeile, spalte):

O		X
	O	
X		X

LE 06 - Felder, Maps und zusammengesetzte Datentypen 80

80

Mehrdimensionale Felder: Beispiel 06.04




Ziel

– Einsatz mehrdimensionaler Felder

Aufgabe

- Schreiben Sie ein Programm, das den Umsatz der ersten drei Geschäftsjahre für jeweils vier Quartale in einem Feld speichert
- Deklarieren Sie ein zweidimensionales Feld eines geeigneten Datentypen
- Initialisieren Sie das Feld mit selbst gewählten Umsatzzahlen
- Geben Sie die Umsatzzahlen mit Hilfe von Zählerschleifen im Direktbereich aus
- Erweiterung: Bilden Sie die Jahressummen und geben Sie diese ebenfalls aus



LE 06 - Felder, Maps und zusammengesetzte Datentypen 81

81

Mehrdimensionale Felder

Speichern Daten als Matrix, z.B. mit Zeilen und Spalten

Index

0

1

2

...

n-1

n

Mehrdimensionales Feld

0

i_{0,0}

i_{0,1}

i_{0,2}

...

i_{0,n-1}

i_{0,n}

1

i_{1,0}

i_{1,1}

i_{1,2}

...

i_{1,n-1}

i_{1,n}

...

...

...

...

...

...

m

i_{m,0}

i_{m,1}

i_{m,2}

...

i_{m,n-1}

i_{m,n}

mehr als zwei Dimensionen möglich

Syntax und Beispiel

' Mehrdimensionales Feld

Dim <Bez>(<n>, <m>, ...) As <DTyp>

Dim strTtt(2,2) As String

Let strTtt(0,2) = "X"

Let strTtt(0,0) = "O"

Let strTtt(2,0) = "X"

Let strTtt(1,1) = "O"

Let strTtt(2,2) = "X"

' ...

LE 06 - Felder, Maps und zusammengesetzte Datentypen

82

82

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder


– Einfache Felder

– Dynamische Felder

– Mehrdimensionale Felder

Maps

Abschluss und Ausblick



LE 06 - Felder, Maps und zusammengesetzte Datentypen

83

83

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder


– Einfache Felder

– Dynamische Felder

– Mehrdimensionale Felder

Maps

Abschluss und Ausblick



LE 06 - Felder, Maps und zusammengesetzte Datentypen

84

84

LE 06 - Felder, Maps und zusammengesetzte Datentypen

27

Inhalt

BHT

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps

Abschluss und Ausblick

LE 06 - Felder, Maps und zusammengesetzte Datentypen

85

85

Map

BHT

Map in Form der VBA-Collection

- dient der Speicherung von Datenelementen auf die anhand eines eindeutigen Schlüssels zugegriffen werden kann



- Generelle Syntax für Deklaration und Initialisierung

```
' Deklaration
Dim <Bezeichner> As Collection
' Initialisierung (Schlüsselwort Set beachten!)
Set <Bezeichner> = New Collection
```

- Beispiel

```
Dim colKunden As Collection
Set colKunden = New Collection
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

86

86

Map

BHT

Map in Form der VBA-Collection

- Generelle Syntax für Zugriffe

```
' Hinzufügen
<CollectionBezeichner>.Add <WertAusdObj>, <KeyAlsString>
' Lesen
<CollectionBezeichner>.Item(<KeyAlsString>)
' Entfernen
<CollectionBezeichner>.Remove(<KeyAlsString>)
```

- Beispiel

```
' ...
colKunden.Add "Mike Müller", "KndNr0815"
colKunden.Add "Ali Yilmaz", "KndNr4711"
colKunden.Add "Beate Meier", "KndNr9021"

Debug.Print colKunden.Item("KndNr4711")

colKunden.Remove("KndNr9021")
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

