```
'Die folgenden 3 Variablen ermöglichen die zufällige Auswahls des nächsten Spielsteins sowie die
'Anzeige dessen, während man noch mit dem vorherigen Spielstein spielt.
 Dim neueZufallsfigur As Integer 'wird ständig im Timer Zufallsgenerator durchgezählt
 Dim NextFigur As Integer        'hat den Wert des "Warteschlangensteins", der rechts angezeigt wird
 Dim Zufallsfigur As Integer     'hat den Wert des aktuellen Spielsteins

'Die folgenden 2 Variablen dienen der Unterscheidung, ob eine horizontale Verschiebung ausgeführt
'wird oder nicht.
 Dim Linksbewegung As Integer 'kann 0 oder 1 annehmen (Verschiebung um eine Spalte bzw. keine Versch.)
 Dim Rechtsbewegung As Integer 'kann 0 oder 1 annehmen (Verschiebung um eine Spalte bzw. keine Versch.)

'Ein Spielstein enthält 4 Kästen. Jeder Kasten ist Teil eines Steuerelementarrays "Figur() as Shape"
'Die Position dieser Shapes hat jedoch unhandliche Werte im 3-4-Stelligen Bereich. Um diese weitest-
'gehend zu vermeiden, habe ich parallel dazu folgende Integerarrays deklariert, von denen die
'einzelnen Werte für KastenX zwischen 1 und 12 und für KastenY zwischen 0 und 20 liegen.
 Dim KastenX(0 To 100000) As Integer  'X-Position des Kastens als Bestandteil des Spielsteins
 Dim KastenY(0 To 100000)  As Integer 'Y-Position des Kastens als Bestandteil des Spielsteins

'Die folgenden Arrays dienen dazu, die Position eines Steins abzuspeichern, bevor er auf einen schon
'bestetzten Platz gefallen ist, damit man diese Position wiederherstellen kann.
 Dim KastenXDAVOR(0 To 3) As Integer '(0 To 3)-> 4 Elemente, da ein Stein 4 Kästen enthält
 Dim KastenYDAVOR(0 To 3)  As Integer 'Sowohl X- als auch Y-Position

'Das folgende Array stellt das Spielfeld dar und gibt an, ob sich an einem Platz schon ein Stein
'befindet oder nicht. Das Spielfeld ist 12 Felder breit und 20 Felder hoch, der Rand beträgt überall
'außer unten ein Feld, damit unten bei einer Drehung des länglichen Steins der Index nicht außerhalb
'des gültigen Bereichs sein kann. Links tritt dieses Problem auch auf, jedoch muss es dort anders
'gelöst werden, weil der Index links bis zu -1 sein müsste.
 Dim Besetzt(0 To 13, 0 To 22) As Boolean

 Dim Position As Integer '4 verschiedene Positionen, da Drehung der Steine möglich
 Dim ReiheVoll As Integer 'wird bis 12 gezählt; zum Überprüfen, ob eine Reihe vervollständigt wurde
 Dim AnzahlReihen As Integer 'wird bis 4 gezählt; Anzahl der Reihen, die in einem Zug vervollständigt wurden
 Dim Linien As Integer 'Anzahl der bereits vervollständigten horizontalen Linien/Reihen in einem Level
 Dim Level As Integer 'Level, in dem man sich gerade befindet (Steine fallen mit höherem Level schneller)
 Dim Zufallssound As Integer 'Je nach zufällig ausgewähltem Wert wird ein anderer Sound abgespielt

'Für das Einspielen von kurzen Sounds während des Spiels habe ich folgende Funktion aus dem Windows-
'Betriebssystem übernommen. Man muss unterscheiden zwischen den verschiedenen punktuell eingespielten
'Sounds und der möglicherweise ständig laufenden Tetrismelodie im Windows-Media-Player. Diese beiden
'Dinge haben nichts miteinander zu tun.
 Private Declare Function sndPlaySound Lib "Winmm.dll" Alias _
 "sndPlaySoundA" (ByVal lpszSoundName As String, ByVal uFlags As Long) As Long

'Startbutton:
'Auf dem Platz des Startbuttons befindet sich auch der Button cmdWeiter, jedoch ist je nach Bedarf immer
'nur einer von beiden sichtbar.
