

;
;
;
;
;
;
;
;
;
;
;
;

CCCCCCCCCCCCCCCC

; A
G
SC
CG
G
GC
S
T
F

;

GGGGGGGG

```

Dim a#(MaxNumk), aSoll#(MaxNumk) ;Koeffizienten der F.-Reihe
Dim b#(MaxNumk), bSoll#(MaxNumk)
Dim y#(Int(AuflX/3))
Dim FktY#(360)
Dim Helptext$(10)
Dim Sinus#(360*PrecSin), Cosinus#(360*PrecSin)

;Grafik laden

Global SlideG = LoadImage(CurrentDir$() + "FGfx\SlideG.bm_")
MaskImage SlideG, 255, 0, 0 : MidHandle SlideG
Global HoeheSlide=ImageHeight(SlideG)

Global SlideButton = LoadAnimImage(CurrentDir$() + "FGfx\SlideButton.bm_", 22,
11, 0, 3)
MaskImage SlideButton, 255, 0, 0 : MidHandle SlideButton

Global Pfeil = LoadImage(CurrentDir$() + "FGfx\Pfeil.bm_")
MaskImage Pfeil, 255, 0, 0 : MidHandle Pfeil

Global ButtonDefault = LoadAnimImage(CurrentDir$() + "FGfx\ButtonDefault.bm_",
114, 37, 0, 2)
Global BreiteButton=ImageWidth(ButtonDefault)
Global HoeheButton=ImageHeight(ButtonDefault)

Global FktButton = LoadAnimImage(CurrentDir$() + "FGfx\FktButton.bm_", 171, 143,
0, 2);???
Global BreiteFktButton=ImageWidth(FktButton)
Global HoeheFktButton=ImageHeight(FktButton)

Global SinusFkt = LoadImage(CurrentDir$() + "FGfx\SinusFkt.bm_")
MaskImage SinusFkt, 255, 0, 0 : MidHandle SinusFkt

Global SinusHackFkt = LoadImage(CurrentDir$() + "FGfx\SinusHackFkt.bm_")
MaskImage SinusHackFkt, 255, 0, 0 : MidHandle SinusHackFkt

Global DreieckFkt = LoadImage(CurrentDir$() + "FGfx\DreieckFkt.bm_")
MaskImage DreieckFkt, 255, 0, 0 : MidHandle DreieckFkt

Global KippFkt = LoadImage(CurrentDir$() + "FGfx\KippFkt.bm_")
MaskImage KippFkt, 255, 0, 0 : MidHandle KippFkt

Global RechteckFkt = LoadImage(CurrentDir$() + "FGfx\RechteckFkt.bm_")
MaskImage RechteckFkt, 255, 0, 0 : MidHandle RechteckFkt

Global SaegelFkt = LoadImage(CurrentDir$() + "FGfx\SaegelFkt.bm_")
MaskImage SaegelFkt, 255, 0, 0 : MidHandle SaegelFkt

Global ExpFkt = LoadImage(CurrentDir$() + "FGfx\ExpFkt.bm_")
MaskImage ExpFkt, 255, 0, 0 : MidHandle ExpFkt

Global AnimButton = LoadAnimImage(CurrentDir$() + "FGfx\AnimButton.bm_", 32, 28,
0, 2);???

Global StartAnimButton = LoadImage(CurrentDir$() + "FGfx\StartAnimButton.bm_")
MaskImage StartAnimButton, 255, 0, 0 : MidHandle StartAnimButton

Global StopAnimButton = LoadImage(CurrentDir$() + "FGfx\StopAnimButton.bm_")
MaskImage StopAnimButton, 255, 0, 0 : MidHandle StopAnimButton

;Hauptprogramm -----

```

```

;Vorberechnungen für höhere Geschwindigkeit
For degree=0 To 360*PrecSin
    Sinus#(degree)=Sin#(degree/PrecSin)
    Cosinus#(degree)=Cos#(degree/PrecSin)
Next

```

```

;Fenster aktualisieren
Cls
SetKoeffZero()
GetValues()
DrawFunction(NumkDefault)
SetDelayAnim()
AddKoeffNum()
CheckMouse()
Flip 1

```

```

Repeat
If KeyDown(3) Then Spektrum()
    Feld=CheckMouse(MouseX(), MouseY())           ;Input überwachen
    Select Feld
        Case 1                                     ;Funktion auswählen
            FktId=ChooseFunction()
            AktualAll=True
        Case 2                                     ;Freie Funktion
            AppTitle Title$ + " - freie Funktion"
            FktId=0
            AktualAll=True
        Case 3                                     ;Ursprüngliche Koeffizienten wiederherstellen
            ResetKoeff(FktId)
            AktualAll=True
        Case 4                                     ;Koeffizienten Null setzen
            SetKoeffZero()
            AktualAll=True
        Case 5                                     ;Koeffizienten einlesen
            Numk=GetValues(MouseX(), MouseY())
            AktualAll=True
        Case 6                                     ;Animation starten
            Anim = True
            NumkTemp=Numk ;eingestellter max. Wert für k
        Case 7                                     ;Animation beenden
            Anim = False
            NumkTemp=0
        Case 8                                     ;Programmende
            End
        Case 9                                     ;Verzögerung für Animation
            DelayAnim=SetDelayAnim(True)
            SetDelayAnim()
        Case 10                                    ;Größere Anzahl an Koeffizienten
            Numk=AddKoeffNum(True)
            AktualAll=True
    End Select

;Animation
If Anim Then
    If (MilliSecs()-TimeUpdateAnim) > DelayAnim Then           ;Verzögerung zur
nächsten Berechnung
        Numk = Numk+1
        If Numk>NumkTemp Then Numk=0
        DrawFunction(Numk, FktId, Spectrum, NumkTemp)
        AddKoeffNum()
        TimeUpdateAnim=MilliSecs()