87

87

Maps: Beispiel 06.05

BHT

Ziel

- Nutzung einer Map in Form einer VBA-Collection

Aufgabe: Freunden anhand von E-Mailadresse ermitteln

- Deklaration einer Collection und Initialisierung mit einem neuen Objekt
- Hinzufügen der Namen von Freunden zur Collection, dabei deren E-Mailadresse als Schlüssel verwenden
- Vom Benutzer (mit einer InputBox) eine E-Mailadresse erfragen
- Ermitteln des zur eingegebenen E-Mailadressen gehörigen Namens
- Ausgabe des Namens im Meldungsfenster



LE 06 - Felder, Maps und zusammengesetzte Datentypen

88

88

Map

BHT

Map in Form der VBA-Collection

- dient der Speicherung von Datenelementen auf die anhand eines eindeutigen Schlüssels zugegriffen werden kann



- Generelle Syntax für Deklaration und Initialisierung

```
' Deklaration
Dim <Bezeichner> As Collection
' Initialisierung
Set <Bezeichner> = New Collection
```

- Generelle Syntax für Zugriffe

```
' Hinzufügen, Lesen und Entfernen
<CollectionBezeichner>.Add <Wert>
<CollectionBezeichner>.Item(<Key>)
<CollectionBezeichner>.Remove(<Key>)
```

```
Dim colKnd As Collection
Set colKnd = New Collection

colKnd.Add "Müller", "K1"
colKnd.Add "Yilmaz", "K4"
colKnd.Add "Meier", "K2"

Debug.Print colKnd.Item("K4")
colKnd.Remove("K2")
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen

89

89

Inhalt

BHT

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

- Einfache Felder
- Dynamische Felder
- Mehrdimensionale Felder

Maps

Abschluss und Ausblick



LE 06 - Felder, Maps und zusammengesetzte Datentypen

90

90

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

Maps

Abschluss und Ausblick

BHT

– Einfache Felder

– Dynamische Felder

– Mehrdimensionale Felder

LE 06 - Felder, Maps und zusammengesetzte Datentypen

91

91

Inhalt

Rückblick

Ausgangspunkt

Zusammengesetzte Datentypen

Felder

Maps

Abschluss und Ausblick

BHT

– Einfache Felder

– Dynamische Felder

– Mehrdimensionale Felder

LE 06 - Felder, Maps und zusammengesetzte Datentypen

92



92

Abschluss

Zusammengesetzte Datentypen

BHT

– fassen mehrere Eigenschaften definierter Datentypen zusammen

– Repräsentieren häufig Dinge der Realität, z.B. "Person" mit Eigenschaften "Name", "Vorname" und "Adresse"

– Werden als Type definiert und zur Deklaration von Variablenbenutzt

– Zugriff auf einzelne Elemente der Variable des zusammengesetzten Datentypen über Punkt-Notation möglich (Lesen, Schreiben)

Person mit Adresse

schreibe "Müller"

Wert lesen

Name

VName

Adresse

Generelle Syntax

Type <Typbezeichner>

<Eigenschaft1> As <Datentyp>

<Eigenschaft2> As <Datentyp>

...

End Type

Definition

Type TPerson

strName As String

adrWohnanschrift As TAdresse

End Type

Deklaration und Nutzung

Dim perTom As TPerson

Let perTom.strName = "Tom"

Debug.Print perTom.strName

LE 06 - Felder, Maps und zusammengesetzte Datentypen

93

93

Abschluss BHT

Einfache Felder (Array)

Liste/Feld

i ₀	i ₁	i ₂	...	i _{n-1}	i _n
----------------	----------------	----------------	-----	------------------	----------------

Index 0 1 2 ... n-1 n

- speichern mehrere Werte des gleichen Datentyps
- unter einem gemeinsamen Namen (Bezeichner) zu speichern
- jeden Wert einzeln über einen Index anzusprechen
- innerhalb eines Bereichs zwischen Untergrenze und Obergrenze
- Bei der Deklaration wird die Anzahl Elemente angegeben

' Generelle Syntax

```
Dim <Bez>(<n>) As <DTyp>
```

Obergrenze, nicht Anzahl der Elemente

```
Let <Bez>(0) = <WertAusd>
Let <Bez>(1) = <WertAusd>
'...
```

' Beispiel

```
Dim strFeld(2) As String
```

3 Elemente

```
Let strFeld(0) = "Wert 1"
Let strFeld(1) = "Wert 2"
Let strFeld(2) = "Wert 3"
Debug.Print strFeld(1)
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 94