'Der Startbutton dient zum Starten des Spiels. Beendet wird das Spiel ausschließlich durch Verlieren.
'Pausen gibt es entweder beim Erreichen des nächsten Levels (dann wird allerdings mit cmdWeiter
'fortgefahren) oder mit der Taste P (für Pause)
 Private Sub cmdStart_Click()

    cmdStart.Enabled = False 'Ein Spiel kann nicht doppelt gestarten werden

   'Erneutes Initialisieren nach vorhergegangenem Spiel
    If lblVerloren.Visible = True Then
       lblVerloren.Visible = False
       Timer1.Interval = 600
       Linien = 0
    End If

    If Not (Timer1.Enabled) Then
       Level = 1

       For x = 1 To 12
          For y = 1 To 20
             Besetzt(x, y) = False '"Freimachen" des gesamten Spielfeldes
          Next
       Next

       For x = 1 To 12
          Besetzt(x, 21) = True 'Herstellen des Bodens
          Besetzt(x, 0) = False 'Freimachen der Reihe über dem Spielfeld,
                                'sodass Steine schon im ersten Intervall gedreht werden können
       Next

       For y = 1 To 21
          Besetzt(0, y) = True 'Herstellen des Rands
          Besetzt(13, y) = True 'Herstellen des Rands
```

```
        Next

            Call NeueFigur 'Ein neuer Stein fällt von oben herunter

            Timer1.Enabled = True
    End If

    Text1.SetFocus 'Um die Eingabe von Befehlen zum Drehen etc. möglich zu machen
End Sub

'Weiterbutton:
'Falls das nächste Level erreicht wurde, wird er sichtbar und verdeckt den Startbutton.
'Er dient zum Fortfahren.
Private Sub cmdWeiter_Click()

    Timer1.Enabled = True
    lblVerloren.Visible = False
    cmdWeiter.Visible = False
    Call NeueFigur
    Text1.SetFocus
End Sub


'Infobutton
'Hiermit wird lediglich eine Messagebox angezeigt, die die für die Bedienung des Spiels wichtigen Befehle
'anzeigt. Falls das Spiel beim Klick auf den Button noch am laufen war, wird es unterbrochen.
Private Sub Command1_Click()

    'Anhalten des Spiels, sodass keine Steine während der Msgbox Anzeige unkontrolliert herunterfallen können
    If Timer1.Enabled = True Then
        Timer1.Enabled = False
        lblPause.Visible = True
    End If

    MsgBox ("W = Drehen, ASD = Steuern, P = Pause")
    Text1.SetFocus 'Erneutes Drücken auf P zum Beenden der Pause muss möglich sein
End Sub


Private Sub Form_Click()

    Text1.SetFocus 'Im Zweifelsfall muss der Focus immer auf dem Eingabetextfeld liegen,
                'sodass man die herunterfallenden Steine steuern kann
End Sub


Private Sub Form_Load()

    'Initialisierungsbefehle
    neueZufallsfigur = 0
    NextFigur = 5
    Linien = 0
    Level = 1
    Zufallssound = 0

    'Laden der Tetrismelodie aus dem Ordner, in dem sich auch die EXE-Datei befindet.
    'Sie wird mit dem Windows Media Player rechts am Rand wiedergegeben.
    'Der Windows Media Player ist ein Add-in-Steuerelement, das extra geladen werden musste.
    Jaa = App.Path
    If Right(Dateiname, 1) <> "\" Then Jaa = Jaa & "\"
    Player.URL = Jaa & "Musik.wav"
End Sub

Public Function NeueFigur()

    '4 neue Kästen werden während der Laufzeit erstellt und weiter unten zu einem Stein angeordnet.
    'Figur.UBound gibt den derzeit größten Index des Steuerelementarrays Figur() an.
    'Somit erweitert sich das Steuerelementarray sinnvoll um die die nächsten 4 Elemente.