```

```

    EndIf
EndIf

;Aktualisierungen

If AktualAll Then
    ClsColor GuiR, GuiG, GuiB
    Cls
    DrawFunction(Numk, FktId, Spectrum, NumkTemp)
    SetDelayAnim()
    AddKoeffNum()
    CheckMouse
    AktualAll = False
EndIf
GetValues()

If KeyDown(1) Then ProgramEnd=True
Flip 1
WaitTimer(Takt)
Until ProgramEnd
End

;Funktionen -----

Function DrawFunction(Numk, Fkt=0, Spect=0, NumkTemp=0)

    Restore DrawFunction
    Read OrigX, OrigY, Length, Height

    Local Sum#
    Local k, z=1
    Local ZoomX# = 3*360.0/Length
    Local ZoomY# = Height/2/2
    Local Fehler#, FehlerMittel#

    Viewport OrigX-DickeRahmen, OrigY-DickeRahmen, Length+DickeRahmen*2+1,
Height+DickeRahmen*2+1
    Origin OrigX, OrigY + Height/2
    ClsColor 200-Spect*30, 200-Spect*40, 240-Spect*20
    Cls

    LockBuffer BackBuffer() ;Puffer für schnelle Pixelverarbeitung sperren

    If Numk=0 Then ;Berechnung erst ab Koeffizientenanzahl>=1
        For x=1 To Length/3
            y#(x)=a#(0)/2
            Color 255, 255, 0
            Line x, -y#(x)*ZoomY#, x+1, -y#(x)*ZoomY#
        Next
    EndIf

    For k=1 To Numk

        ;Berechnung der Kurve
        For x=0 To Length/3 ;Berechnung nur von einer Periode
            (Rechenaufwand)
            If k=1 Then
                Sum# = a#(0)/2 ;Erster Koeffizient
            Else
                Sum# = y#(x)
            EndIf
            ;Sum# = Sum# + a#(k)*Cosinus#(Int(PrecSin*k*x*ZoomX#)Mod(PrecSin*360)) +

```

```

b#(k)*Sinus#(Int(PrecSin*k*x*ZoomX#)Mod(PrecSin*360))
Sum# = Sum# + a#(k)*Cos#(k*x*ZoomX#) + b#(k)*Sin#(k*x*ZoomX#)
y#(x) = Sum#
Next

;Errechnete Kurve zeichnen (nur eine Periode)
If Spect=1 Or k=Numk Then ;Kurvenschar nur bei Wahl anzeigen
Color 50-Float#(k)/Numk*50, 255-Float#(k)/Numk*255 ,Float#(k)/Numk*255
;Farbübergang
z=0
For x=1 To Length/3
Line x-1, -y#(x-1)*ZoomY#, x, -y#(x)*ZoomY#
Next
EndIf

Next

UnlockBuffer BackBuffer() ;Puffer wieder freigeben

;Schnelle Grafik für 2. und 3. Periode
For x=1 To 2
CopyRect OrigX+1, OrigY-Height/2-2, Length/3, Height+4, x*Length/3+1,
-Height/2-2, BackBuffer(), BackBuffer()
Next

;Vorgegebene Funktion mit Fehler von 2T bis 3T zeichnen
If Fkt<> 0 And Spect=0 Then
For x=1 To Length/3
;Fehler markieren
Color 255, 0, 0
If Int(FktY#(x*ZoomX#)*ZoomY#) <> Int(y#(x)*ZoomY#) Then ;Vermeidung von
Grafikfehlern
Line x+Length*2/3, -FktY#(x*ZoomX#)*ZoomY#, x+Length*2/3, -y#(x)*ZoomY#
EndIf
;Vorgegebene Funktion
Color 0, 233, 100
Line x+Length*2/3-1, -FktY#(x*ZoomX#-1)*ZoomY#, x+Length*2/3, -FktY#
(x*ZoomX#)*ZoomY#
Line x+Length*2/3-1, -FktY#(x*ZoomX#-1)*ZoomY#+1, x+Length*2/3, -FktY#
(x*ZoomX#)*ZoomY#+1 ;dicke Linie
Line x+Length*2/3-1, -FktY#(x*ZoomX#-1)*ZoomY#-1, x+Length*2/3, -FktY#
(x*ZoomX#)*ZoomY#-1
Next
EndIf

;Koordinatensystem

SetFont FontBeschriftung
Color 0, 0, 0
Rect 0, 0, Length, 2
Color 50, 50, 50
Rect Length/3, -Height/2, 2, Height
Line Length*2/3, -Height/2, Length*2/3, Height/2
;Pfeil
Line Length/3-3, -Height/2+3, Length/3+1, -Height/2 : Line Length/3+4, -Height
/2+3, Length/3+1, -Height/2
Line Length/3-3, -Height/2+4, Length/3+1, -Height/2+1 : Line Length/3+4,
-Height/2+4, Length/3+1, -Height/2+1
Text Length/3-10, -Height/2+2, "y", 0, 1

Color 0+Spect*255, 0+Spect*255, 0+Spect*255
Line Length/6, -3, Length/6, 3 : Text Length/6, 2, "-T/2", 1

```

```

Text Length/3+5, 2, "0"
Line Length*3/6, -3, Length*3/6, 3 : Text Length*3/6, 2, "1/2 T", 1
Text Length*2/3+5, 2, "T"
Line Length*5/6, -3, Length*5/6, 3 : Text Length*5/6, 2, "3/2 T", 1
Text Length*3/3+5, 2, "2T"

Line Length/3-3, -1.5*ZoomY#, Length/3+3, -1.5*ZoomY# : Text Length/3-20,
-1.5*ZoomY#, "1.5", 0, 1
Line Length/3-3, 1.5*ZoomY#, Length/3+3, 1.5*ZoomY# : Text Length/3-20,
1.5*ZoomY#, "-1.5", 0, 1
Line Length/3-3, -0.5*ZoomY#, Length/3+3, -0.5*ZoomY# : Text Length/3-20,
-0.5*ZoomY#, "0.5", 0, 1
Line Length/3-3, 0.5*ZoomY#, Length/3+3, 0.5*ZoomY# : Text Length/3-20,
0.5*ZoomY#, "-0.5", 0, 1
Line Length/3-3, -1*ZoomY#, Length/3+3, -1*ZoomY# : Text Length/3-20,
-1*ZoomY#, "1.0", 0, 1
Line Length/3-3, 1*ZoomY#, Length/3+3, 1*ZoomY# : Text Length/3-20, 1*ZoomY#,
"-1.0", 0, 1

;Rahmen
Color GuiR*0.7, GuiG*0.7, GuiB*0.7
Line -DickeRahmen, -DickeRahmen-Height/2, Length+DickeRahmen, -DickeRahmen-
Height/2
Line -DickeRahmen, -DickeRahmen-Height/2, -DickeRahmen, DickeRahmen+Height/2
Color GuiR*0.4, GuiG*0.4, GuiB*0.4
Line -DickeRahmen+1, -DickeRahmen-Height/2+1, Length+DickeRahmen-1,
-DickeRahmen-Height/2+1
Line -DickeRahmen+1, -DickeRahmen-Height/2+1, -DickeRahmen+1,
DickeRahmen+Height/2-1

Color GuiR*1.05, GuiG*1.05, GuiB*1.05
Line Length+DickeRahmen, DickeRahmen+Height/2, -DickeRahmen,
DickeRahmen+Height/2
Line Length+DickeRahmen, DickeRahmen+Height/2, Length+DickeRahmen,
-DickeRahmen-Height/2
Color GuiR*1.11, GuiG*1.11, GuiB*1.11
Line Length+DickeRahmen-1, DickeRahmen+Height/2-1, -DickeRahmen+2,
DickeRahmen+Height/2-1
Line Length+DickeRahmen-1, DickeRahmen+Height/2-1, Length+DickeRahmen-1,
-DickeRahmen-Height/2+2

;Fehler anzeigen und berechnen, falls vorgegebene Funktion
Read OrigXDisplay, OrigYDisplay, LengthDisplay, HeightDisplay
Viewport OrigXDisplay, OrigYDisplay, LengthDisplay, HeightDisplay
Origin OrigXDisplay, OrigYDisplay
ClsColor GuiR, GuiG, GuiB
Cls

SetFont FontSchrift

If Fkt<>0 Then
Fehler# = 0
For x=0 To 360
Fehler# = Fehler + (FktY#(x)-y#(x/ZoomX#))^2
Next
FehlerMittel# = Fehler#/360

Color 0, 0, 0
Text 10, 5, "Mittlerer quadratischer"
Text 10, 20, "Fehler:"
Else
Color GuiR*1.1, GuiG*1.1, GuiB*1.1