94

Abschluss BHT

Dynamische Erweiterung des Feldes

- Ober- und Untergrenze legen mögliche Speicherplätze fest
- Erweiterung um zusätzliche Speicherplätze möglich

Liste/Feld

i ₀	i ₁	i ₂	...	i _{n-1}	i _n	i _{n+1}	...	i _m
----------------	----------------	----------------	-----	------------------	----------------	------------------	-----	----------------

Index 0 1 2 ... n-1 n n+1 ... m (mit n < m)

- Syntax
- Vorhandene Inhalte werden bei Vergrößerung gelöscht

```
Dim <Bezeichner>() As <Datentyp>
ReDim <Bezeichner>(<n>)
```

Obergrenze des Feldes n

- Vorhandene Inhalte bleiben bei Vergrößerung erhalten

```
ReDim Preserve <Bezeichner>(<m>)
```

Neue Obergrenze des Feldes m

LE 06 - Felder, Maps und zusammengesetzte Datentypen 95

95

Abschluss BHT

Mehrdimensionale Felder

– speichern Daten als Matrix, z.B. mit Zeilen und Spalten

Index		0	1	2	...	n-1	n
Mehrdimensionales Feld	0	i _{0,0}	i _{0,1}	i _{0,2}	...	i _{0,n-1}	i _{0,n}
	1	i _{1,0}	i _{1,1}	i _{1,2}	...	i _{1,n-1}	i _{1,n}

	m	i _{m,0}	i _{m,1}	i _{m,2}	...	i _{m,n-1}	i _{m,n}

- mehr als zwei Dimensionen möglich
- Syntax

' Mehrdimensionales Feld

```
Dim <Bezeichner>(<n>, <m>, ...) As <Datentyp>
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 96

96

Abschluss BHT

Map in Form der VBA-Collection

– dient der Speicherung von Datenelementen auf die anhand eines eindeutigen Schlüssels zugegriffen werden kann

– Generelle Syntax für Deklaration und Initialisierung

```
' Deklaration
Dim <Bezeichner> As Collection
' Initialisierung
Set <Bezeichner> = New Collection
```

– Generelle Syntax für Zugriffe

```
' Hinzufügen, Lesen und Entfernen
<CollectionBezeichner>.Add <Wert>, <Schlüssel>
<CollectionBezeichner>.Item(<Schlüssel>)
<CollectionBezeichner>.Remove(<Schlüssel>)
```

```
Dim colKnd As Collection
Set colKnd = New Collection
colKnd.Add "Müller", "K1"
colKnd.Add "Yilmaz", "K4"
colKnd.Add "Meier", "K2"
Debug.Print colKnd.Item("K4")
colKnd.Remove("K2")
```

LE 06 - Felder, Maps und zusammengesetzte Datentypen 97

97

Ausblick BHT

06 – Zusammenfassung

05 – Debugger und Testen

04 – Fortgeschrittene Konzepte

04.A Oberflächen (Teil 1) Elemente und Eigenschaften

04.B Oberflächen (Teil 2) Ereignisverarbeitung

04.C Zugriff auf Dateisystem und Anwendungen

03 – Grundkonzepte

03.A Wert Ausdruck Variable Konstante Datentyp

03.B Bedingte Ausführung/ Verzweigungen

03.C Schleifen

03.D Felder Mengen

03.E Prozedur Funktion Modul

02 – Grundlagen der Programmierung

01 – Grundbegriffe der Wirtschaftsinformatik

LE 06 - Felder, Maps und zusammengesetzte Datentypen 98

98

Quellen BHT

[1] Produktfotos: <http://www.ikea.de>

LE 06 - Felder, Maps und zusammengesetzte Datentypen 99

99



Wirtschaftsinformatik 1
LE 06 – Felder, Mengen und zusammengesetzte
Datentypen

Prof. Dr. Thomas Off
<http://www.ThomasOff.de/lehre/beuth/wi1>

100
