



BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

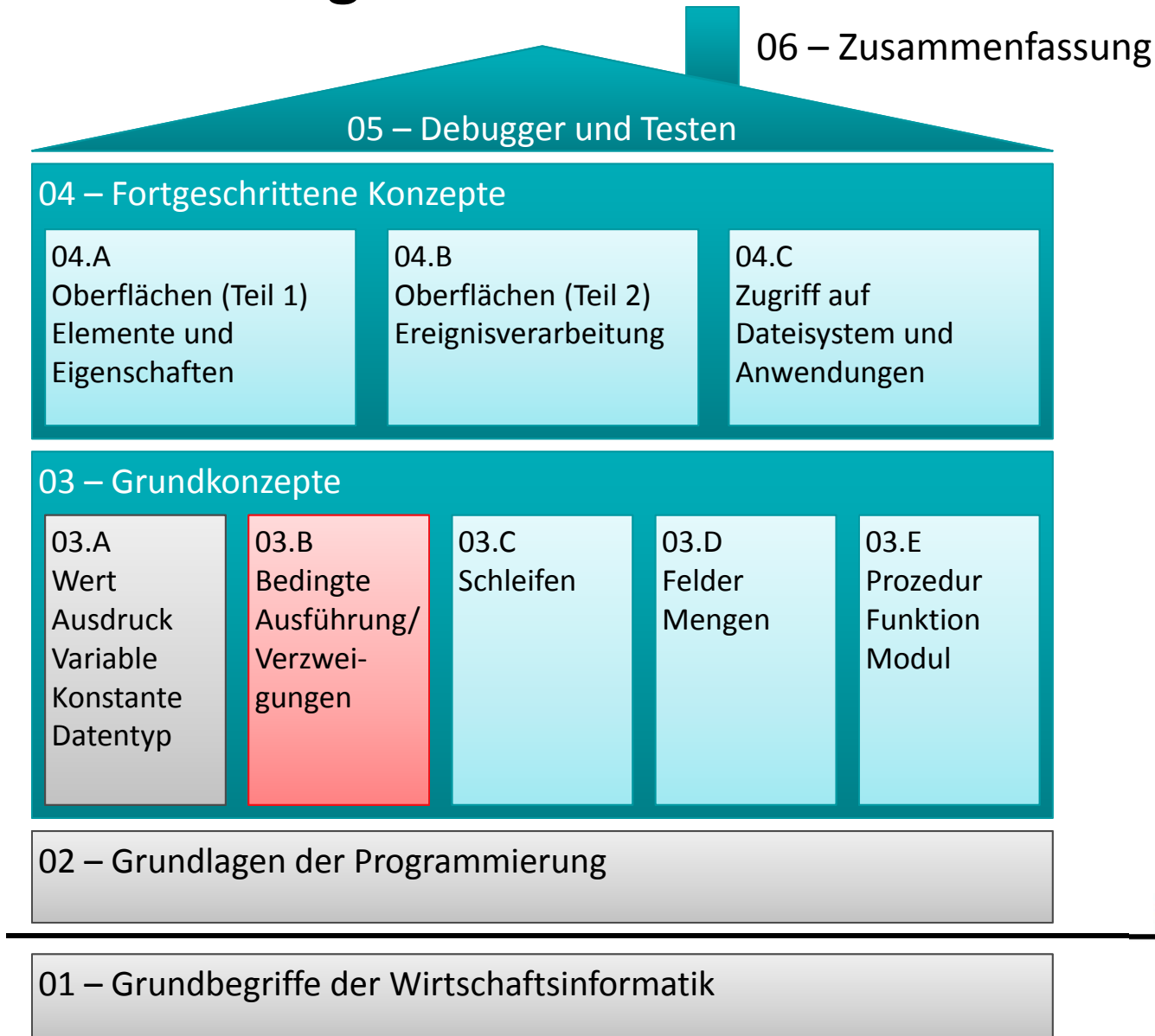
Wirtschaftsinformatik 1

LE 04 – Verzweigungen, Ein-/Ausgabe

Prof. Dr. Thomas Off

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

Einordnung





Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick

Rückblick



Rückblick



Datentyp

- definiert einen Bereich oder eine Menge von Werten und
- Operationen, die auf den Werten möglich sind
 - insbesondere logische Operatoren auf Datentyp Boolean
- bestimmt den zur Speicherung von Werten benötigten Platz
- hat einen eindeutigen Bezeichner

Wert, Ausdruck, Zuweisung, insbesondere

- Auswertung von Ausdrücken
- Operatorprioritäten
- Auswertung der rechten Seite und dann Zuweisung zur linken

Variable



- wird deklariert mit Schlüsselwort Dim
- hat einen **Bezeichner**
- und ist von einem definierten **Datentyp**
- **Werte** oder **Ausdrücke** werden ihr **zugewiesen**
- erstmalige Zuweisung heißt **Initialisierung**
- bietet **Zugriff** auf gespeicherten Wert
 - lesend
 - schreibend/ändernd

Syntax

```
Dim <VarBezeich> As <Datentyp>  
Let <VarBezeich> = <Wert/Ausdr>
```

Beispiele

```
' Weniger gute Bezeichner  
Dim i As Integer  
Dim s As String  
' Aussagekräftige Bezeichner  
Dim bytAlter As Byte  
Dim sglBetrag As Single  
' Initialisierung mit Wert  
Let bytAlter = 20  
' Initialisierung mit Ausdruck  
Let sglBetrag = bytAlter * 3  
' Lesender Zugriff  
Debug.Print bytAlter  
' Ändernder Zugriff  
Let bytAlter = bytAlter + 1  
Let sglBeitrag = 42
```


Rückblick



Konstante

- wird deklariert mit Schlüsselwort **Const**
- hat einen **Bezeichner**
- und ist von einem definierten **Datentyp**
- **Werte** oder **Ausdrücke** können einmalig **zugewiesen** werden
- erstmalige Zuweisung (**Initialisierung**) erfolgt gemeinsam mit **Deklaration**
- bietet lesenden **Zugriff** auf gespeicherten Wert

```
' Generelle Syntax  
' Deklaration und Initialisierung  
' Const <Bezeichner> As <Datentyp> = <WertOderAusdruck>
```

```
' Beispiel
```

```
Const PI As Double = 3.14159265359  
Let dblUmfang = 2 * PI * dblRadius
```

Rückblick



Typumwandlungen - Unterscheidungsmöglichkeit 1

- zwischen nah verwandten Typen (automatisch von VBA ausgeführt, als implizite Typumwandlung bezeichnet)

```
Dim d As Date, s As String  
Let d = #04/05/2015#  
Let s = "Liefertermin: " & d
```

```
Dim i As Integer  
Dim b As Byte  
Let b = 3  
Let i = b
```

- zwischen beliebigen Typen (explizite Nutzung einer Funktion zur Typumwandlung erforderlich)

```
Dim l As Long, s As String  
Let s = "4711"  
Let l = CLng(s)
```

Hier: Nah verwandte Typen, z.B. auch die numerischen Typen

Rückblick



Typumwandlungsfunktionen

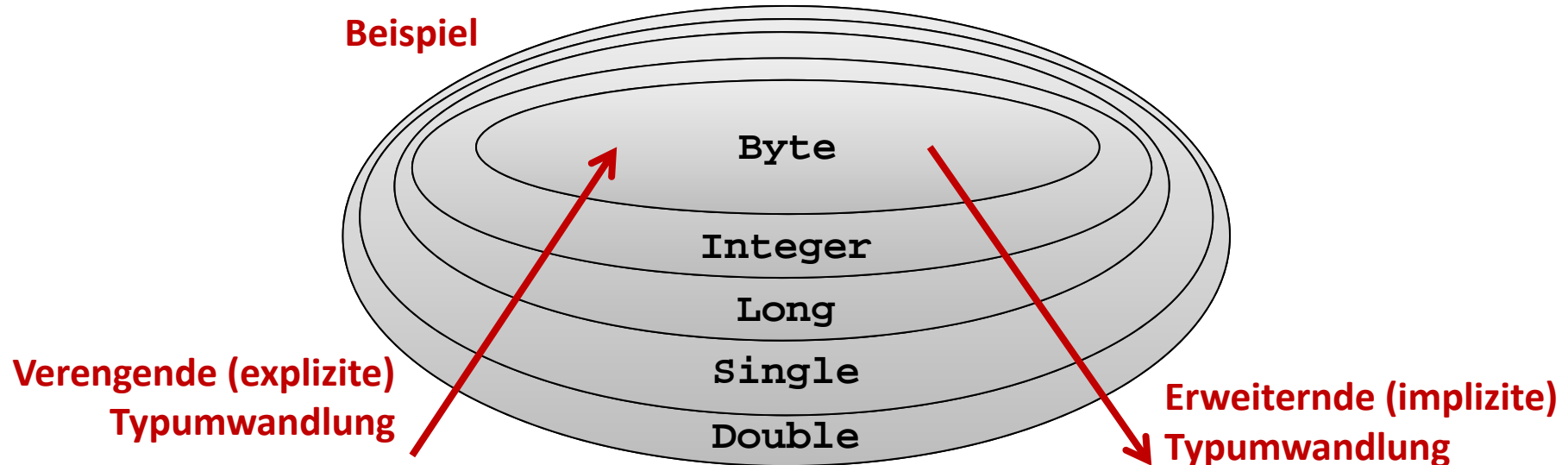
Funktion	Rückgabetyp	Beispiel
CBool	Boolean	Eine gültige Zeichenfolge oder ein gültiger numerischer Ausdruck.
CByte	Byte	0 bis 255.
CCur	Currency	-922.337.203.685.477,5808 bis 922.337.203.685.477,5807.
CDate	Date	Ein beliebiger gültiger Datumsausdruck.
CDbl	Double	-1,79769313486231E308 bis -4,94065645841247E-324 für negative Werte; 4,94065645841247E-324 bis 1,79769313486232E308 für positive Werte.
CInt	Integer	-32.768 bis 32.767; Nachkommastellen werden gerundet.
CLng	Long	-2.147.483.648 bis 2.147.483.647; Nachkommastellen werden gerundet.
CLngLng	LongLong	-9.223.372.036.854.775.808 bis 9.223.372.036.854.775.807; Dezimalen werden gerundet (nur auf 64-Bit-Plattformen zulässig)
CSng	Single	-3,402823E38 bis -1,401298E-45 für negative Werte; 1,401298E-45 bis 3,402823E38 für positive Werte.
CStr	String	Rückgabe für CStr hängt vom Argument <i>Ausdruck</i> ab.