```

```

Text 10+1, 5+1, "Mittlerer quadratischer"
Text 10+1, 20+1, "Fehler:"
Color GuiR*0.8, GuiG*0.8, GuiB*0.8
Text 10, 5, "Mittlerer quadratischer"
Text 10, 20, "Fehler:"
EndIf

```

```

Local DisplayX=10, DisplayY=45
DrawImage ButtonDefault, DisplayX, DisplayY, 1
SetFont FontSchriftGross
Text DisplayX+BreiteButton/2, DisplayY+HoeheButton/2, FehlerMittel#, 1, 1

```

```

Origin 0, 0
Viewport 0, 0, AuflX, AuflY

```

End Function

;

Function GetValues(x=0, y=0)

```

Restore GetValues
Read OrigX, OrigY, Length, Height

```

```

Local Faktor# = 0.014
Local NumkField

```

;Nur maximal sichtbare Anzahl an Koeffizienten zulassen

```

If Numk>MaxNumkField Then
    NumkField = MaxNumkField

```

Else

```

    NumkField = Numk

```

EndIf

;Änderung der Anzahl an Koeffizienten über Mausrad

```

Diff=MouseZSpeed()

```

```

If Diff<>0 Then

```

```

    Numk=Numk+Diff

```

```

    If Numk<0 Then ;Zulässiger Bereich

```

```

        Numk=0

```

```

    ElseIf Numk>MaxNumk

```

```

        Numk=MaxNumk

```

```

    EndIf

```

```

    AktualAll=True ;Aktualisieren

```

EndIf

;Werte aktualisieren, wenn Button gedrückt

```

If x<>0 And y<>0 Then

```

;Änderung der Anzahl der Koeffizienten

```

If y>OrigY+Height/2+HoeheSlide/2 Then

```

```

    If FieldActive=-2 Then

```

```

        Numk=Int(Float#(x-OrigX)/(BreiteSlide*2))-1

```

```

        If Numk<0 Then Numk=0

```

;nur wenn angeklickt

;sinnvolle Anzahl

```

    ElseIf FieldActive=-1 Then

```

```

        FieldActive=-2

```

;Kennziffer für k

```

    EndIf

```

Else

;Änderung der Koeffizienten

;Button nur im zulässigen Bereich

```

If (y-OrigY-Height/2) > HoeheSlide/2 Then y=HoeheSlide/2+OrigY+Height/2

```

```

        If (y-OrigY-Height/2) < -HoeheSlide/2 Then y=-HoeheSlide/2+OrigY+Height/2

        Wert# = -(y - OrigY-Height/2)*Faktor#           ;neuer Wert des
Koeffizienten
        Num = Int(Float#(x-OrigX)/(BreiteSlide)-0.5)   ;Nummer des geänderten
Slides
        If FieldActive=-1 Then                         ;Neuer Klick
            FieldActive=Num
        EndIf

        If Num=FieldActive Then                         ;nur angeklickten Slide
ändern
            If Num = 0 Then
                a#(0) = Wert#                           ;Koeffizient a_0
            Else
                If Num/2.0-Int(Num/2) <> 0 Then           ;gerade Nummer: Koeffizient
b
                    k = Int(Num/2)
                    b#(k) = Wert#
                Else                                     ;sonst a
                    k = Int(Num/2)
                    a#(k) = Wert#
                EndIf
            EndIf

        EndIf

    EndIf

EndIf

;Fenster aktualisieren
Viewport OrigX - DickeRahmen*2, OrigY - DickeRahmen*2-5, Length +
DickeRahmen*4+5, Height + DickeRahmen*4+5
Origin OrigX, OrigY + Height/2
ClsColor GuiR, GuiG, GuiB
Cls

;Rahmen
SetFont FontSchrift
Color GuiR*0.6, GuiG*0.6, GuiB*0.6
Rect -DickeRahmen*2, -Height/2-DickeRahmen*2, Length+DickeRahmen*4-1,
Height+DickeRahmen*4-1, 0
Color GuiR*1.1, GuiG*1.1, GuiB*1.1
Rect -DickeRahmen*2+1, -Height/2-DickeRahmen*2+1, Length+DickeRahmen*4-1,
Height+DickeRahmen*4-1, 0
Color GuiR, GuiG, GuiB
Rect 0, -DickeRahmen*2-Height/2, StringWidth("Koeffizienten")+2*4, 2, 1
Color GuiR*1.1, GuiG*1.1, GuiB*1.1
Text 5, -DickeRahmen*2-Height/2+1, "Koeffizienten", 0, 1
Color 30, 30, 30
Text 4, -DickeRahmen*2-Height/2, "Koeffizienten", 0, 1

;Beschriftung
Color 0, 0, 0
Line 0, 0, Length, 0
Line BreiteSlide*1.8, -HoeheSlide/2, BreiteSlide*1.8, HoeheSlide/2
SetFont FontBeschriftung
Line BreiteSlide*1.8-5, -0.5/Faktor#, BreiteSlide*1.8+6, -0.5/Faktor# : Text
BreiteSlide*1.8-20, -0.5/Faktor#, "0.5", 0, 1
Line BreiteSlide*1.8-5, 0.5/Faktor#, BreiteSlide*1.8+6, 0.5/Faktor# : Text