    For a = 0 To 3
        Load Figur(Figur.UBound + 1)
    Next

    Zufallsfigur = NextFigur 'Übernehmen der Warteschlangenfigur

    Select Case Zufallsfigur
    Case 1
        'Anordnen der neu erstellten Kästen zu einem Stein mit bestimmter Form oben in der Mitte des Spielfelds
        'Der Wert der Variable Zufallsfigur bestimmt die Form des Steins
        KastenX(Figur.UBound - 0) = 5
        KastenY(Figur.UBound - 0) = 1
```

```
        KastenX(Figur.UBound - 1) = 6
        KastenY(Figur.UBound - 1) = 1
        KastenX(Figur.UBound - 2) = 6
        KastenY(Figur.UBound - 2) = 2
        KastenX(Figur.UBound - 3) = 7
        KastenY(Figur.UBound - 3) = 2

    Case 2
        KastenX(Figur.UBound - 0) = 5
        KastenY(Figur.UBound - 0) = 2
        KastenX(Figur.UBound - 1) = 6
        KastenY(Figur.UBound - 1) = 2
        KastenX(Figur.UBound - 2) = 6
        KastenY(Figur.UBound - 2) = 1
        KastenX(Figur.UBound - 3) = 7
        KastenY(Figur.UBound - 3) = 1

        For a = 0 To 3
            Figur(Figur.UBound - a).FillColor = 255 'Jede Steinform hat ihre eigene Farbe
        Next

    Case 3
        KastenX(Figur.UBound - 0) = 6
        KastenY(Figur.UBound - 0) = 1
        KastenX(Figur.UBound - 1) = 5
        KastenY(Figur.UBound - 1) = 2
        KastenX(Figur.UBound - 2) = 6
        KastenY(Figur.UBound - 2) = 2
        KastenX(Figur.UBound - 3) = 7
        KastenY(Figur.UBound - 3) = 2

        For a = 0 To 3
            Figur(Figur.UBound - a).FillColor = 6299648 'Jede Steinform hat ihre eigene Farbe
        Next

    Case 4

        KastenX(Figur.UBound - 0) = 6
        KastenY(Figur.UBound - 0) = 1
        KastenX(Figur.UBound - 1) = 7
        KastenY(Figur.UBound - 1) = 1
        KastenX(Figur.UBound - 2) = 6
        KastenY(Figur.UBound - 2) = 2
        KastenX(Figur.UBound - 3) = 7
        KastenY(Figur.UBound - 3) = 2

        For a = 0 To 3
            Figur(Figur.UBound - a).FillColor = 331549 'Jede Steinform hat ihre eigene Farbe
        Next

    Case 5

        KastenX(Figur.UBound - 0) = 5
        KastenY(Figur.UBound - 0) = 1
        KastenX(Figur.UBound - 1) = 6
        KastenY(Figur.UBound - 1) = 1
        KastenX(Figur.UBound - 2) = 7
        KastenY(Figur.UBound - 2) = 1
        KastenX(Figur.UBound - 3) = 8
        KastenY(Figur.UBound - 3) = 1

        For a = 0 To 3
            Figur(Figur.UBound - a).FillColor = 4173311 'Jede Steinform hat ihre eigene Farbe
        Next


    Case 6

        KastenX(Figur.UBound - 0) = 5
        KastenY(Figur.UBound - 0) = 2
        KastenX(Figur.UBound - 1) = 6
        KastenY(Figur.UBound - 1) = 2
        KastenX(Figur.UBound - 2) = 7
        KastenY(Figur.UBound - 2) = 2
        KastenX(Figur.UBound - 3) = 7
        KastenY(Figur.UBound - 3) = 1

        For a = 0 To 3
            Figur(Figur.UBound - a).FillColor = 15773696 'Jede Steinform hat ihre eigene Farbe
        Next
```

```
    Case 7

        KastenX(Figur.UBound - 0) = 5
        KastenY(Figur.UBound - 0) = 2
        KastenX(Figur.UBound - 1) = 6
        KastenY(Figur.UBound - 1) = 2
        KastenX(Figur.UBound - 2) = 7
        KastenY(Figur.UBound - 2) = 2
        KastenX(Figur.UBound - 3) = 5
        KastenY(Figur.UBound - 3) = 1

        For a = 0 To 3
            Figur(Figur.UBound - a).FillColor = 55535 'Jede Steinform hat ihre eigene Farbe
        Next

    End Select

    For a = 0 To 3 'Sichtbarmachen des neu erstellten Steins
        Figur(Figur.UBound - a).Visible = True
    Next

    Position = 1 'Anfangswert für spätere Drehungen

    NextFigur = neueZufallsfigur 'übernehmen eines zufälligen Werts für die Warteschlangenfigur

    'Je nachdem, welche Figur als nächstes drankommt, muss die Anzeige rechts in der Mitte umgestellt werden
    'Es existieren alle 7 verschiedenen Steine als Steuerelementarrays "Zahl(0 to 3)", jedoch ist immer
    'nur eins von ihnen sichtbar.