Rückblick



Typumwandlungen - Unterscheidungsmöglichkeit 2

- Verengende Typumwandlung
- Erweiternde Typumwandlung



Rückblick



Typumwandlungen - Unterscheidungsmöglichkeit 2

– Verengende Typumwandlung

- Konvertierung in Typ mit kleinerem Wertebereich
- Verlust von Information (z.B. Genauigkeit)
- müssen explizit befohlen werden, z.B.:
- Beispiele

```
Dim i As Integer, s As Single  
Let s = 3.14  
Let i = CInt(s)
```

– Erweiternde Typumwandlung

- Konvertierung in Typ mit größerem Wertebereich
- Erweiternde Typumwandlungen implizit, d.h. automatisch vorgenommen
- Beispiele

```
Dim i As Integer, b As Byte  
Let b = 3  
Let i = b
```

Rückblick



Typumwandlungsfunktionen

– Generelle Syntax

```
' Generelle Syntax  
' <TUmwdlFkt>(<WertOderAusdruck>)  
' Generelle Syntax in Zuweisung  
' Let <VarBezeichner> = <TUmwdlFkt>(<WertOderAusdruck>)
```

– Beispiele

```
' Verwendung in Zuweisung  
Let intZahl = CInt("42")  
Let strPlz = CStr(14476)  
' Vergleichbar mit Typumwandlung sind zahlreiche  
' Hilfsfunktionen, die auch im Fehlerfall definierte  
' Werte liefern  
Let intZahl = Val("14476")  
Let datGebDatum = DateValue("23.04.1994")
```

Rückblick



Bisherige Bestandteile unserer Programme

- Deklaration
- Anweisungen
 - Initialisierung
 - Zuweisung von Werten und Ausdrücken
 - Ausgabe

mit einem linearem Programmablauf.

```
(Allgemein)
Option Compare Database
Option Explicit

Sub demo0302 ()

' Deklaration der Variablen
Dim curEuro As Currency
Dim curDollar As Currency
Dim sglKursEurUsd As Single

' Initiale Zuweisung der vorgegebenen Werte
Let curEuro = 234
Let sglKursEurUsd = 1.32

' Ausgabe der Werte im Direktbereich
Debug.Print curEuro
Debug.Print sglKursEurUsd

' Durchführung der Berechnung
Let curDollar = curEuro * sglKursEurUsd

' Ausgabe des Ergebnisses im Direktbereich
Debug.Print curDollar

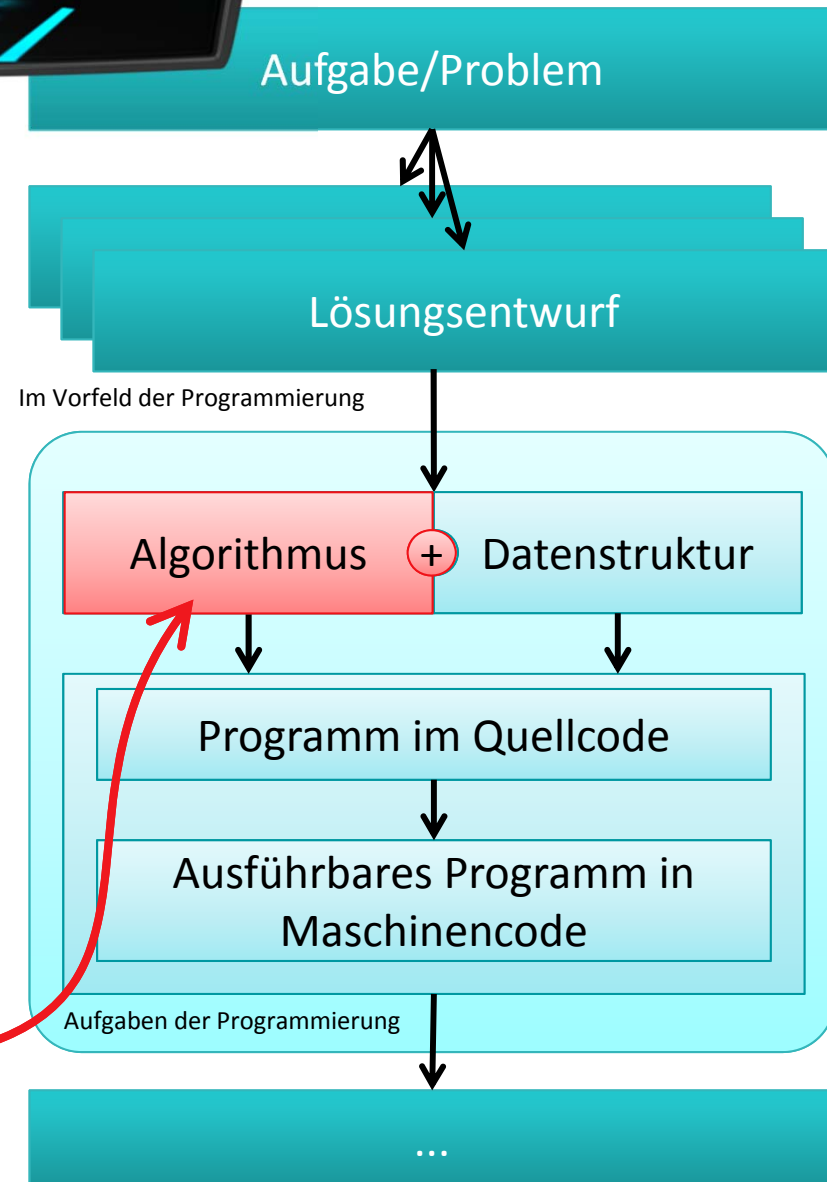
End Sub
```

Rückblick

Programmierung als stufenweise Umsetzung

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

*Heute:
Verzweigungen und bedingte Anweisungen*





Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick





Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



Ausgabe und Eingabe

Ausgabe erfolgt bisher im Direktbereich

- Beispiel

```
Debug.Print "Hallo Welt!"
```

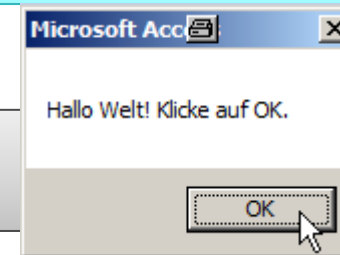
Ausgabemöglichkeit im Dialog mittels MessageBox

- Grundlegende Syntax (einfache Form)

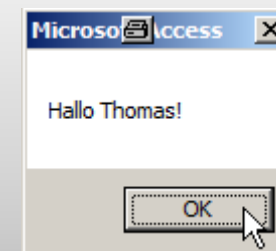
```
MsgBox ("<Meldungstext>")
```

- Beispiel

```
MsgBox ("Hallo Welt! Klicke auf OK.")
```



```
Dim strVorname As String  
Let strVorname = "Thomas"  
MsgBox ("Hallo " & strVorname & "!")
```

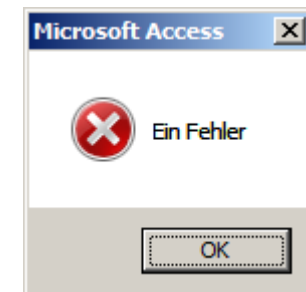
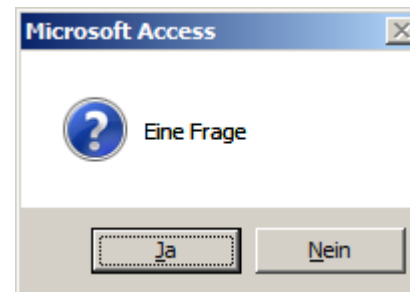
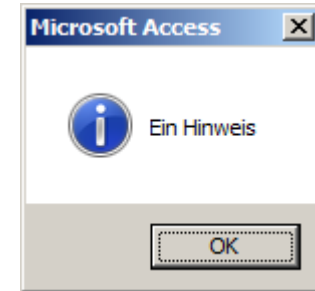


Ausgabe und Eingabe



Erweiterte Ausgabemöglichkeit mittels MessageBox

- Gestaltung der MessageBox mit Konstanten für
 - Symbole, z.B. `vbInformation`, `vbCritical`, `vbQuestion`
 - Schaltflächen, z.B. `vbOKOnly`, `vbYesNo`, `vbOKCancel`
 - ...
- Erfordert den Einsatz der Meldung in einer Zuweisung, um Ergebnis zu speichern
- Generelle Syntax



```
Let <IntegerVar> = MsgBox("<Meldungstext>", <Konst1>+<Konst2>)
```

- Beispiele

```
Dim intResult As Integer  
Let intResult = MsgBox("Ein Hinweis", vbInformation)  
Let intResult = MsgBox("Ein Fehler", vbCritical)  
Let intResult = MsgBox("Eine Frage", vbQuestion + vbYesNo)
```

Ausgabe und Eingabe



Hinweis:

- Verarbeitung des Rückgabewertes durch Vergleich mit weiteren Konstanten (folgt später)

Ausgabe und Eingabe: Beispiel 04.01



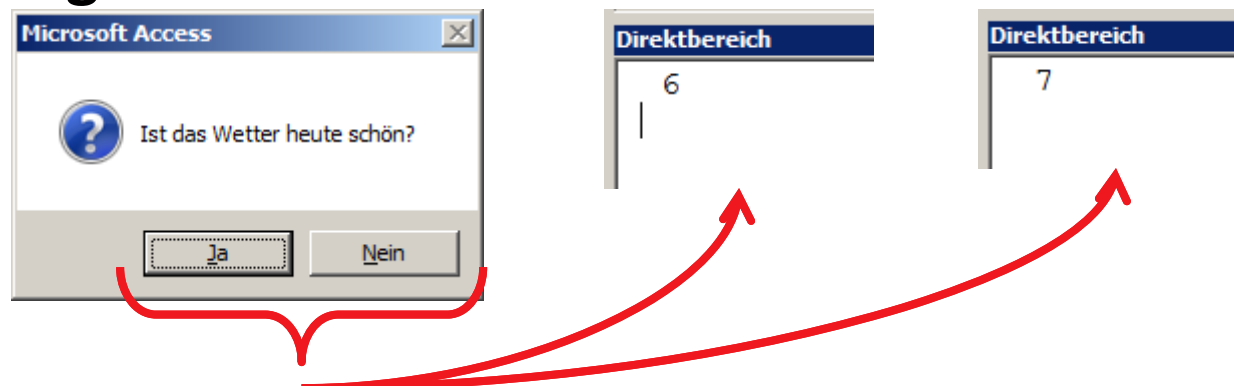
Ziel

- **MsgBox** als Meldungsfenster für eine Ja/Nein-Frage nutzen

Aufgabe

- Schreiben Sie ein Programm, das die Frage "Ist das Wetter schön heute?" in einem Meldungsfenster (MsgBox) darstellt
- Meldungsfenster soll ein Fragezeichen-Icon und die Schaltfläche "Ja" und "Nein" haben
- Rückgabewert des Meldungsfensters soll im Direktbereich ausgegeben werden

Ergebnis



Ausgabe und Eingabe



Eingabe im Dialog mittels InputBox

- InputBox liefert einen String zurück, der auch leer sein kann
- Grundlegende Syntax

```
Let <Variable-vom-Typ-String> = InputBox("<Meldungstext>")
```

- Beispiel

```
Let strName = InputBox("Ihr Name:")  
Let strVorname = InputBox("Ihr Vorname:")
```

Microsoft Access

Ihr Name:

OK

Abbrechen

Microsoft Access

Ihr Vorname:

OK

Abbrechen

Ausgabe und Eingabe



Hinweise

- Typumwandlung in anderen Datentyp als String erfolgt implizit
- Problem ist leerer String (bei Klick auf Abbrechen oder OK bei leerer Eingabe), dem auch durch Typumwandlung kein Wert zugeordnet wird
- um Fehler mit leeren Zeichenketten abzufangen
 - deshalb implizite Typumwandlung in Kombination mit Val()-Funktion (liefert Double) nutzen oder
 - explizite Typumwandlung in Kombination mit Val()-Funktion
- Beispiele

```
Let strName = InputBox("Name:") ' Kein Problem bei leer
Let bytAlter = InputBox("Alter:") ' Problem bei leer
Let bytAlter = CByte(InputBox("Alter:")) ' So ist es besser
Let dblGehalt = Val(InputBox("Gehalt:")) ' Kein Problem
Let datGeb = DateValue(InputBox("GebDat:")) ' Kein Problem
```