```



```

BreiteSlide*1.8-20, 0.5/Faktor#, "-0.5", 0, 1
Line BreiteSlide*1.8-5, -1/Faktor#, BreiteSlide*1.8+6, -1/Faktor# : Text
BreiteSlide*1.8-15, -1/Faktor#, "1", 0, 1
Line BreiteSlide*1.8-5, 1/Faktor#, BreiteSlide*1.8+6, 1/Faktor# : Text
BreiteSlide*1.8-15, 1/Faktor#, "-1", 0, 1

;Slides zeichnen
SetFont FontSchrift : Text BreiteSlide/2, -HoeheSlide/2-25, "a", 1
SetFont FontBeschriftung : Text BreiteSlide/2+4, -HoeheSlide/2-20, "0"
DrawImage SlideG, BreiteSlide/2, 0 ;Kein Slide für b_0
DrawImage SlideButton, BreiteSlide/2, -a#(0)/Faktor#, 0
For k=1 To NumkField
    SetFont FontSchrift : Text BreiteSlide/2*(4*k+1), -HoeheSlide/2-25, "a", 1
    SetFont FontBeschriftung : Text BreiteSlide/2*(4*k+1)+4, -HoeheSlide/2-20, k
    DrawImage SlideG, BreiteSlide/2*(4*k+1), 0
    DrawImage SlideButton, BreiteSlide/2*(4*k+1), -a#(k)/Faktor#, 0

    SetFont FontSchrift : Text BreiteSlide/2*(4*k+3), -HoeheSlide/2-25, "b", 1
    SetFont FontBeschriftung : Text BreiteSlide/2*(4*k+3)+4, -HoeheSlide/2-20, k
    DrawImage SlideG, BreiteSlide/2*(4*k+3), 0
    DrawImage SlideButton, BreiteSlide/2*(4*k+3), -b#(k)/Faktor#, 1
Next
For k=NumkField+1 To MaxNumkField ;inaktive Slides
    SetFont FontSchrift : Text BreiteSlide/2*(4*k+1), -HoeheSlide/2-25, "a", 1
    SetFont FontBeschriftung : Text BreiteSlide/2*(4*k+1)+4, -HoeheSlide/2-20, k
    DrawImage SlideG, BreiteSlide/2*(4*k+1), 0
    DrawImage SlideButton, BreiteSlide/2*(4*k+1), -a#(k)/Faktor#, 2

    SetFont FontSchrift : Text BreiteSlide/2*(4*k+3), -HoeheSlide/2-25, "b", 1
    SetFont FontBeschriftung : Text BreiteSlide/2*(4*k+3)+4, -HoeheSlide/2-20, k
    DrawImage SlideG, BreiteSlide/2*(4*k+3), 0
    DrawImage SlideButton, BreiteSlide/2*(4*k+3), -b#(k)/Faktor#, 2
Next

;Regler für Anzahl der Koeffizienten zeichnen
Color 0, 0, 0
SetFont FontSchrift
Text 10, Height/2+HoeheSlide/2+25, "k:", 0, 1
Line 60, Height/2+HoeheSlide/2+25, Length-5, Height/2+HoeheSlide/2+25
For k=2 To MaxNumkField
    Color GuiR*1.1, GuiG*1.1, GuiB*1.1
    Line k*BreiteSlide*2+1, Height/2-HoeheSlide/2-25+1, k*BreiteSlide*2+1,
Height/2+HoeheSlide/2+25+1
    Color GuiR*0.6, GuiG*0.6, GuiB*0.6
    Line k*BreiteSlide*2, Height/2-HoeheSlide/2-25, k*BreiteSlide*2,
Height/2+HoeheSlide/2+25
Next
DrawImage Pfeil, (NumkField+1)*BreiteSlide*2, Height/2+HoeheSlide/2+25

Origin 0, 0
Viewport 0, 0, AuflX, AuflY
Return Numk

End Function

;

Function SetKoefZero()

For k=0 To MaxNumk
    a#(k) = 0
    b#(k) = 0

```

```

Next
Return True

End Function

;

Function ResetKoeff(FktId)

    a#(0)=aSol#(0)
    For k=1 To MaxNumk
        a#(k)=aSol#(k)
        b#(k)=bSol#(k)
    Next

End Function

;

Function ChooseFunction()

    Local OffsetY=55
    Local FktIdOld=FktId

    Restore ChooseFunction
    Read OrigX, OrigY, Spalten, Zeilen

    Viewport OrigX-DickeRahmen*2, OrigY-DickeRahmen*2,
    Spalten*BreiteFktButton+DickeRahmen*4,
    Zeilen*HoeheFktButton+DickeRahmen*4+OffsetY
    Origin OrigX, OrigY
    ClsColor GuiR, GuiG, GuiB
    Cls

    Local Feld=0
    Local x, y

    ;Rahmen
    Color GuiR*0.2, GuiG*0.4, GuiB*0.9
    Rect -DickeRahmen*2, -DickeRahmen*2, Spalten*BreiteFktButton+DickeRahmen*4,
    Zeilen*HoeheFktButton+DickeRahmen*4+OffsetY, 0
    Rect -DickeRahmen*2+1, -DickeRahmen*2+1,
    Spalten*BreiteFktButton+DickeRahmen*4-2,
    Zeilen*HoeheFktButton+DickeRahmen*4+OffsetY-2, 0
    Color GuiR*0.6, GuiG*0.7, GuiB*1.1
    Line -DickeRahmen*2, -DickeRahmen*2, Spalten*BreiteFktButton+DickeRahmen*2,
    -DickeRahmen*2
    Line -DickeRahmen*2, -DickeRahmen*2, -DickeRahmen*2,
    Zeilen*HoeheFktButton+DickeRahmen*2+OffsetY
    Line -DickeRahmen*2+1, -DickeRahmen*2+1,
    Spalten*BreiteFktButton+DickeRahmen*2-1, -DickeRahmen*2+1
    Line -DickeRahmen*2+1, -DickeRahmen*2+1, -DickeRahmen*2+1,
    Zeilen*HoeheFktButton+DickeRahmen*2+OffsetY-1

    ;Fenster zu Beginn aktualisieren

    SetFont FontSchriftGross
    Color GuiR*1.1, GuiG*1.1, GuiB*1.1
    Text 11, 16, "Funktion auswählen:"
    Color 0, 0, 0
    Text 10, 15, "Funktion auswählen:"
    SetFont FontSchrift