    Select Case NextFigur
    Case 1
    For a = 0 To 3
        Eins(a).Visible = True
        Zwei(a).Visible = False
        Drei(a).Visible = False
        Vier(a).Visible = False
        Fünf(a).Visible = False
        Sechs(a).Visible = False
        Sieben(a).Visible = False
    Next

    Case 2
    For a = 0 To 3
        Eins(a).Visible = False
        Zwei(a).Visible = True
        Drei(a).Visible = False
        Vier(a).Visible = False
        Fünf(a).Visible = False
        Sechs(a).Visible = False
        Sieben(a).Visible = False
    Next

    Case 3
    For a = 0 To 3
        Eins(a).Visible = False
        Zwei(a).Visible = False
        Drei(a).Visible = True
        Vier(a).Visible = False
        Fünf(a).Visible = False
        Sechs(a).Visible = False
        Sieben(a).Visible = False
    Next

    Case 4
    For a = 0 To 3
        Eins(a).Visible = False
        Zwei(a).Visible = False
        Drei(a).Visible = False
        Vier(a).Visible = True
        Fünf(a).Visible = False
        Sechs(a).Visible = False
        Sieben(a).Visible = False
    Next

    Case 5
    For a = 0 To 3
        Eins(a).Visible = False
        Zwei(a).Visible = False
        Drei(a).Visible = False
        Vier(a).Visible = False
```

```
        Fünf(a).Visible = True
        Sechs(a).Visible = False
        Sieben(a).Visible = False
    Next

    Case 6
    For a = 0 To 3
        Eins(a).Visible = False
        Zwei(a).Visible = False
        Drei(a).Visible = False
        Vier(a).Visible = False
        Fünf(a).Visible = False
        Sechs(a).Visible = True
        Sieben(a).Visible = False
    Next

    Case 7
    For a = 0 To 3
        Eins(a).Visible = False
        Zwei(a).Visible = False
        Drei(a).Visible = False
        Vier(a).Visible = False
        Fünf(a).Visible = False
        Sechs(a).Visible = False
        Sieben(a).Visible = True
    Next
    End Select

    Call Darstellung
End Function

'Timer Reihe:
'Dieser Timer wird aktiviert, wenn Timer1, also der Spieltimer, bei Vervollständigung einer Reihe
'ausgeschaltet wurde. Jedoch schaltet sich der Timer "Reihe" beim ersten Ereignis sofort selbst aus
'und er schaltet den Spieltimer wieder an. Somit steht das Intervall des Timers "Reihe" für die Länge
'der Pause nach der Vervollständigung einer/mehrer Reihe/n
Private Sub Reihe_Timer()
    imgBear.Visible = False
    Timer1.Enabled = True
    Reihe.Enabled = False
End Sub

'Eingabetextfeld:
'Die Steuerung des Spiels läuft über dieses Textfeld. Während der Spielzeit hat es immer den Focus, sodass die
'Steuerung immer möglich ist. Eine Ausnahme ist das Bedienen des Windows Media Players. Hiernach kann man jedoch
'irgendwo auf das Formular klicken und das Textfeld bekommt wieder den Focus (dank Form_Click).
'Die automatisch in dieses Sub übergebene Variable KeyAscii enthält den Asciicode des zuletzt eingegebenen Zeichens.
'Hieraus kann man ableiten, ob w,a,s,d oder p gedrückt wurde und demnach kann man die richtige Operation ausführen.