Ausgabe und Eingabe: Beispiel 04.02



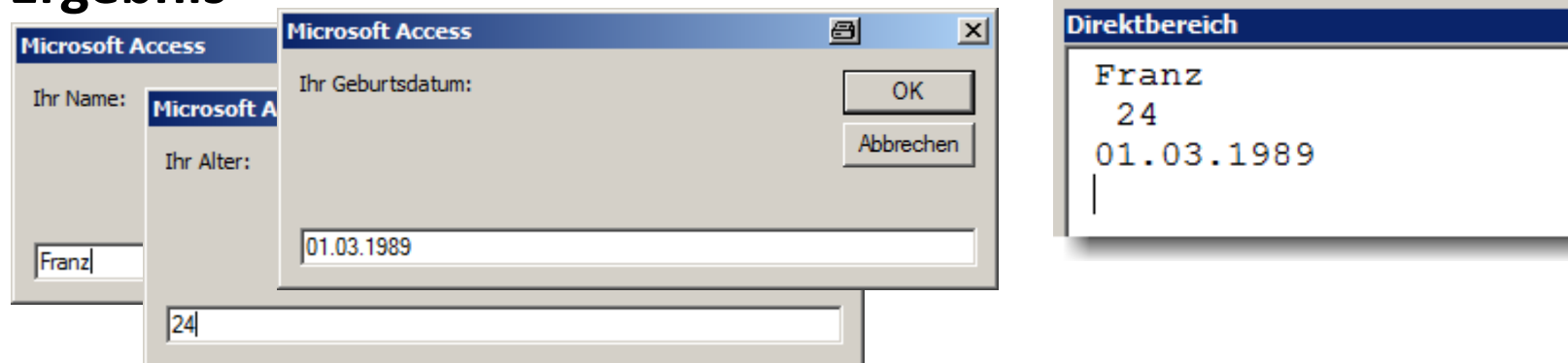
Ziel

- **InputBox** nutzen, um Daten verschiedener Datentypen einzulesen

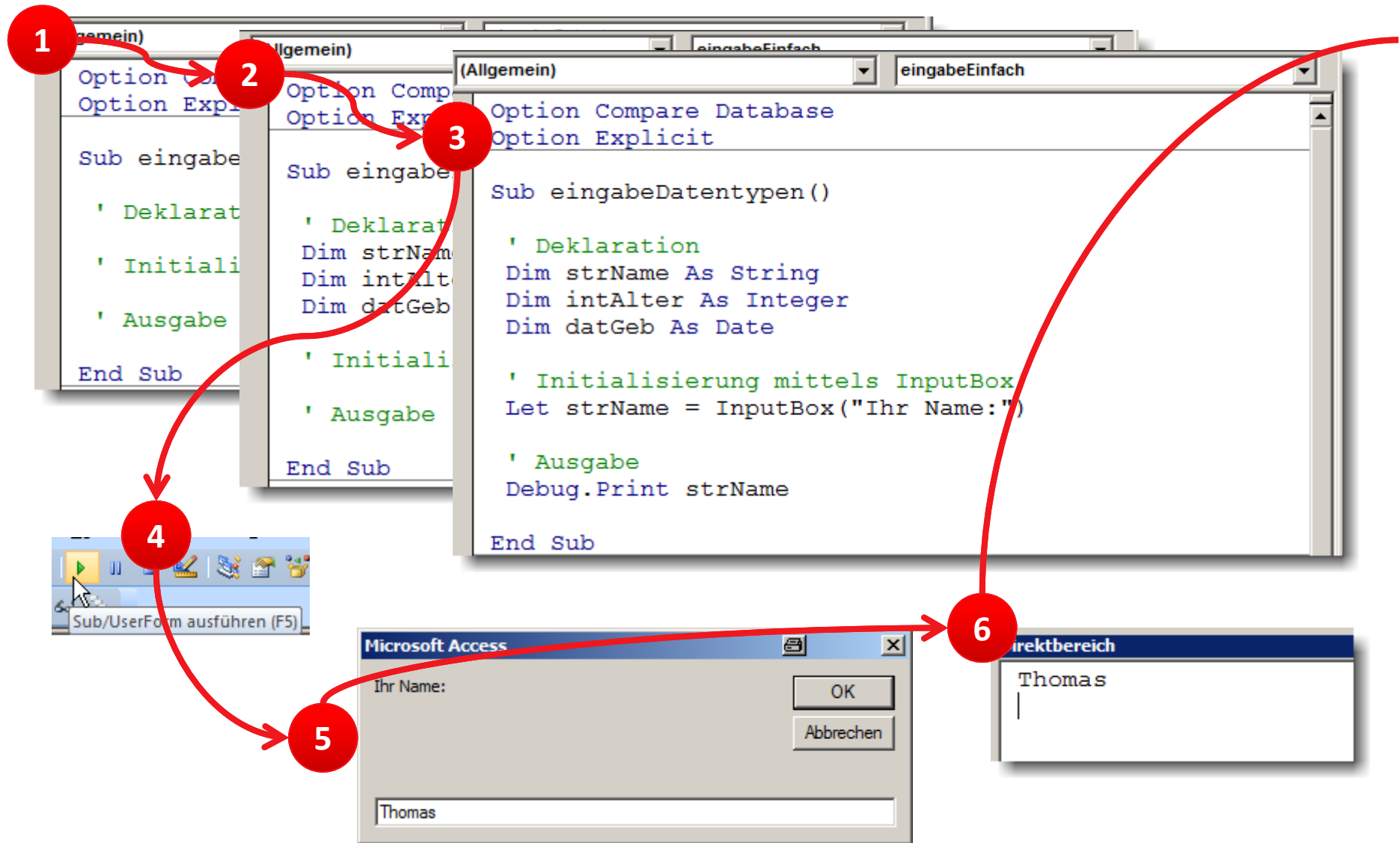
Aufgabe

- Schreiben Sie ein Programm, das per **InputBox** zur Eingabe von Name, Alter und Geburtsdatum auffordert
- Eingegebene Werte sollen jeweils in einer Variable für den Namen (als **String**), das Alter (als **Byte**) und das Geburtsdatum (als **Date**) gespeichert werden
- Anschließend sollen die Werte testweise ausgegeben werden

Ergebnis



Ausgabe und Eingabe: Beispiel 04.02



Ausgabe und Eingabe: Beispiel 04.02



The screenshot illustrates the execution of a Visual Basic program. The main window shows the code editor with the following code:

```
Option Compare Database
Option Explicit

Sub eingabeDatentypen()

    ' Deklaration
    Dim strName As String
    Dim intAlter As Integer
    Dim datGeb As Date

    ' Initialisierung mittels InputBox
    Let strName = InputBox("Ihr Name:")
    Let intAlter = InputBox("Ihr Alter:") ' ob das gutgeht?
    Let datGeb = InputBox("Ihr Geburtsdatum:") ' ob das gutgeht?

    ' Ausgabe
    Debug.Print strName
    Debug.Print intAlter
    Debug.Print datGeb
End Sub
```

Annotations and execution flow:

- 7**: Points to the start of the code.
- 8**: Points to the `Let intAlter = InputBox("Ihr Alter:");` line, with a tooltip showing the 'Sub/User Form ausführen (F5)' button.
- 9**: Points to the `InputBox` dialog box for 'Ihr Alter:'.
- 10**: Points to the `InputBox` dialog box for 'Ihr Name:' with the text 'Thomas' entered.
- 11**: Points to the 'Microsoft Visual Basic' error dialog box showing 'Laufzeitfehler '13': Typen unverträglich'.

Ausgabe und Eingabe: Beispiel 04.02



The image shows a VBA code editor window titled 'eingabeEinfach' with the following code:

```
Option Compare Database
Option Explicit

' Ausgabe und Eingabe: Beispiel 04.01
Sub eingabeDatentypen()

' Deklaration
Dim strName As String
Dim intAlter As Integer
Dim datGeb As Date

' Initialisierung mittels InputBox
Let strName = InputBox("Name:")
Let intAlter = Val(InputBox("Alter:")) ' das geht so
Let datGeb = CDate(Val(InputBox("GebDatum:"))) ' so ist's ganz sicher

' ...

End Sub
```

Three Microsoft Access dialog boxes are shown, illustrating the execution flow:

- 12**: Points to the code editor window.
- 13**: Points to the 'Sub/UserForm ausführen (F5)' button in the code editor's toolbar.
- 14**: Points to the first dialog box titled 'Microsoft Access' with the label 'Ihr Alter:' and a text box containing 'Thomas'. It has 'OK' and 'Abbrechen' buttons.
- 15**: Points to the 'OK' button of the first dialog box.
- 16**: Points to the second dialog box titled 'Microsoft Access' with the label 'GebDatum:' and a text box. It has 'OK' and 'Abbrechen' buttons.
- 17**: Points to the third dialog box titled 'Microsoft Access' with the label 'GebDatum:' and a text box containing 'Thomas', '0', and '00:00:00'. It has 'OK' and 'Abbrechen' buttons.

Ausgabe und Eingabe: Beispiel 04.02



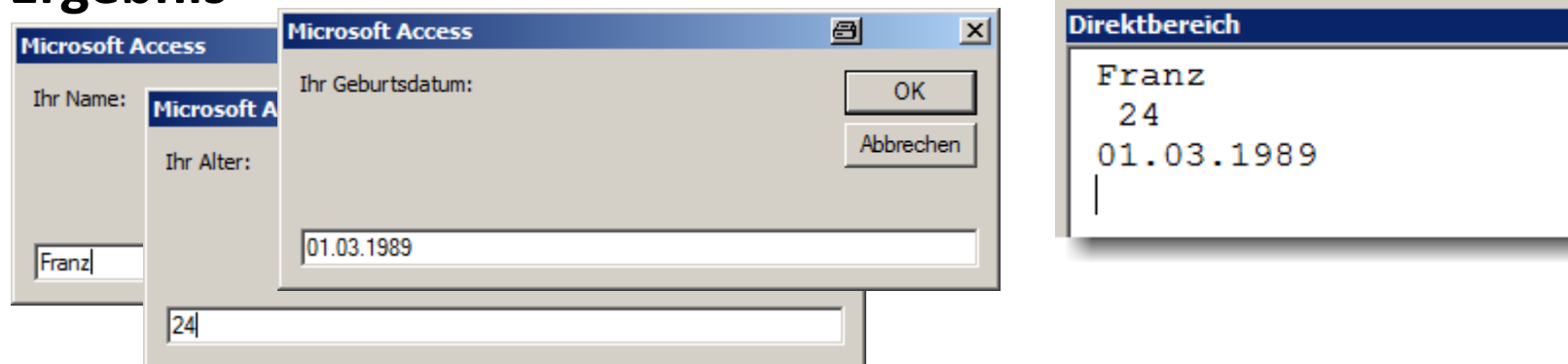
Ziel

- **InputBox** nutzen, um Daten verschiedener Datentypen einzulesen

Aufgabe

- Schreiben Sie ein Programm, das per **InputBox** zur Eingabe von Name, Alter und Geburtsdatum auffordert
- Eingegebene Werte sollen jeweils in einer Variable für den Namen (als **String**), das Alter (als **Byte**) und das Geburtsdatum (als **Date**) gespeichert werden
- Anschließend sollen die Werte testweise ausgegeben werden