```

```

    DrawImage ButtonDefault, 200, 5, 0 : Text 200+BreiteButton/2, 5+HoeheButton/2,
    "Übernehmen", 1, 1
    DrawImage ButtonDefault, 350, 5, 0 : Text 350+BreiteButton/2, 5+HoeheButton/2,
    "Abbrechen", 1, 1

    SetFont FontSchriftGross

    For Zeile=1 To Zeilen
        For Spalte=1 To Spalten
            If (Zeile-1)*Spalten+Spalte=Feld Then ;momentane Auswahl
                Aktiv = True
            Else
                Aktiv = False
            EndIf
            DrawImage FktButton, BreiteFktButton*(Spalte-1)+DickeRahmen,
            HoeheFktButton*(Zeile-1)+OffsetY, Aktiv

            ;Bezeichnung der Buttons mit Funktionen
            Zelle = (Zeile-1)*Spalten + Spalte
            Select Zelle
            Case 1
                DrawImage RechteckFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen +
                Aktiv*3, HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
                HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "Rechteckschwingung", 1, 1
            Case 2
                DrawImage SinusFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen,
                HoeheFktButton*(Zeile-0.3)+OffsetY
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen, HoeheFktButton*(Zeile-
                0.7)+OffsetY, "|sin(x)|", 1, 1
            Case 3
                DrawImage KippFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
                HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
                HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "Kippschwingung", 1, 1
            Case 4
                DrawImage DreieckFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen +
                Aktiv*3, HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
                HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "Dreieckschwingung", 1, 1
            Case 5
                DrawImage SaegelFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
                HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
                HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "Sägezahn", 1, 1
            Case 6
                DrawImage ExpFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
                HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
                HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "e-Funktion", 1, 1
            End Select
        Next
    Next

    Repeat
        If MouseDown(1) Then
            x=MouseX() : y=MouseY()

            ;Maus auf Kollision prüfen
            If y<OrigY+OffsetY And y>OrigY Then ;Mausklick auf Buttons
                If x>OrigX+200 And x<OrigX+200+BreiteButton Then

```

```

        DrawImage ButtonDefault, 200, 5, 1 : Text 200+BreiteButton/2+3,
5+HoeheButton/2+3, "Übernehmen", 1, 1
        Flip 1
        Delay DelayButton

        ;Funktion setzen
        LoadKoef(Feld)
        ;Slides setzen
        ResetKoef(Feld)

        Viewport 0, 0, AuflX, AuflY
        Origin 0, 0
        Return Feld
    ElseIf x>OrigX+350 And x<OrigX+350+BreiteButton Then
        ;Abbruch
        DrawImage ButtonDefault, 350, 5, 1 : Text 350+BreiteButton/2+3,
5+HoeheButton/2+3, "Abbrechen", 1, 1
        Flip 1
        Delay DelayButton
        Return FktIdOld
    EndIf
    ElseIf x>OrigX And x<OrigX+BreiteFktButton*Spalten      ;Auswahl einer
Funktion
        Spalte=Int((x-OrigX-DickeRahmen)/BreiteFktButton) + 1
        Zeile=Int((y-OrigY-OffsetY-DickeRahmen)/HoeheFktButton)
        Feld=Zeile*Spalten + Spalte
        If Feld > 6 Then Feld = 0                                ;Nicht vorhandene
Fkt nicht auswählbar
    EndIf

    ;Fenster aktualisieren
    For Zeile=1 To Zeilen
        For Spalte=1 To Spalten
            If (Zeile-1)*Spalten+Spalte=Feld Then      ;momentane Auswahl
                Aktiv = True
            Else
                Aktiv = False
            EndIf
            DrawImage FktButton, BreiteFktButton*(Spalte-1)+DickeRahmen,
HoeheFktButton*(Zeile-1)+OffsetY, Aktiv

            ;Bezeichnung der Buttons mit Funktionen
            Zelle = (Zeile-1)*Spalten + Spalte
            Select Zelle
            Case 1
                DrawImage RechteckFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen +
Aktiv*3, HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "Rechteckschwingung", 1, 1
            Case 2
                DrawImage SinusFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen +
Aktiv*3, HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "|sin(x)|", 1, 1
            Case 3
                DrawImage KippFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen +
Aktiv*3, HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
                Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "Kippschwingung", 1, 1
            Case 4
                DrawImage DreieckFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen +
Aktiv*3, HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3

```

```

        Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "Dreieckschwingung", 1, 1
    Case 5
        DrawImage SaegeFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen +
Aktiv*3, HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
        Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "Sägezahn", 1, 1
    Case 6
        DrawImage ExpFkt, BreiteFktButton*(Spalte-0.5)+DickeRahmen +
Aktiv*3, HoeheFktButton*(Zeile-0.3)+OffsetY + Aktiv*3
        Text BreiteFktButton*(Spalte-0.5)+DickeRahmen + Aktiv*3,
HoeheFktButton*(Zeile-0.7)+OffsetY + Aktiv*3, "e-Funktion", 1, 1
    End Select
Next
Next

EndIf

Flip 1
WaitTimer(Takt)
Until KeyDown(1); Or MouseDown(2)

Viewport 0, 0, AuflX, AuflY
Origin 0, 0
Return 0

End Function

;

Function GetTask(x, y)

End Function

;

Function CheckMouse(x=0, y=0)

    Local Feld=0

    If KeyHit(57) Then
        Spectrum=1-Spectrum           ;Kurvenschar anzeigen
        AktualAll=True
        FlushKeys
    EndIf

    ;Mausklick festsetzen, um versehentliche Änderungen durch Verwackeln zu
    verhindern
    If MouseDown(1) Then
        If ClickMouse=False Then
            FieldActive=-1
            ClickMouse=True
        EndIf
    Else
        ClickMouse=False
    EndIf

    If MouseDown(1) Or (x=0 And y=0) Or MouseDown(2) Then    ;?????????!!!

        ;Bei Mausclick auf Kollision mit Slides testen oder nur aktualisieren
        Restore GetValues
        Read OrigX, OrigY, Length, Height