Private Sub Text1_KeyPress(KeyAscii As Integer)

If Timer1.Enabled Then

   'falls die Bewegung in die Hose geht, muss die vorherige Position gespeichert werden,
   'um sie evt. wiederherzustellen
    For a = 0 To 3
        KastenXDAVOR(a) = KastenX(Figur.UBound - a)
        KastenYDAVOR(a) = KastenY(Figur.UBound - a)
    Next

    If (KeyAscii = 119) Then 'W zum Drehen des Steins

        Position = Position + 1 'Rotieren
        If Position = 5 Then 'Es gibt nur 4 Positionen/Anordnungen
            Position = 1
        End If


        Select Case Zufallsfigur 'Hat noch den Wert des Aktuellen Spielsteins, der gedreht werden muss
        Case 1
            Select Case Position 'Ist nötig, um zu wissen, wie man den Stein drehen muss

           'Neuanordnung der Kästen, sodass der Stein am Ende "gedreht" ist
            Case 2
                KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) - 2
                KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) - 2
            Case 3
                KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) + 2
                KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) + 2
            Case 4
                KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) - 2
```

```
            KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) - 2
        Case 1
            KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) + 2
            KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) + 2
        End Select

    Case 2
        Select Case Position
        Case 2
            KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) - 2
            KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) + 2
        Case 3
            KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) + 2
            KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) - 2
        Case 4
            KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) - 2
            KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) + 2
        Case 1
            KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) + 2
            KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) - 2
        End Select

    Case 3
        Select Case Position
        Case 2
            KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) + 1
            KastenX(Figur.UBound - 1) = KastenX(Figur.UBound - 1) + 1
        Case 3
            KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) + 1
            KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) - 1
        Case 4
            KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) - 1
            KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) - 1
        Case 1
            KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) - 1
            KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) + 1
            KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) - 1
            KastenX(Figur.UBound - 1) = KastenX(Figur.UBound - 1) - 1
            KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) + 1
            KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) + 1

        End Select

    Case 5 'länglicher Stein

        If KastenX(Figur.UBound) = 1 And (Position = 1 Or Position = 3) Then
            'Sonderfall vom Rückgängig-Machen eines Zuges:
            'Falls der längliche Stein parallel zur linken Seitenwand ist, direkt an ihr liegt und
            'gedreht wird, so ist das nicht möglich, da der Stein, der sich anschließend horizontal
            'am weitesten links befände, dann außerhalb des Bereichs des Arrays Besetzt(,) wäre und
            'eine folgende Abfrage eine Fehlermeldung bringen würde.
            'Der Wert 0 für die X-Koordinate von Besetzt(,) gilt nämlich schon für die Spalte direkt
            'neben dem Spielfeld. Auf der linken Seite wäre aber dann immer noch ein Kasten bei der
            'Koordinate -1, die nicht existiert. Dieser Fall wird mit dieser Verzweigung ausgeschlossen.

            Position = 4 'vertikale Position wieder einnehmen
        Else

            Select Case Position
            Case 2
                KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) + 2
                KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) + 2
                KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) - 1
                KastenX(Figur.UBound - 1) = KastenX(Figur.UBound - 1) + 1
                KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) + 1
                KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) - 1

            Case 3
                KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) - 2
                KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) - 2
                KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) + 1
                KastenX(Figur.UBound - 1) = KastenX(Figur.UBound - 1) - 1
                KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) - 1
                KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) + 1

            Case 4
                KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) + 2
                KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) + 2
                KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) - 1
                KastenX(Figur.UBound - 1) = KastenX(Figur.UBound - 1) + 1
```

```
            KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) + 1
            KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) - 1

        Case 1
            KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) - 2
            KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) - 2
            KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) + 1
            KastenX(Figur.UBound - 1) = KastenX(Figur.UBound - 1) - 1
            KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) - 1
            KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) + 1
        End Select
    End If

Case 6
    Select Case Position
    Case 2
        KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) - 2
        KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) + 1
        KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) - 1

    Case 3
        KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) + 1
        KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) - 1
        KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) - 1
        KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) - 1
        KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) + 1
        KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) - 1
    Case 4
        KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) + 2
        KastenX(Figur.UBound - 2) = KastenX(Figur.UBound - 2) - 1
        KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) + 1
    Case 1
        KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) - 1
        KastenX(Figur.UBound - 2) = KastenX(Figur.UBound - 2) + 1
        KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) + 1
        KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) + 1
        KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) - 1
        KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) + 1
    End Select

Case 7
    Select Case Position
    Case 2
        KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) + 1
        KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) - 1
        KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) - 1
        KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) + 2
        KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) + 1

    Case 3
        KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) + 1
        KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) - 2
        KastenX(Figur.UBound - 3) = KastenX(Figur.UBound - 3) - 1
    Case 4
        KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) - 2
        KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) - 1
        KastenY(Figur.UBound - 1) = KastenY(Figur.UBound - 1) + 1
        KastenX(Figur.UBound - 2) = KastenX(Figur.UBound - 2) - 1
        KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) + 1
    Case 1
        KastenY(Figur.UBound - 0) = KastenY(Figur.UBound - 0) + 2
        KastenX(Figur.UBound - 0) = KastenX(Figur.UBound - 0) - 1
        KastenY(Figur.UBound - 2) = KastenY(Figur.UBound - 2) + 1
        KastenX(Figur.UBound - 2) = KastenX(Figur.UBound - 2) + 1
        KastenY(Figur.UBound - 3) = KastenY(Figur.UBound - 3) - 1
    End Select
End Select

For a = 0 To 3
    'Falls die angestrebte Position besetzt ist:
    'Hiermit werden "normale" Züge rückgängig gemacht, bei denen die Problematik von nicht
    'existierenden Feldern nicht existiert.
    'Unmögliche Drehungen am Rand von allen Figuren außer der länglichen Figur betreffen nämlich
    'nur die 1. Spalte/Zeile außerhalb des Spielfelds, die deklariert ist.
    If Besetzt(KastenXFigur.UBound - a), KastenY(Figur.UBound - a) = True Then

        For j = 0 To 3
            KastenX(Figur.UBound - j) = KastenXDAVOR(j)
            KastenY(Figur.UBound - j) = KastenYDAVOR(j)
        Next
```

```
            Position = Position - 1 'Zurückrotieren
            If Position = 0 Then
               Position = 4
            End If
            Exit For
         End If
      Next

   End If


   If (KeyAscii = 115) Then 'Für das schnellere Absenken des Steins mithilfe der S-Taste
     'Ereignis des Spieltimers wird hier auch ausgeführt
      For a = 0 To 3
         KastenY(Figur.UBound - a) = KastenY(Figur.UBound - a) + 1
      Next
      Call SenkrechteBewegung
   End If


   If (KeyAscii = 97) Then 'a für nach LINKS

      For a = 0 To 3
        'Abfrage, ob die Plätze links neben den Kästen des Steins überhaupt frei sind
         If Besetzt(KastenXFigur.UBound - a) - 1, KastenY(Figur.UBound - a) = True Then
            Linksbewegung = 0
            Exit For
         Else
            Linksbewegung = 1
         End If
      Next

     'Verschieben (1) bzw. nicht-Verschieben (0) des Steins
      For a = 0 To 3
         KastenX(Figur.UBound - a) = KastenX(Figur.UBound - a) - Linksbewegung
      Next

   End If


   If (KeyAscii = 100) Then 'd für nach RECHTS

      For a = 0 To 3
        'Abfrage, ob die Plätze rechts neben den Kästen des Steins überhaupt frei sind
         If Besetzt(KastenXFigur.UBound - a) + 1, KastenY(Figur.UBound - a) = True Then
            Rechtsbewegung = 0
            Exit For
         Else
            Rechtsbewegung = 1
         End If
      Next

     'Verschieben (1) bzw. nicht-Verschieben (0) des Steins
      For a = 0 To 3
         KastenX(Figur.UBound - a) = KastenX(Figur.UBound - a) + Rechtsbewegung
      Next

   End If

   Call Darstellung 'Egal, welche Eingabe welche Bewegung verursacht hat, die Bewegung muss
                    'dargestellt werden.

   If KeyAscii = 112 And lblPause.Visible = False Then 'Falls P für das Starten der Pause gedrückt wurde
      Timer1.Enabled = False
      lblPause.Visible = True
   End If

Else
   If KeyAscii = 112 And lblPause.Visible = True Then 'Falls P für das Beenden der Pause gedrückt wurde
      Timer1.Enabled = True
      lblPause.Visible = False
   End If
End If
End Sub

'Dieser Timer ist der Spieltimer, der während der Spielzeit nahezu immer aktiviert ist.