Ergebnis





Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick





Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



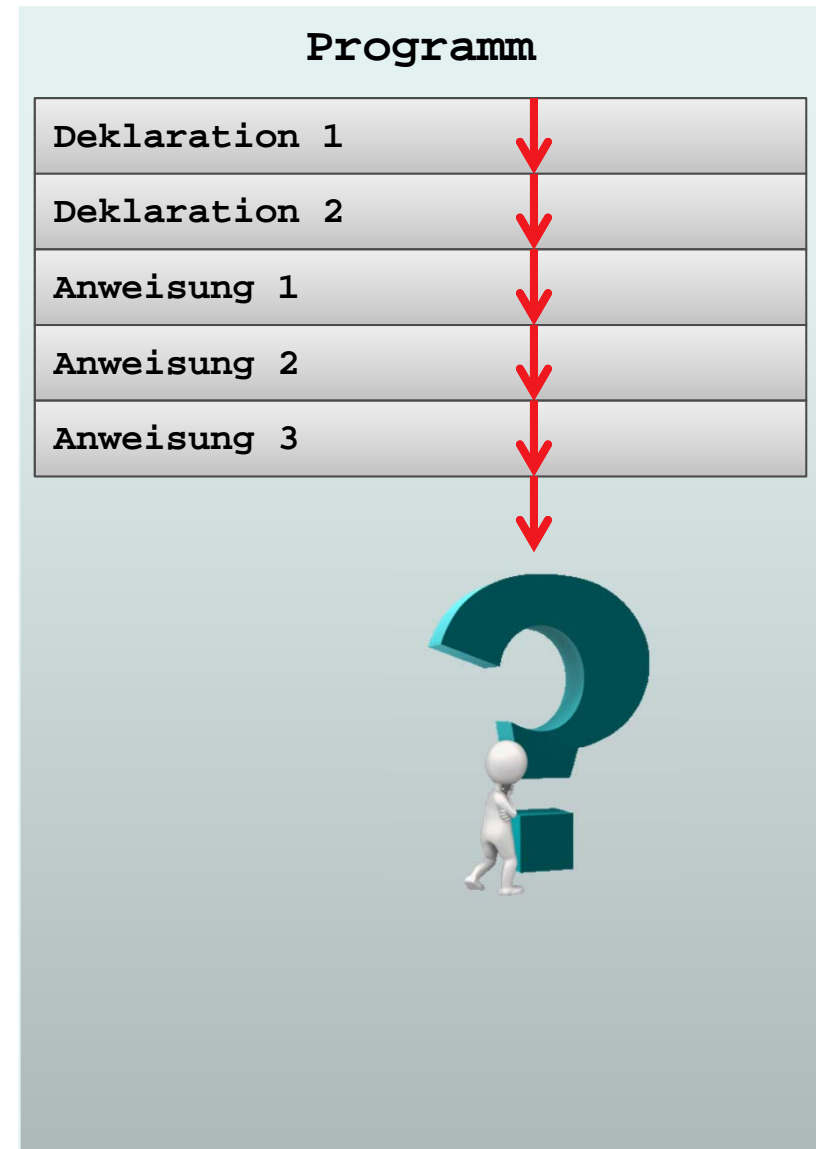
Arten von Verzweigungen

Bisherige Programme mit linearem Ablauf

- erste Deklaration/Anweisung wird ausgeführt,
- dann die nächste ... usw.

Was tun wenn eine Anweisung nur ausgeführt werden soll, wenn Bedingung erfüllt?

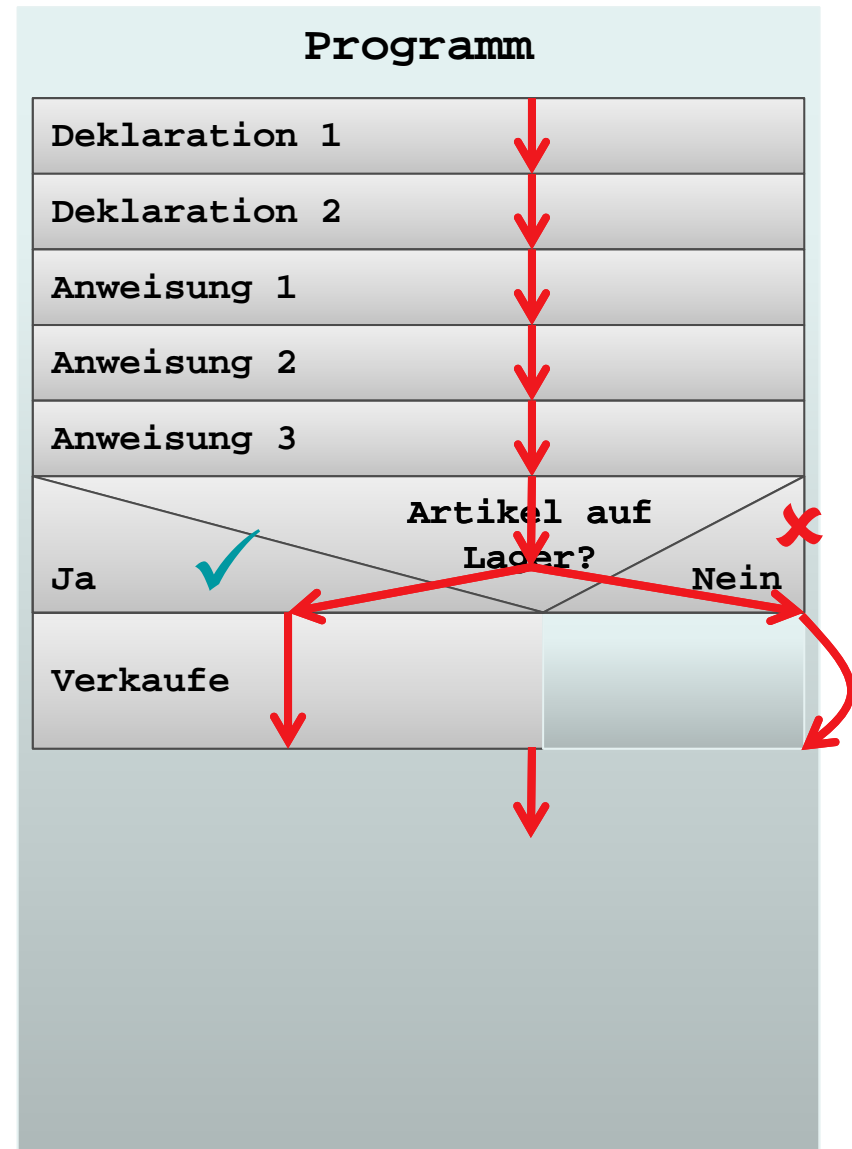
- Verkaufen nur, wenn Artikel auf Lager
- Bestellen nur, wenn Warenkorb nicht leer
- *<Anweisung>* nur, wenn *<Bedingung>* erfüllt



Arten von Verzweigungen

Was tun wenn Anweisung nur ausgeführt werden soll, wenn Bedingung erfüllt?

- Verkaufen nur, wenn Artikel auf Lager



Arten von Verzweigungen



Was tun wenn Anweisung nur ausgeführt werden soll, wenn Bedingung erfüllt?

- Verkaufen nur, wenn Artikel auf Lager
- Bestellen nur, wenn Warenkorb nicht leer





Arten von Verzweigungen

Was tun wenn Anweisung nur ausgeführt werden soll, wenn Bedingung erfüllt?

- Verkaufen nur, wenn Artikel auf Lager
- Bestellen nur, wenn Warenkorb nicht leer
- *<Anweisung 4>* nur, wenn *<Bedingung>* erfüllt

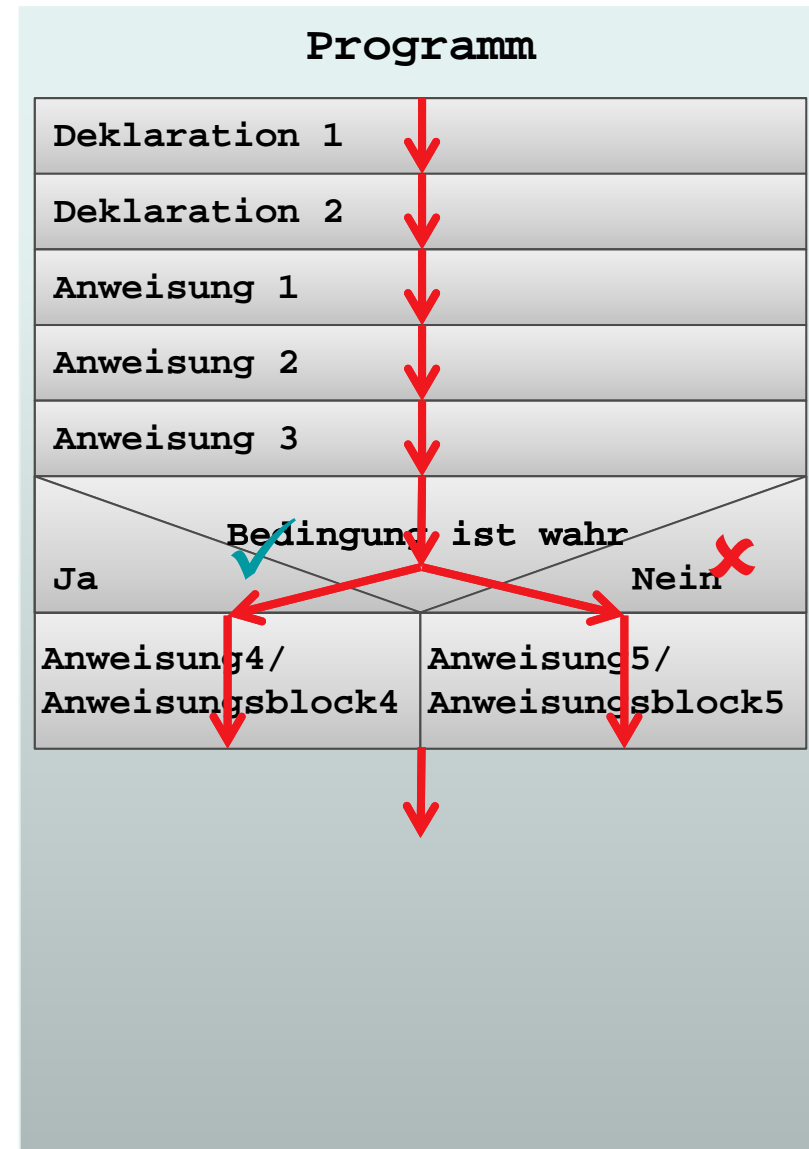




Arten von Verzweigungen

Was tun wenn Anweisung nur ausgeführt werden soll, wenn Bedingung erfüllt, andernfalls aber eine andere Anweisung?