```

```

If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
  If MouseDown(2) Then
    HelpBubble(x, y, "Änderung der Koeffizienten$durch Verschieben der
Regler$mit der linken Maustaste")
  Else
    Feld = 5
  EndIf
EndIf

Restore SetDelayAnim
Read OrigX, OrigY, Length, Height
If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
  Feld = 9
EndIf

Restore AddKoeffNum
Read OrigX, OrigY, Length, Height
If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
  Feld = 10
EndIf

;Buttons testen/aktualisieren
SetFont FontSchrift
Color 0, 0, 0

Restore ChooseFktButton
Read OrigX, OrigY, Length, Height
DrawImage ButtonDefault, OrigX, OrigY, 0 ;normaler Button
Text OrigX+BreiteButton/2, OrigY+HoeheButton/2, "Auswahl", 1, 1
If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
  If MouseDown(2) Then ;Kontexthilfe (V 2.0)
    HelpBubble(x, y, "Menü zum Auswählen$voreingestellter Funktionen")
    SetFont FontSchrift
  ElseIf MouseDown(1)
    DrawImage ButtonDefault, OrigX, OrigY, 1 ;aktivierter Button
    Text OrigX+BreiteButton/2+2, OrigY+HoeheButton/2+2, "Auswahl", 1, 1
    Feld = 1
  EndIf
EndIf

Restore ResetKoeffButton
Read OrigX, OrigY, Length, Height
DrawImage ButtonDefault, OrigX, OrigY, 0
If FktId<>0 Then ;Steht nur bei vorgegebener
Funktion zur Verfügung
  Text OrigX+BreiteButton/2, OrigY+HoeheButton/2, "Reset", 1, 1
  If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
    If MouseDown(2) Then
      HelpBubble(x, y, "Koeffizienten auf ursprüngliche$Werte zurücksetzen")
      SetFont FontSchrift
    ElseIf MouseDown(1) Then
      DrawImage ButtonDefault, OrigX, OrigY, 1
      Text OrigX+BreiteButton/2+2, OrigY+HoeheButton/2+2, "Reset", 1, 1
      Feld = 3
    EndIf
  EndIf
Else ;Button deaktiviert
  If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length And
MouseDown(2) Then
    HelpBubble(x, y, "Funktion in diesem$Modus nicht verfügbar")
    SetFont FontSchrift
  EndIf

```

```

Color GuiR*1.1, GuiG*1.1, GuiB*1.1
Text OrigX+BreiteButton/2+1, OrigY+HoeheButton/2+1, "Reset", 1, 1
Color GuiR*0.8, GuiG*0.8, GuiB*0.8
Text OrigX+BreiteButton/2, OrigY+HoeheButton/2, "Reset", 1, 1
Color 0, 0, 0
EndIf

Restore SetKoefZeroButton
Read OrigX, OrigY, Length, Height
DrawImage ButtonDefault, OrigX, OrigY, 0
Text OrigX+BreiteButton/2, OrigY+HoeheButton/2, "Null setzen", 1, 1
If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
    If MouseDown(2) Then
        HelpBubble(x, y, "Alle Koeffizienten Null setzen")
        SetFont FontSchrift
    ElseIf MouseDown(1) Then
        DrawImage ButtonDefault, OrigX, OrigY, 1
        Text OrigX+BreiteButton/2+2, OrigY+HoeheButton/2+2, "Null setzen", 1, 1
        Feld = 4
    EndIf
EndIf

Restore FreeFktButton
Read OrigX, OrigY, Length, Height
DrawImage ButtonDefault, OrigX, OrigY, 0
Text OrigX+BreiteButton/2, OrigY+HoeheButton/2, "Freie Fkt.", 1, 1
If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
    If MouseDown(2) Then
        HelpBubble(x, y, "Freie Wahl der Koeffizienten")
        SetFont FontSchrift
    ElseIf MouseDown(1) Then
        DrawImage ButtonDefault, OrigX, OrigY, 1
        Text OrigX+BreiteButton/2+2, OrigY+HoeheButton/2+2, "Freie Fkt.", 1, 1
        Feld = 2
    EndIf
EndIf

Restore StartAnimButton
Read OrigX, OrigY, Length, Height
DrawImage AnimButton, OrigX, OrigY, 0
DrawImage StartAnimButton, OrigX+Length/2, OrigY+Height/2
If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
    If MouseDown(2) Then
        HelpBubble(x, y, "Animation abspielen")
        SetFont FontSchrift
    ElseIf MouseDown(1) Then
        DrawImage AnimButton, OrigX, OrigY, 1
        DrawImage StartAnimButton, OrigX+Length/2+2, OrigY+Height/2+2
        Delay DelayButton
        AktualAll=True
        FlushKeys
        Feld = 6
    EndIf
EndIf

Restore StopAnimButton
Read OrigX, OrigY, Length, Height
DrawImage AnimButton, OrigX, OrigY, 0
DrawImage StopAnimButton, OrigX+Length/2, OrigY+Height/2
If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
    If MouseDown(2) Then
        HelpBubble(x, y, "Animation anhalten")
    EndIf
EndIf

```

```

        SetFont FontSchrift
    ElseIf MouseDown(1)
        DrawImage AnimButton, OrigX, OrigY, 1
        DrawImage StopAnimButton, OrigX+Length/2+2, OrigY+Height/2+2
        Delay DelayButton
        AktualAll=True
        FlushKeys
        Feld = 7
    EndIf
EndIf

Restore EndButton
Read OrigX, OrigY, Length, Height
DrawImage ButtonDefault, OrigX, OrigY, 0
Text OrigX+BreiteButton/2, OrigY+HoeheButton/2, "Beenden", 1, 1
If y>OrigY And y<OrigY+Height And x>OrigX And x<OrigX+Length Then
    If MouseDown(2) Then
        HelpBubble(x, y, "Programm verlassen")
        SetFont FontSchrift
    ElseIf MouseDown(1) Then
        DrawImage ButtonDefault, OrigX, OrigY, 1
        Text OrigX+BreiteButton/2+2, OrigY+HoeheButton/2+2, "Beenden", 1, 1
        Feld = 8
    EndIf
EndIf
EndIf