'Sein Intervall ist abhängig vom Level. In Level 1 beträgt es 600ms und pro Level wird es um 0.8 kürzer.
Private Sub Timer1_Timer()
```

```
    'Stein fällt um eine Koordinate pro Timerereignis herunter;
    'der gleiche Code befindet sich beim eigenständigen Absenken eines Steins in Text1_KeyPress
     For a = 0 To 3
        KastenY(Figur.UBound - a) = KastenY(Figur.UBound - a) + 1
     Next

     Call SenkrechteBewegung
End Sub

'Dieser Timer ist für die Zufälligkeit der Figuren und der Sounds verantwortlich. Er wird nie
'ausgeschaltet.
Private Sub Zufallsgenerator_Timer()

    'Zufällige Figur wird ständig durchgezählt, bevor die Variable für den Warteschlangenstein
    '"NextFigur" beim Zu-Boden-Fallen des Spielsteins den Wert von neueZufallsfigur annimmt.
    'Bemerkung:
    'Das Intervall des Zufallsgenerators beträgt eine Millisekunde. Weil jeder Spielstein verschieden
    'lange zum Herunterfallen braucht (eigenständiges Absenken mit s, verschiedene Fallhöhen gegeben
    'durch Spielfeld oder verschiedene Steingrößen etc.) ist auch wirklich eine Zufälligkeit für den
    'nächsten Stein gegeben. Nur, wenn man es zu Stande bringt, dass die Anzahl der Millisekunden
    'zwischen den Auswahlereignissen der neuen Steine immer durch 7 teilbar ist, bekommt man immer
    'den gleichen neuen Stein.
     neueZufallsfigur = neueZufallsfigur + 1
     If neueZufallsfigur = 8 Then
        neueZufallsfigur = 1
     End If

    'Der Zufallssound wird eingespielt, wenn man 4 Reihen auf einmal vervollständigt hat. Er ist ebenfalls
    'von der Zeit abhängig.
     Zufallssound = Zufallssound + 1
     If Zufallssound = 2 Then
        Zufallssound = 0
     End If

    'sonstige nützliche Befehle, die immer gelten
     lblLevel.Caption = Level
     lblLinien.Caption = Linien
End Sub

'Umsetzen der Position auf dem selbst erstellten Raster (KastenXY) in eine konkrete Position im Fenster
Public Function Darstellung()

    For a = 0 To 3
       With Figur(Figur.UBound - a)
         .Top = 360 * (KastenYFigur.UBound - a) - 1) + 1200
         .Left = 360 * (KastenXFigur.UBound - a) - 1) + 1200
       End With
    Next
End Function

'Diese Funktion wird immer aufgerufen, wenn eine senkrechte Bewegung gemacht wurde.
Public Function SenkrechteBewegung()

    For a = 0 To 3
      'Abfrage, ob einer der Kästen des Steins auf einem bereits besetzten Feld gelandet ist
       If Besetzt(KastenXFigur.UBound - a), KastenY(Figur.UBound - a) = True Then

          For p = 0 To 3
            'Alle Kästen des Steins müssen wieder um ein Feld höher gerückt werden
             KastenY(Figur.UBound - p) = KastenY(Figur.UBound - p) - 1

            'diese Position abspeichern, sodass der nächste Stein dort nicht hin kann
             Besetzt(KastenXFigur.UBound - p), KastenY(Figur.UBound - p) = True
          Next

          Call NeueFigur 'Da alte Figur ihren Platz gefunden hat
          Exit For
       End If
    Next

    'Falls sich die Steine bis oben hin aufgetürmt haben, verliert man
     If Besetzt(6, 1) Or Besetzt(5, 1) Or Besetzt(7, 1) Then
        lblVerloren.Caption = "Fail!"