- Verkaufen nur, wenn Artikel auf Lager, sonst Produktion
- Bestellen nur, wenn Warenkorb nicht leer, sonst Fehlermeldung
- *<Anweisung 4>* nur, wenn *<Bedingung>* erfüllt, sonst *<Anweisung 5>*

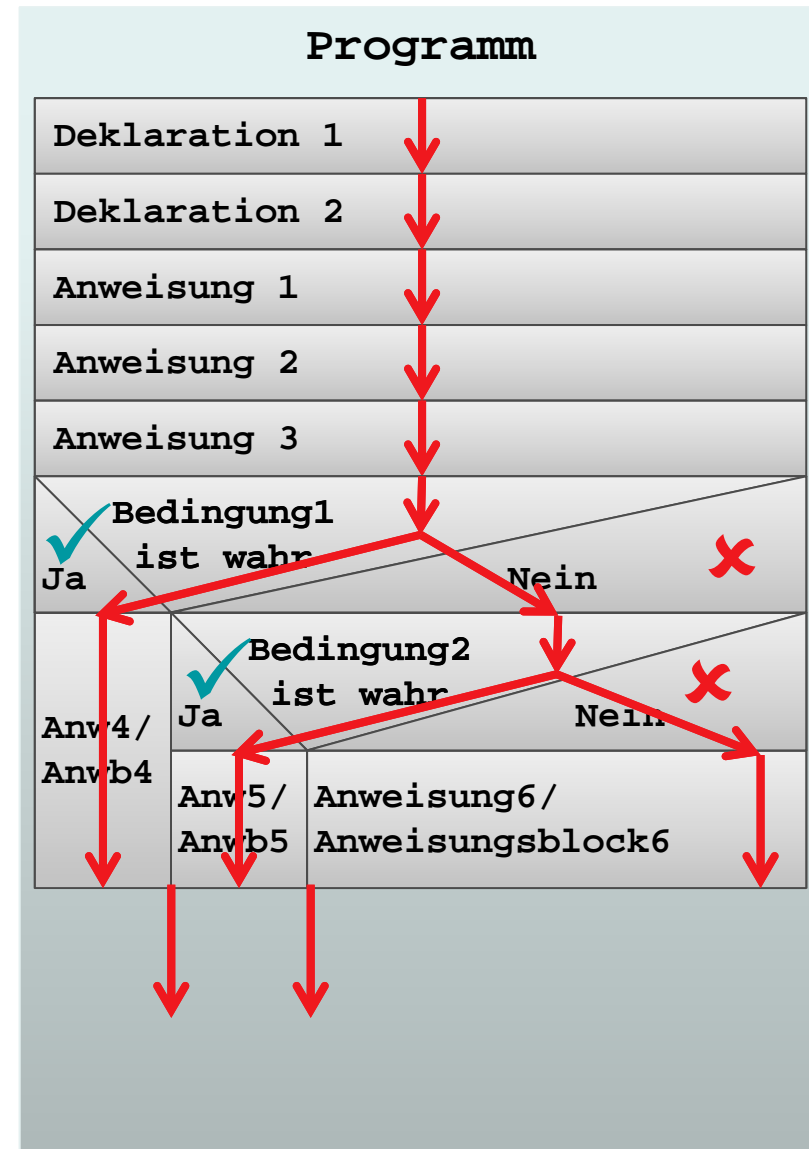


Arten von Verzweigungen



Was tun, wenn

- für *<Anweisung4>* die *<Bedingung1>* erfüllt sein muss,
- andernfalls für *<Anweisung5>* die *<Bedingung2>* erfüllt sein muss,
- und nur sonst *<Anweisung 6>* auszuführen ist?

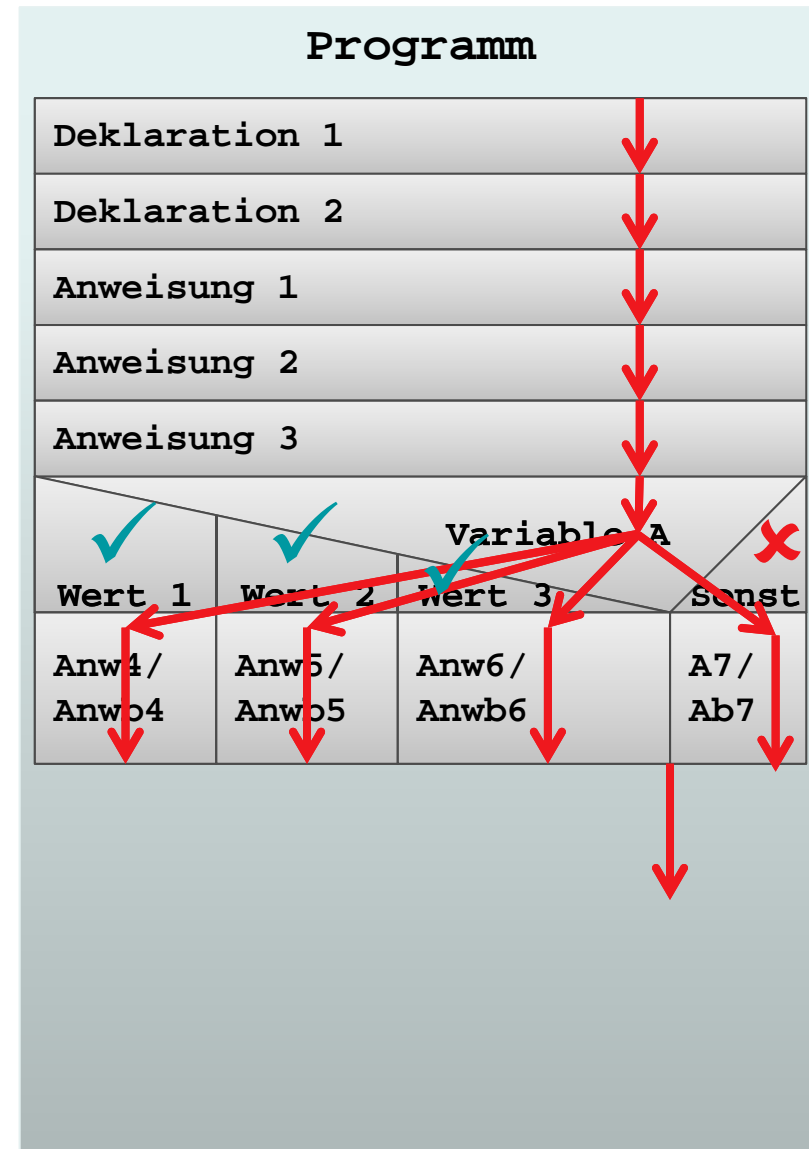


Arten von Verzweigungen



Was tun, wenn

- für *<Anweisung4>* die Variable *A* den Wert 1 haben muss,
- andernfalls für *<Anweisung5>* die Variable *A* den Wert 2 haben muss,
- andernfalls für *<Anweisung6>* die Variable *A* den Wert 3 haben muss,
- und nur sonst *<Anweisung7>* auszuführen ist?



Formulierung von Bedingungen



Bedingungen

- sind Ausdrücke, die zu Wahrheitswert ausgewertet werden
 - True entspricht Wahr, Ja, ...
 - False entspricht Falsch, Nein, ...
- nutzen in der Regel Vergleichsoperatoren und/oder boolesche Operatoren sowie (sinnvollerweise) immer Variablen
- Beispiele

```
5 > 7 ' immer falsch  
5 + 3 = 8 ' immer wahr  
True ' immer wahr
```

```
strName = "Müller"  
curBetrag < 123.45  
i + 5 <= 12
```

Logische Operatoren zur Verknüpfung von Bedingungen

- Nutzung von And, Or, Xor und Not um mehrere Bedingungen zu Verknüpfen (vgl. LE 03 Datentyp Wahrheitswert)

Formulierung von Bedingungen



Beispiele

- Wenn Stammkunde und Einkaufswert > 100 EUR, dann 10% Rabatt, andernfalls wenn Stammkunde und Einkaufswert > 50 EUR dann 5% Rabatt sonst 0% Rabatt
- Wenn Vielflieger und aktueller Flug nicht eingelöster Prämienflug, dann Meilengutschrift.
- Wenn E-Mail-Adresse oder Telefonnummer des Kunden bekannt sind, dann über neue Produkte informieren

```
(bolStKunde And curEkWert > 100)
```

```
(bolStKunde And curEkWert > 50)
```

```
(bolFrqTrvl And Not(bolIsAward))
```

```
(bolHasMail Or bolHasPhone)
```



Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick





Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

- Variante 1 ...
- Variante 2 ...

Verzweigungen mit VBA



Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- **If**
- **Then**
- **Else**
- **ElseIf**
- **End If**

– Variante 2 ...

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	Anw 5/ Anwb 5

Bedingung1 ist wahr	
Ja	Nein
Anw5/ Anwb5	Bedingung2 ist wahr
	Ja Nein
	Anw5/ Anwb5 Anweisung5/ Anweisungsblock5

Verzweigungen mit VBA

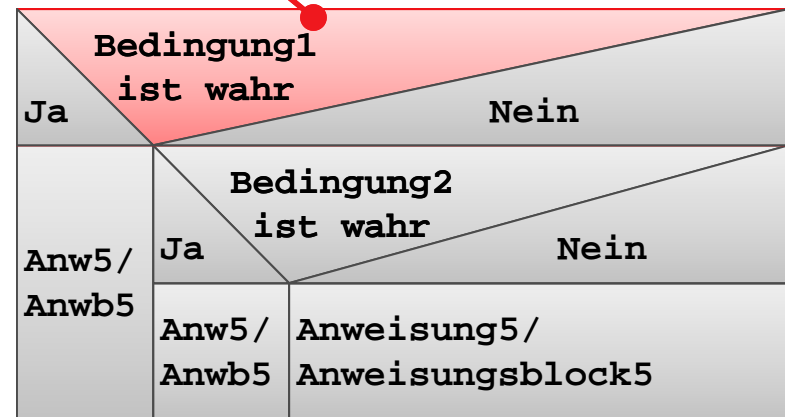
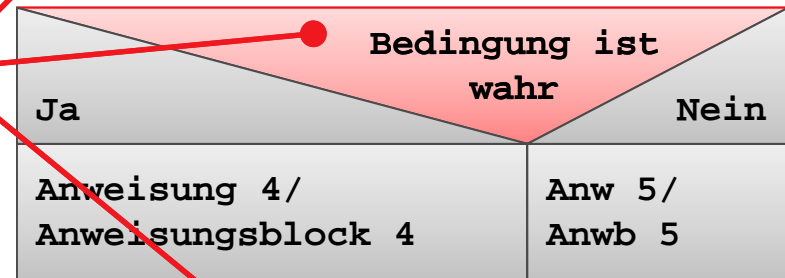
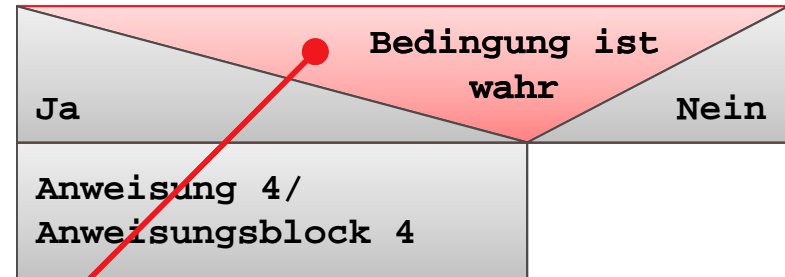


Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- **If**
- **Then**
- **Else**
- **ElseIf**
- **End If**

– Variante 2 ...