```

EndIf

```

;Dauer bis zum Zurücksetzen eines gedrückten Buttons
If Feld<>10 And Feld<>5 And Feld<>0 Then ;Button gedrückt
    Flip 1
    Delay DelayButton
EndIf

```

Return Feld

End Function

;

Function SetDelayAnim(Klick=False)

```

    Restore SetDelayAnim
    Read OrigX, OrigY, Length, Height

    Viewport OrigX-DickeRahmen, OrigY-DickeRahmen, Length+DickeRahmen*2+1,
Height+DickeRahmen*2+1
    Origin OrigX, OrigY
    ClsColor 255, 255, 255
    Cls

    ;Rahmen
    Color GuiR*1.05, GuiG*1.05, GuiB*1.05
    Line -DickeRahmen, Height+DickeRahmen, Length+DickeRahmen, Height+DickeRahmen
    Line Length+DickeRahmen, Height+DickeRahmen, Length+DickeRahmen, -DickeRahmen
    Color GuiR*1.1, GuiG*1.1, GuiB*1.1
    Line -DickeRahmen+1, Height+DickeRahmen-1, Length+DickeRahmen-1,
Height+DickeRahmen-1
    Line Length+DickeRahmen-1, Height+DickeRahmen-1, Length+DickeRahmen-1,
-DickeRahmen+1
    Color GuiR*0.7, GuiG*0.7, GuiB*0.7
    Line -DickeRahmen, Height+DickeRahmen, -DickeRahmen, -DickeRahmen

```



```

Line -DickeRahmen, -DickeRahmen, Length+DickeRahmen, -DickeRahmen
Color GuiR*0.4, GuiG*0.4, GuiB*0.4
Line -DickeRahmen+1, Height+DickeRahmen-1, -DickeRahmen+1, -DickeRahmen+1
Line -DickeRahmen+1, -DickeRahmen+1, Length+DickeRahmen-1, -DickeRahmen+1

```

```

SetFont FontSchriftGross
Color 0, 0, 0

```

```

If Klick Then
    Locate OrigX+1, OrigY+1
    DelayAnim=Input() : FlushKeys
    If Numk=0 Then Numk=1
Else
    Text Length/2, Height/2, DelayAnim, 1, 1
EndIf

```

```

Viewport 0, 0, AuflX, AuflY
Origin 0, 0

```

```

;Trennlinie
Color GuiR*0.7, GuiG*0.7, GuiB*0.7
Line OrigX-20, OrigY-45, OrigX+110, OrigY-45
Color GuiR*1.1, GuiG*1.1, GuiB*1.1
Line OrigX-20, OrigY-44, OrigX+110, OrigY-44

```

```

Color GuiR*0.7, GuiG*0.7, GuiB*0.7
Line OrigX-20, OrigY+70, OrigX+110, OrigY+70
Color GuiR*1.1, GuiG*1.1, GuiB*1.1
Line OrigX-20, OrigY+71, OrigX+110, OrigY+71

```

```

Color 0, 0, 0

```

```

Text OrigX-20, OrigY-40, "Animation (var. k)"
SetFont FontBeschriftung
Text OrigX-10, OrigY-20, "Verzögerung in ms:"

```

```

Return DelayAnim

```

```

End Function

```

```

;

```

```

Function LoadKoef(FktId)

```

```

    Local HoeheFkt#=1.0

```

```

    aSoll#(0)=0

```

```

    Select FktId
    Case 1 ;Rechteck
        AppTitle Title$ + " - Rechteckschwingung"
        ;Koeffizienten einstellen
        For k=1 To MaxNumk
            If k Mod 2 <> 0 Then ;k ungerade
                bSoll#(k)=4.0*HoeheFkt#*1/Pi/k
            Else
                bSoll#(k)=0
            EndIf
            aSoll#(k)=0
        Next

```

```

    ;Vorgegebene Funktionswerte setzen

```

```

For x=0 To 179
    FktY#(x)=1*HoeheFkt#
Next
For x=180 To 360
    FktY#(x)=-1*HoeheFkt#
Next

Case 2                                ;Sinusimpuls
AppTitle Title$ + " - Sinusimpuls"
; aSoll#(0)=4*HoeheFkt#*0.7/Pi
aSoll#(0)=4*HoeheFkt#/Pi
For k=1 To MaxNumk
    If k Mod 2 = 0 Then                ;k gerade
        aSoll#(k)=-4*HoeheFkt#*0.7/Pi/((k)^2-1)
        aSoll#(k)=-4*HoeheFkt#/Pi/((k)^2-1)
    Else
        aSoll#(k)=0
    EndIf
    bSoll#(k)=0
Next

For x=0*PrecSin To 180*PrecSin
    FktY#(x/PrecSin)=HoeheFkt#*Sinus#(x)
Next
For x=180*PrecSin To 360*PrecSin
    FktY#(x/PrecSin)=-HoeheFkt#*Sinus#(x)
Next

Case 3                                ;Kippschwingung
AppTitle Title$ + " - Kippschwingung"
For k=1 To MaxNumk
    bSoll#(k)=-2*HoeheFkt#/Pi/k
    aSoll#(k)=0
Next

For x=0 To 360
    FktY#(x)=2*HoeheFkt#*x/360-HoeheFkt#
Next

Case 4                                ;Dreieck
AppTitle Title$ + " - Dreieckschwingung"
For k=1 To MaxNumk
    If k Mod 2 <> 0 Then                ;k ungerade
        aSoll#(k)=HoeheFkt#*2.5/Pi/k^2
    Else
        aSoll#(k)=0
    EndIf
    bSoll#(k)=0
Next

For x=0 To 179
    FktY#(x)=-2*HoeheFkt#*x/180+HoeheFkt#
Next
For x=180 To 360
    FktY#(x)=2*HoeheFkt#*(x-180)/180-HoeheFkt#
Next

Case 5                                ;Sägezahn
AppTitle Title$ + " - Sägezahn"
aSoll#(0)=Pi/2*HoeheFkt#/Pi
For k=1 To MaxNumk
    aSoll#(k)=((-1.0)^k-1)/k^2/Pi/Pi*HoeheFkt#

```

```

    bSoll#(k)=((-1)^(k-1))/k/Pi*HoeheFkt#
Next

```

```

For x=0 To 179
    FktY#(x)=HoeheFkt#*x/180
Next
For x=180 To 360
    FktY#(x)=0
Next