        Call VerlorenGewonnen
        Exit Function
     End If

    'Prüfen, ob eine Reihe vervollständigt wurde
     ReiheVoll = 0 'Initialisierung
     AnzahlReihen = 0 'Initialisierung
```

```
    For y = 1 To 20

        For x = 1 To 12 'Durchzählen
            If Besetzt(x, y) = True Then
                ReiheVoll = ReiheVoll + 1
            Else
                ReiheVoll = 0
                Exit For
            End If
        Next x

        If ReiheVoll = 12 Then '12 -> Reihe ist voll, da alle 12 horizontalen Felder besetzt sind

            AnzahlReihen = AnzahlReihen + 1

            For C = 1 To Figur.UBound
                If (FigurC).Top - 1200) / 360 = y - 1 Then
                    Figur(C).Visible = False 'Ausblenden der Reihe
                End If

                If (FigurC).Top - 1200) / 360 < y - 1 Then
                    Figur(C).Top = Figur(C).Top + 360 'Runterrutschenlassen der unvollständigen Reihen darüber
                                                      '(Anzeige der Steuerelemente)
                End If
            Next C

            For a = y To 1 Step -1
                For b = 1 To 12
                    Besetzt(b, a) = Besetzt(b, a - 1) 'Runterrutschenlassen der unvollständigen Reihen darüber
                                                      '(Array, mit dem gerechnet wird)
                Next b
            Next a

            ReiheVoll = 0 'damit für die nächste Y-Zeile der Wert schon zurückgesetzt ist

        End If

    Next y

    If AnzahlReihen > 0 Then

            Linien = Linien + AnzahlReihen

            If Linien >= 20 Then 'Bedingung für das nächste Level
                lblVerloren.Caption = "Gut Gemacht!"
                Call VerlorenGewonnen 'Spielunterbrechung
                Level = Level + 1
                Linien = 0
                Timer1.Interval = Timer1.Interval * 4 / 5 'erhöhter Schwierigkeitsgrad
                cmdWeiter.Visible = True
                Exit Function
            End If

            Select Case AnzahlReihen
            Case 4 'Bei 4 Reihen auf einmal wird "400 Babies!" oder "Bear Blasting!" eingespielt
                Select Case Zufallssound
                Case 1
                    Dateiname = App.Path
                    If Right(Dateiname, 1) <> "\" Then Dateiname = Dateiname & "\"
                    Dateiname = Dateiname & "Babies.wav"
                    sndPlaySound Dateiname, 1 'Nutzen der Windowsfunktion
                Case 0
                    Dateiname = App.Path
                    If Right(Dateiname, 1) <> "\" Then Dateiname = Dateiname & "\"
                    Dateiname = Dateiname & "BearBlasting.wav"
                    sndPlaySound Dateiname, 1 'Nutzen der Windowsfunktion
                End Select
            End Select

            imgBear.Visible = True
            Reihe.Interval = 500 + AnzahlReihen * 150 'je mehr Reihen man vervollständigt,
            Reihe.Enabled = True                      'desto länger ist die Pause danach
            Timer1.Enabled = False
    End If

    Call Darstellung

End Function

'Diese Funktion wird immer aufgerufen, wenn man verloren oder gewonnen (nächstes Level) hat.
```

```
Public Function VerlorenGewonnen()

    Timer1.Enabled = False 'Anhalten des Spiels

    For a = 0 To Figur.UBound 'Unsichtbarmachen aller Steine
        Figur(a).Visible = False
    Next

    For x = 1 To 12
        For y = 1 To 20
            Besetzt(x, y) = False 'Freimachen des Gesamten Spielfelds
        Next
    Next

   'Falls man verloren hat, wird der Versagersound eingespielt
    If lblVerloren.Caption = "Fail!" Then
        Dateiname = App.Path
        If Right(Dateiname, 1) <> "\" Then Dateiname = Dateiname & "\"
        Dateiname = Dateiname & "OhLord.wav"
        sndPlaySound Dateiname, 1 'Nutzen der Windowsfunktion
        cmdStart.Enabled = True
    End If

   'Falls man gewonnen hat, wird der Gewinnersound eingespielt
    If lblVerloren.Caption = "Gut Gemacht!" Then
        Dateiname = App.Path
        If Right(Dateiname, 1) <> "\" Then Dateiname = Dateiname & "\"
        Dateiname = Dateiname & "TopScore.wav"
        sndPlaySound Dateiname, 1 'Nutzen der Windowsfunktion
    End If

    lblVerloren.Visible = True 'Anzeigen des Banners
End Function
```