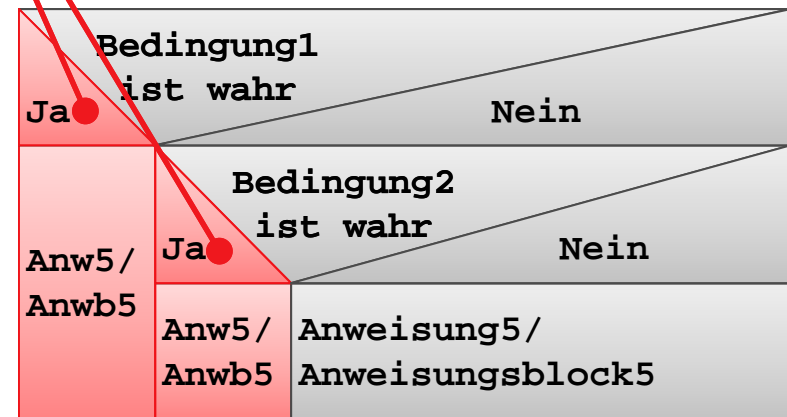
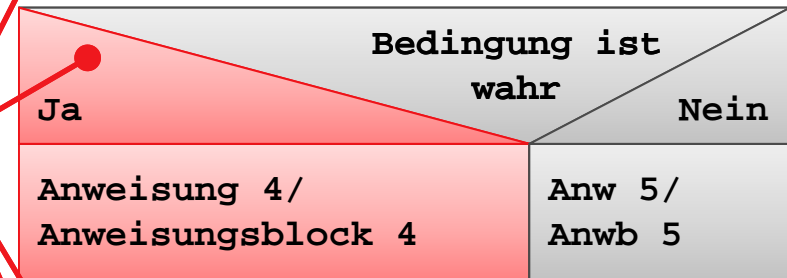
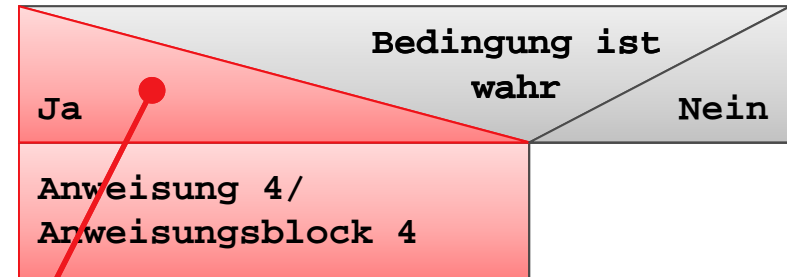
Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- **If**
- **Then**
- **Else**
- **ElseIf**
- **End If**

– Variante 2 ...



Verzweigungen mit VBA



Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- If
- Then
- Else
- ElseIf
- End If

– Variante 2 ...

	Bedingung ist wahr	
Ja		Nein
Anweisung 4/ Anweisungsblock 4		

	Bedingung ist wahr	
Ja		Nein
Anweisung 4/ Anweisungsblock 4	Anw 5/ Anwb 5	

	Bedingung1 ist wahr	
Ja		Nein
Anw5/ Anwb5	Bedingung2 ist wahr	
	Ja	Nein
	Anw5/ Anwb5	Anweisung5/ Anweisungsblock5

Verzweigungen mit VBA



Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- **If**
- **Then**
- **Else**
- **ElseIf**
- **End If**

– Variante 2 ...

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	Anw 5/ Anwb 5

Bedingung1 ist wahr	
Ja	Nein
Anw5/ Anwb5	Bedingung2 ist wahr
	Ja Nein
Anw5/ Anwb5	Anweisung5/ Anweisungsblock5

Verzweigungen mit VBA



Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- **If**
- **Then**
- **Else**
- **ElseIf**
- **End If**

– Variante 2 ...

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	Anw 5/ Anwb 5

Bedingung1 ist wahr	
Ja	Nein
Anw5/ Anwb5	Bedingung2 ist wahr
	Nein
Anw5/ Anwb5	Anweisung5/ Anweisungsblock5

Verzweigungen mit VBA



Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- **If**
- **Then**
- **Else**
- **ElseIf**
- **End If**

– Variante 2 ...

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	Anw 5/ Anwb 5

Bedingung1 ist wahr	
Ja	Nein
Anw5/ Anwb5	Bedingung2 ist wahr
	Ja Nein
Anw5/ Anwb5	Anweisung5/ Anweisungsblock5



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- **If**
- **Then**
- **Else**
- **ElseIf**
- **End If**

– Variante 2 ...

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	Anw 5/ Anwb 5

Bedingung1 ist wahr							
Ja	Nein						
Anw5/ Anwb5	<table border="1"><thead><tr><th colspan="2">Bedingung2 ist wahr</th></tr><tr><th>Ja</th><th>Nein</th></tr></thead><tbody><tr><td>Anw5/ Anwb5</td><td>Anweisung5/ Anweisungsblock5</td></tr></tbody></table>	Bedingung2 ist wahr		Ja	Nein	Anw5/ Anwb5	Anweisung5/ Anweisungsblock5
	Bedingung2 ist wahr						
Ja	Nein						
Anw5/ Anwb5	Anweisung5/ Anweisungsblock5						



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

– Variante 1 mit insgesamt fünf Schlüsselworten

- **If**
- **Then**
- **Else**
- **ElseIf**
- **End If**

– Variante 2 ...

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	Anw 5/ Anwb 5

Bedingung1 ist wahr	
Ja	Nein
Anw5/ Anwb5	Bedingung2 ist wahr
	Ja
Anw5/ Anwb5	Anweisung5/ Anweisungsblock5



Verzweigungen mit VBA: Beispiel 04.03

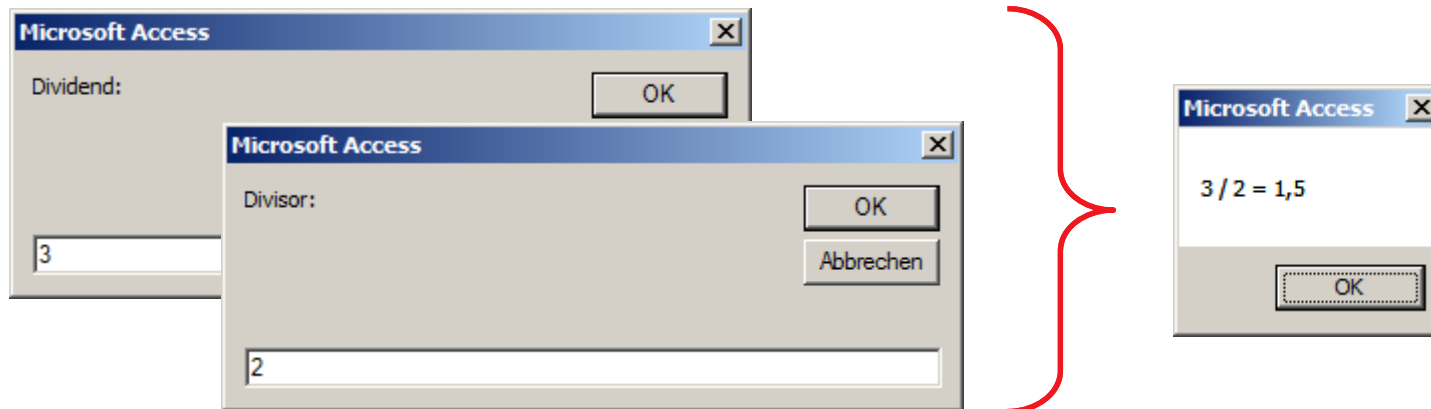
Ziel

- Bedingte Anweisung praktisch anwenden

Aufgabe

- Deklaration von drei Variablen: als Dividend, als Divisor und als Quotient
- Initialisierung von Dividend und Divisor durch Dialogeingabe vom Benutzer
- wenn der Divisor ungleich 0 ist
 - Ausführung der Division und
 - Speicherung des errechneten Quotienten und
 - Ausgabe der drei Variablenwerte als Rechnung in einem einfachen Meldungsfenster

Ergebnis



Verzweigungen mit VBA: Beispiel 04.03



```
Microsoft Visual Basic for Applications - Database11 - [Modul1 (Code)]
Datei Bearbeiten Ansicht Einfügen Debuggen Ausfuehren Extras Add-Ins Fenster ?
Frage hier eingeben

Projekt: (Allgemein) demo0403

Option Explicit

Sub demo0403 ()

    ' Deklaration
    Dim intDividend As Integer
    Dim intDivisor As Integer
    Dim sglQuotient As Single

    ' Initialisierung
    Let intDividend = Val(InputBox("Dividend:"))
    Let intDivisor = Val(InputBox("Divisor:"))

    ' Bedingte Anweisung
    If (intDivisor <> 0) Then
        Let sglQuotient = intDividend / intDivisor
        MsgBox (intDividend & " / " & intDivisor & " = " & sglQuotient)
    End If

End Sub

Direktbereich
```

Verzweigungen mit VBA: Beispiel 04.04



Ziel

- Einfache Verzweigung praktisch anwenden

Aufgabe

- Deklaration einer Variablen und ihre Initialisierung durch Dialogeingabe vom Benutzer
- Wenn eingegebene Zahl gerade ist, dann Ausgabe "<x> ist eine gerade Zahl" im Direktbereich, andernfalls "<x> ist eine ungerade Zahl"

Ergebnis

```
Direktbereich
3 ist eine ungerade Zahl
2 ist eine gerade Zahl
8 ist eine gerade Zahl
15 ist eine ungerade Zahl
```

Verzweigungen mit VBA: Beispiel 04.04



```
Microsoft Visual Basic for Applications - Database11 - [Modul1 (Code)]
Datei Bearbeiten Ansicht Einfügen Debuggen Ausführen Extras Add-Ins Fenster ?
Frage hier eingeben

Projekt - Database11
Database11 (Database11)
  Module
    Modul1

Eigenschaften - Modul1
Modul1 Modul
Alphabetisch Nach Kategorien
(Name) Modul1

(Allgemein) demo0404
Option Compare Database
Option Explicit

Sub demo0404()

  ' Deklaration
  Dim intZahl As Integer

  ' Initialisierung
  Let intZahl = Val(InputBox("Eine Zahl:"))

  ' Verzweigung anhand des Rests der ganzzahligen Division
  If (intZahl Mod 2 = 0) Then
    ' Gerade, weil kein Rest
    Debug.Print intZahl & " ist eine gerade Zahl"
  Else
    ' Ungerade, weil Rest
    Debug.Print intZahl & " ist eine ungerade Zahl"
  End If

Direktbereich
3 ist eine ungerade Zahl
2 ist eine gerade Zahl
8 ist eine gerade Zahl
15 ist eine ungerade Zahl
```

Verzweigungen mit VBA: Beispiel 04.05



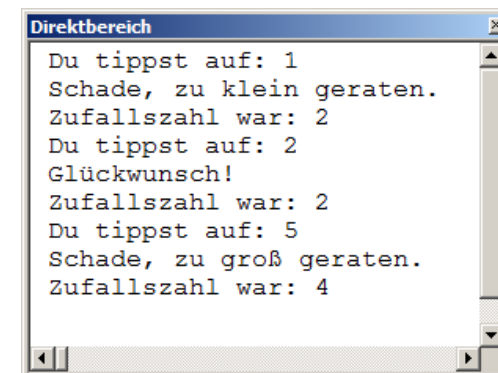
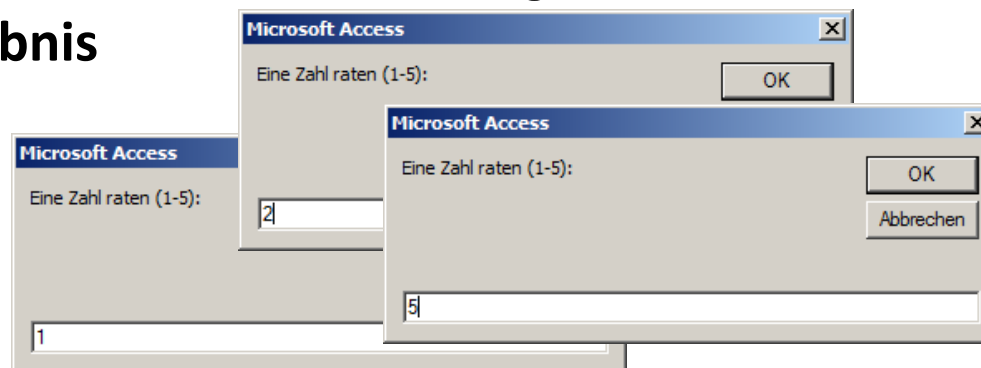
Ziel

- Mehrfachverzweigung praktisch anwenden

Aufgabe

- Zufallszahlen im Bereich von 1 bis 5 ermitteln, indem Sie eine Variable mit folgendem Ausdruck initialisieren:
`CInt(1 + 4 * Rnd())`
- Benutzer soll Zahl erraten und seinen Tipp in einer InputBox eingeben
- Die geratene Zahl soll ausgegeben werden: "Du hast <y> geraten."
- Wenn geratene Zahl
 - kleiner als Zufallszahl, Ausgabe "Schade, zu klein geraten."
 - größer als Zufallszahl, Ausgabe "Schade, zu groß geraten."
 - gleich Zufallszahl, Ausgabe "Glückwunsch!"
- In allen Fällen soll die Ausgabe der Zufallszahl erfolgen "Zufallszahl war: <x>"

Ergebnis



Verzweigungen mit VBA: Beispiel 04.05



```
Microsoft Visual Basic for Applications - beuth_wi_le04-su_grundkonzepte-verzweigungen_v1-0_vl-demo - [Modul1 (Code)]
Datei Bearbeiten Ansicht Einfügen Debuggen Ausführen Extras Add-Ins Fenster ?
Projekt - Database11
Database11 (beuth_wi_le)
  Module
    Modul1
Eigenschaften - Modul1
Modul1 Modul
Alphabetisch Nach Kategorien
(Name) Modul1

Sub demo0405 ()

' Deklaration
Dim bytZufall As Byte
Dim bytZahl As Byte

' Initialisierung
Let bytZufall = CInt(1 + 4 * Rnd())
Let bytZahl = Val(InputBox("Eine Zahl raten (1-5):"))

' Ausgabe der geratenen Zahl
Debug.Print "Du tippst auf: " & bytZahl

' Verzweigung
If bytZahl < bytZufall Then
    Debug.Print "Schade, zu klein geraten."
ElseIf bytZahl > bytZufall Then
    Debug.Print "Schade, zu groß geraten."
Else
    Debug.Print "Glückwunsch!"
End If

' Ausgabe der Zufallszahl
Debug.Print "Zufallszahl war: " & bytZufall

End Sub
```



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

- Variante 1 ...
- Variante 2 ...



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

- Variante 1 ...
- Variante 2 mit insgesamt vier Worten
 - Select Case
 - Case
 - Case Else
 - End Select

und zusätzlichen Ergänzungen

Variable A			
Wert 1	Wert 2	Wert 3	Sonst
Anw4 / Anwb4	Anw5 / Anwb5	Anw6 / Anwb6	A7 / Ab7



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

- Variante 1 ...
- Variante 2 mit insgesamt vier Worten
 - Select Case
 - Case
 - Case Else
 - End Select

und zusätzlichen Ergänzungen

Variable A			
Wert 1	Wert 2	Wert 3	Sonst
Anw4 / Anwb4	Anw5 / Anwb5	Anw6 / Anwb6	A7 / Ab7



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

- Variante 1 ...
- Variante 2 mit insgesamt vier Worten
 - Select Case
 - Case
 - Case Else
 - End Select

und zusätzlichen Ergänzungen

Variable A			
wert 1	wert 2	wert 3	Sonst
Anw4 / Anwb4	Anw5 / Anwb5	Anw6 / Anwb6	A7 / Ab7



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

- Variante 1 ...
- Variante 2 mit insgesamt vier Worten
 - Select Case
 - Case
 - Case Else
 - End Select

und zusätzlichen Ergänzungen

Variable A			
Wert 1	Wert 2	Wert 3	Sonst
Anw4 / Anwb4	Anw5 / Anwb5	Anw6 / Anwb6	A7 / Ab7



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

- Variante 1 ...
- Variante 2 mit insgesamt vier Worten
 - Select Case
 - Case
 - Case Else
 - End Select

und zusätzlichen Ergänzungen

Variable A			
Wert 1	Wert 2	Wert 3	Sonst
Anw4 / Anwb4	Anw5 / Anwb5	Anw6 / Anwb6	A7 / Ab7



Verzweigungen mit VBA: Beispiel 04.06

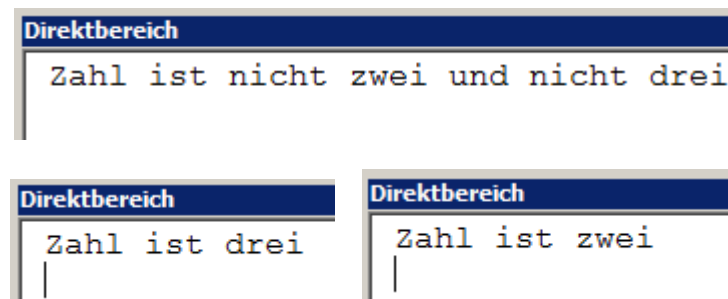
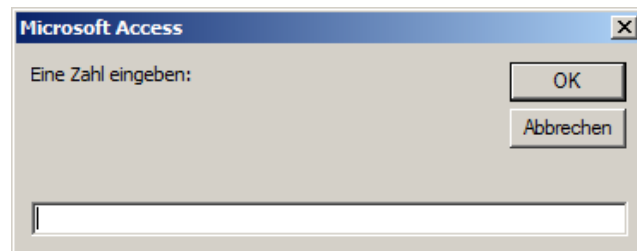
Ziel

- Verzweigungen mit Select Case praktisch anwenden

Aufgabe

- Prüfen, ob eine Variable einen bestimmten Wert oder einen anderen Wert hat oder nicht
 - Ist der Variablenwert gleich zwei?
 - Ist der Variablenwert gleich drei?
- Sie Verzweigungen vom Typ Select Case, um das Ergebnis im Direktbereich auszugeben

Ergebnis



Verzweigungen mit VBA: Beispiel 04.06

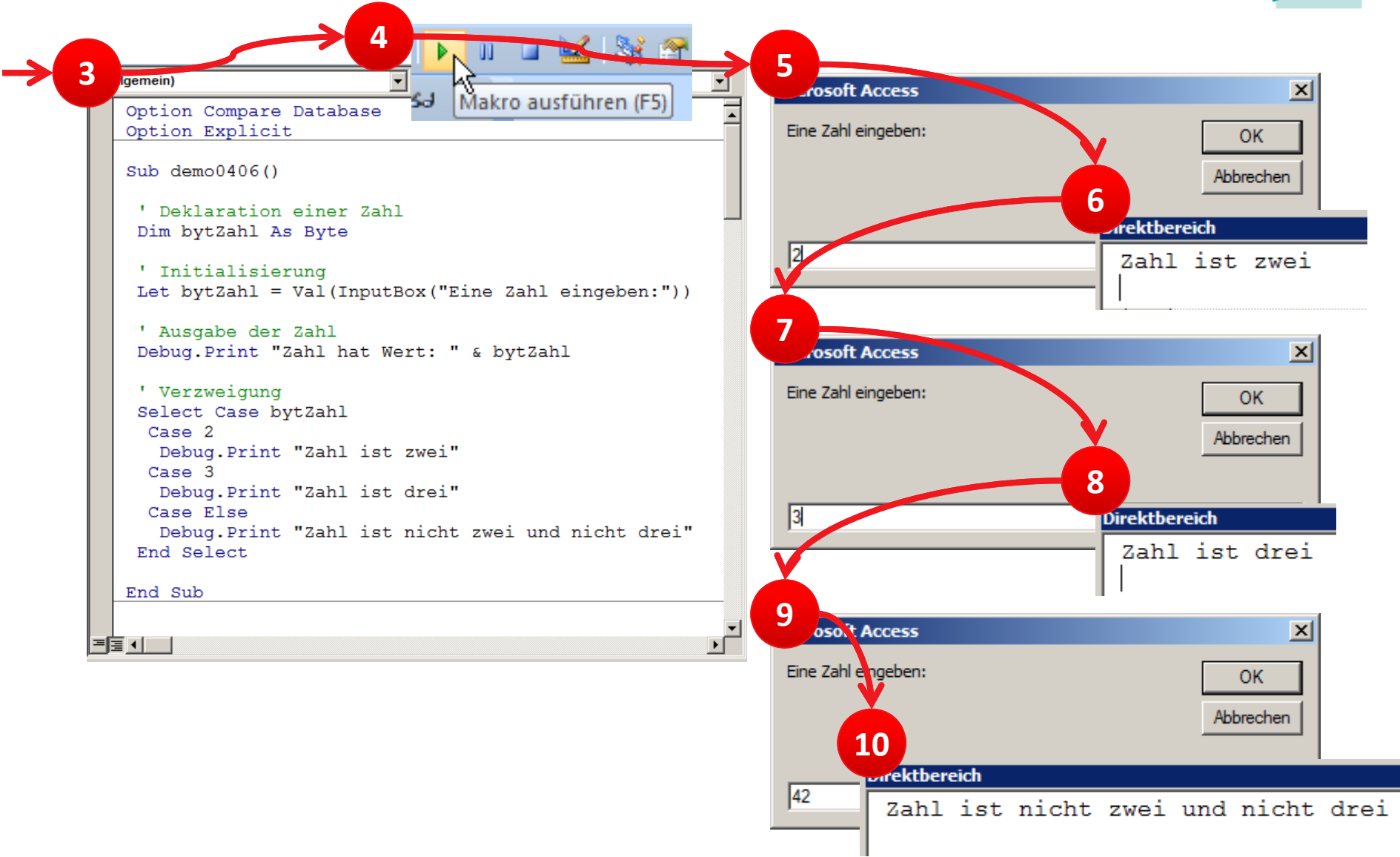


1 → 2

Zahl As ganze Zahl		
Zahl		
Zahl		
2	3	Sonst
"zwei"	"drei"	"nicht zwei und nicht drei"

```
Sub demo0406()  
  
' Deklaration einer Zahl  
Dim bytZahl As Byte  
  
' Initialisierung  
Let bytZahl = Val(InputBox("Zahl eingeben:"))  
  
' Ausgabe der Zahl  
Debug.Print "Zahl hat Wert: " & bytZahl  
  
' Verzweigung  
Select Case bytZahl  
Case 2  
    Debug.Print "Zahl ist zwei"  
Case 3  
    Debug.Print "Zahl ist drei"  
Case Else  
    Debug.Print "Zahl nicht zwei und nicht drei"  
End Select  
  
End Sub
```

Verzweigungen mit VBA: Beispiel 04.06





Verzweigungen mit VBA: Beispiel 04.06

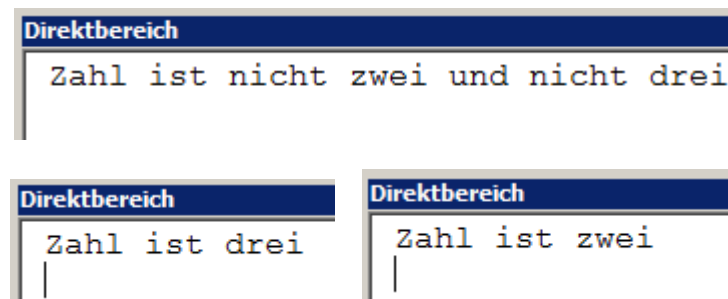
Ziel

- Verzweigungen mit Select Case praktisch anwenden

Aufgabe

- Prüfen, ob eine Variable einen bestimmten Wert oder einen anderen Wert hat oder nicht
 - Ist der Variablenwert gleich zwei?
 - Ist der Variablenwert gleich drei?
- Sie Verzweigungen vom Typ Select Case, um das Ergebnis im Direktbereich auszugeben

Ergebnis



Verzweigungen mit VBA: Beispiel 04.07



Ziel

- Verzweigungen mit Select Case praktisch anwenden und weitere Möglichkeiten nutzen

Aufgabe

- Beispiel 04.06 ändern, so dass folgende Möglichkeiten von Case Select genutzt werden
 - Aufzählungen von Werten
 - Zahlenbereiche
- Prüfung und Ausgaben, ob
 - Zahl zwischen 1 und 3
 - Zahl 4 oder 5
 - Sonstige Zahl

Verzweigungen mit VBA: Beispiel 04.07



```
Sub demo0407()  
    ' Deklaration einer Zahl  
    Dim bytZahl As Byte  
  
    ' Initialisierung  
    Let bytZahl = Val(InputBox("Eine Zahl eingeben:"))  
  
    ' Ausgabe der Zahl  
    Debug.Print "Zahl hat Wert: " & bytZahl  
  
    ' Verzweigung  
    Select Case bytZahl  
        Case 1 To 3  
            Debug.Print "Zahl ist ein, zwei oder drei"  
        Case 4, 5  
            Debug.Print "Zahl ist vier oder fünf"  
        Case Else  
            Debug.Print "Zahl ist nicht eins bis fünf"  
    End Select  
  
End Sub
```

The screenshot shows the VBA editor with the code above. An InputBox dialog box is open, titled "Microsoft Access", with the text "Eine Zahl eingeben:". The input field contains the number "2". Below the dialog box, a message box displays "Zahl ist ein, zwei oder drei". Red circles and arrows indicate the execution flow: 1 points to the start of the subroutine, 2 points to the Select Case statement, 3 points to the InputBox dialog, and 4 points to the output message.

Verzweigungen mit VBA: Beispiel 04.07



Ziel

- Verzweigungen mit Select Case praktisch anwenden und weitere Möglichkeiten nutzen

Aufgabe

- Beispiel 04.06 ändern, so dass folgende Möglichkeiten von Case Select genutzt werden
 - Aufzählungen von Werten
 - Zahlenbereiche
- Prüfung und Ausgaben, ob
 - Zahl zwischen 1 und 3
 - Zahl 4 oder 5
 - Sonstige Zahl

Verzweigungen mit VBA: Beispiel 04.08



Ziel

- Verzweigungen mit Select Case praktisch anwenden und weitere Möglichkeiten nutzen

Aufgabe

- Beispiel 04.07 ändern, so dass die folgende Möglichkeit von Case Select genutzt werden
 - Formulierung einer Bedingung mit Bezug zum Wert der Variable
- Prüfung und Ausgaben, ob
 - Zahl kleiner vier
 - Zahl kleiner sechs
 - Sonstige Zahl

Verzweigungen mit VBA: Beispiel 04.08



The screenshot shows the VBA editor in Microsoft Access. On the left, a code window contains the following VBA code for a subprocedure named `demo0407`:

```
Sub demo0407()  
    ' Deklaration einer Zahl  
    Dim bytZahl As Byte  
  
    ' Initialisierung  
    Let bytZahl = Val(InputBox(  
  
    ' Ausgabe der Zahl  
    Debug.Print "Zahl hat Wert:  
  
    ' Verzweigung  
    Select Case bytZahl  
        Case Is < 4  
            Debug.Print "Zahl ist kle  
        Case Is < 6  
            Debug.Print "Zahl ist kle  
        Case Else  
            Debug.Print "Zahl ist grö  
    End Select  
End Sub
```

On the right, another code window shows the same code, but with the `Case Is < 4` and `Case Is < 6` lines highlighted with red boxes. An input dialog box titled "Microsoft Access" is open, with the text "Eine Zahl eingeben:" and a text box containing the number "2". The dialog has "OK" and "Abbrechen" buttons. Below the dialog, the output window shows the text "Zahl ist kleiner vier".

Four red circles with numbers 1, 2, 3, and 4 are placed over the code and dialog. Red arrows indicate the flow: from circle 1 to the start of the code, from circle 2 to the `InputBox` function, from circle 3 to the dialog box, and from circle 4 to the output window.

Verzweigungen mit VBA: Beispiel 04.08



Ziel

- Verzweigungen mit Select Case praktisch anwenden und weitere Möglichkeiten nutzen

Aufgabe

- Beispiel 04.07 ändern, so dass die folgende Möglichkeit von Case Select genutzt werden
 - Formulierung einer Bedingung mit Bezug zum Wert der Variable
- Prüfung und Ausgaben, ob
 - Zahl kleiner vier
 - Zahl kleiner sechs
 - Sonstige Zahl



Verzweigungen mit VBA

Prüfen von Bedingungen und Festlegen von Anweisungen

- Variante 1 ...
- Variante 2 ...



Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick



Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick





Inhalt

Rückblick

Ausgabe und Eingabe

- Ausgabe im Direktbereich
- Ausgabe und Eingabe mit Dialogen

Konzepte für Verzweigungen

- Arten von Verzweigungen
- Formulierung von Bedingungen

Implementierung von Verzweigungen mit VBA

- Einfachverzweigung
- Mehrfachverzweigung

Abschluss und Ausblick

Abschluss



Eingabe und Ausgabe

– Eingabe mittels Dialog

```
' Generelle Syntax:  
' Let <Variable-vom-Typ-String> = InputBox("<Meldungstext>")  
  
' Beispiele  
Let strName = InputBox("Ihr Name:")  
Let strVorname = InputBox("Ihr Vorname:")
```

– Ausgabe im Debug-Bereich

```
' Beispiel  
Debug.Print "Hallo Welt!"
```

– Ausgabe im Meldungsfenster

```
' Generelle Syntax: MsgBox ("<Meldungstext>")  
' Beispiel:  
MsgBox ("Hallo Welt! Klicke auf OK.")
```



Abschluss

Ausgangspunkt

- Beschränkung bisheriger Programme auf linearen Programmablauf
- Notwendigkeit zu Verzweigungen

Konzepte

- Arten von Verzweigungen, z.B.

Bedingung ist wahr	
Ja	Nein
Anweisung 4/ Anweisungsblock 4	Anw 5/ Anwb 5

Variable A			
Wert 1	Wert 2	Wert 3	Sonst
Anw4/ Anwb4	Anw5/ Anwb5	Anw6/ Anwb6	A7/ Ab7

- Formulierung von Bedingungen für Verzweigungen
 - als Ausdrücke, in der Regel mit Vergleichsoperator
 - mit Wahrheitswert als Ergebnis der Auswertung
 - Einsatz logischer Operationen

Abschluss



Verzweigungen mit VBA

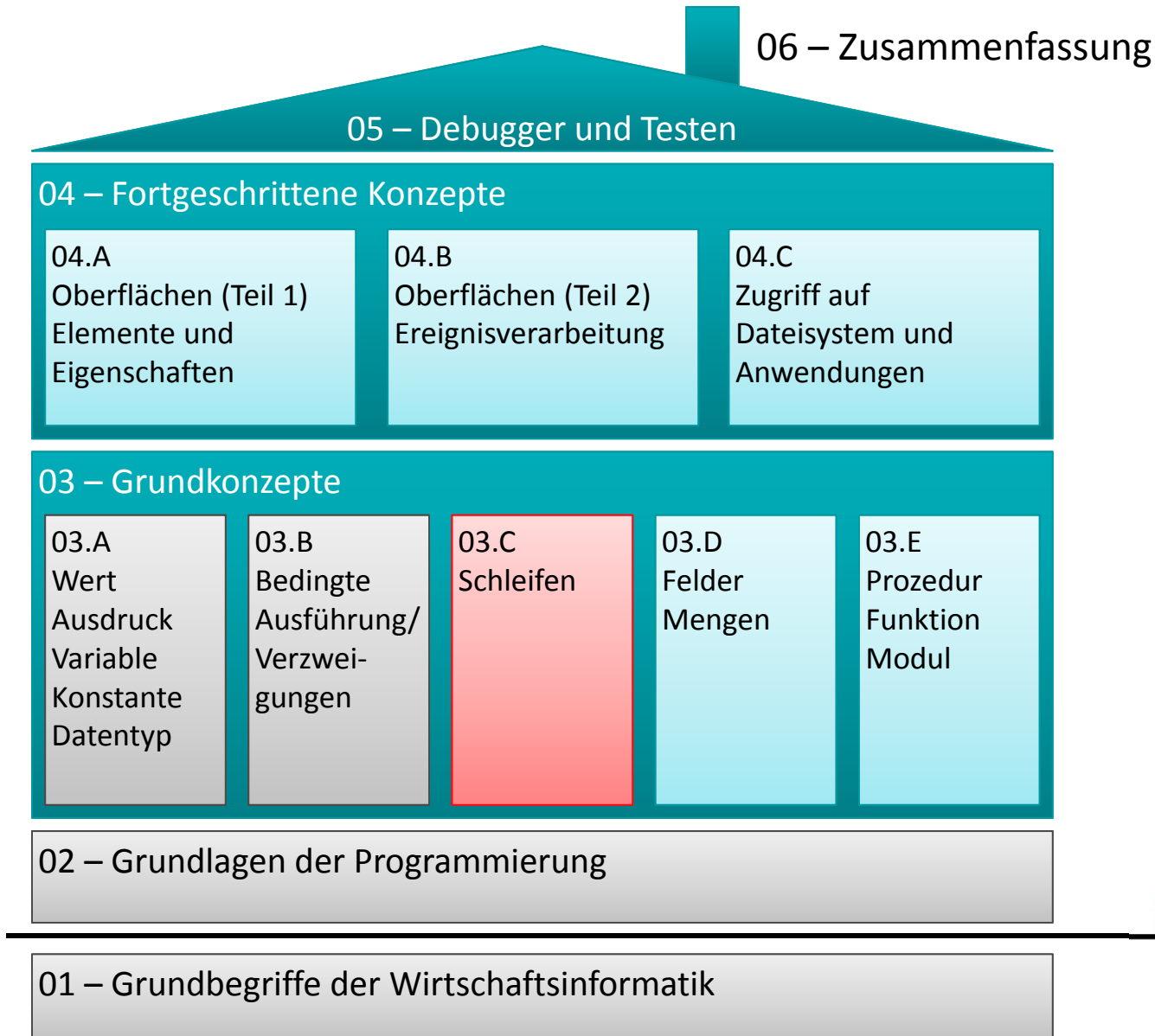
- Einfach Verzweigungen in Form von If-Then-Else-End If
- Mehrfach Verzweigungen
 - Elself-Erweiterung
 - Select-Case-Anweisung

```
' Beispiel  
If intZahl > 4 Then  
    Debug.Print "Größer 4"  
Else  
    Debug.Print "Kleiner 5"  
End If
```

```
' Beispiel  
Select Case intZahl  
Case 1  
    Debug.Print "Eins"  
Case 2,3  
    Debug.Print "Zwei oder Drei"  
Case Else  
    Debug.Print "Etwas anderes"  
End Select
```

```
' Beispiel  
If intZahl > 4 Then  
    Debug.Print "Größer 4"  
ElseIf intZahl = 4 Then  
    Debug.Print "Gleich 4"  
Else  
    Debug.Print "Kleiner 4"  
End If
```

Ausblick





BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Wirtschaftsinformatik 1

LE 04 – Verzweigungen, Ein-/Ausgabe

Prof. Dr. Thomas Off

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