```

```

Case 6 ;e-Funktion
AppTitle Title$ + " - Exponentialfunktion"
aSoll#(0)=HoeheFkt#*(Exp#(Pi)-Exp#(-Pi))/Pi/10 -2
For k=1 To MaxNumk
    aSoll#(k)=((-1.0)^k*2*(Exp#(Pi)-Exp#(-Pi))/2/Pi)/(1+k^2) *HoeheFkt#/10
    bSoll#(k)=-((-1.0)^k*2*k*(Exp#(Pi)-Exp#(-Pi))/2/Pi)/(1+k^2) *HoeheFkt#/10
Next

For x=0 To 180
    FktY#(x)=HoeheFkt#*Exp#(Float#(x)/180*Pi)/10 -1
Next
For x=180 To 360
    FktY#(x)=HoeheFkt#*Exp#(Float#(x-360)/180*Pi)/10 -1
Next

```

```

End Select

```

```

End Function

```

```

;

```

```

Function AddKoefNum(Klick=False)

```

```

    Restore AddKoefNum
    Read OrigX, OrigY, Length, Height

```

```

    Viewport OrigX-DickeRahmen, OrigY-DickeRahmen, Length+DickeRahmen*2+1,
Height+DickeRahmen*2+1
    Origin OrigX, OrigY
    ClsColor 255, 255, 255
    Cls

```

```

;Rahmen
Color GuiR*1.05, GuiG*1.05, GuiB*1.05
Line -DickeRahmen, Height+DickeRahmen, Length+DickeRahmen, Height+DickeRahmen
Line Length+DickeRahmen, Height+DickeRahmen, Length+DickeRahmen, -DickeRahmen
Color GuiR*1.1, GuiG*1.1, GuiB*1.1
Line -DickeRahmen+1, Height+DickeRahmen-1, Length+DickeRahmen-1,
Height+DickeRahmen-1
Line Length+DickeRahmen-1, Height+DickeRahmen-1, Length+DickeRahmen-1,
-DickeRahmen+1
Color GuiR*0.7, GuiG*0.7, GuiB*0.7
Line -DickeRahmen, Height+DickeRahmen, -DickeRahmen, -DickeRahmen
Line -DickeRahmen, -DickeRahmen, Length+DickeRahmen, -DickeRahmen
Color GuiR*0.4, GuiG*0.4, GuiB*0.4
Line -DickeRahmen+1, Height+DickeRahmen-1, -DickeRahmen+1, -DickeRahmen+1
Line -DickeRahmen+1, -DickeRahmen+1, Length+DickeRahmen-1, -DickeRahmen+1

```

```

SetFont FontSchriftGross
Color 0, 0, 0

```

```

If Klick Then

```

```

Locate OrigX+1, OrigY+1
FlushKeys
Numk=Input() : FlushKeys
If Numk=0 Then Numk=1
If Numk>MaxNumk Then Numk=MaxNumk           ;k im zulässigen Bereich halten
If Anim Then NumkTemp=Numk
Else
    Text Length/2, Height/2, Numk, 1, 1
EndIf

```

```

Viewport 0, 0, AuflX, AuflY
Origin 0, 0

```

```

;Trennlinie
Color GuiR*0.7, GuiG*0.7, GuiB*0.7
Line OrigX-20, OrigY-45, OrigX+110, OrigY-45
Color GuiR*1.1, GuiG*1.1, GuiB*1.1
Line OrigX-20, OrigY-44, OrigX+110, OrigY-44

```

```

Color 0, 0, 0

```

```

Text OrigX-10, OrigY-30, "Erweitertes k:"
SetFont FontBeschriftung
Text OrigX+Length+13, OrigY+Height/2, "von max. " + MaxNumk, 0, 1

```

```

Return Numk

```

```

End Function

```

```

;

```

```

Function HelpBubble(x, y, Help$)

```

```

    SetFont FontBeschriftung

```

```

    Local BreiteRand=4

```

```

;Text in Zeilen einlesen

```

```

set=0

```

```

lines=1

```

```

Repeat

```

```

    set=set+1

```

```

    t$=Mid$(Help$, set, 1)

```

```

    If t$="$" Then           ;Zeilenwechsel

```

```

        Lines=Lines+1

```

```

    ElseIf t$<>""

```

```

        Helptext$(Lines)=Helptext$(Lines)+t$

```

```

    EndIf

```

```

Until t$=""

```

```

;Max. Breite des Textes ermitteln

```

```

LenMax=0

```

```

For row=1 To lines

```

```

    Length=StringWidth(Helptext$(row))

```

```

    If Length>LenMax Then

```

```

        LenMax=Length

```

```

    EndIf

```

```

Next

```

```

If x+LenMax+10 > AuflX Then

```

```

    Modus=-1           ;Text links vom Cursor

```

```

Else

```

```

    Modus=0           ;rechts

```

```

EndIf

Color BubbleR, BubbleG, BubbleB
Rect x + Modus*(LenMax+2*BreiteRand), y - 20, LenMax+2*BreiteRand,
lines*15+2*BreiteRand, 1
Color 0, 0, 0
Rect x + Modus*(LenMax+2*BreiteRand), y - 20, LenMax+2*BreiteRand,
lines*15+2*BreiteRand, 0

For row=1 To lines
    Text x + Modus*(LenMax+2*BreiteRand)+BreiteRand, y-20+BreiteRand+1+(row-
1)*15, Helptext$(row)
Next

For row=1 To lines
    Helptext$(row)=""
Next

Flip 1
;Warten bis Timeout oder Maus weiterbewegt oder Mausklick
tick=MilliSecs()
Repeat
    WaitTimer(Takt)
Until MilliSecs()-tick>DelayBubble * lines Or (MouseX()-x)^2+(MouseY()-y)^2 >
BubbleMotion^2 Or MouseDown(1)
FlushMouse
AktualAll=True    ;Fenster neu aufbauen

End Function

Function Spektrum()
Viewport 0, 0, 500, 700
Origin 30, 100
fy#=100.0
Cls
Color 0, 0, 0
Line 0, -10, 0, 10
Line 0, 0, 400, 0
Line -10, -fy#, 0, -fy#
Line -10, fy#, 0, fy#
Line -10, -fy#/2, 0, -fy#/2
Line -10, fy#/2, 0, fy#/2
Color 200, 20, 10
For k=0 To 20
If aSoll#(k)>=0Then
Rect k*20, -aSoll#(k)*fy#-2, 10, aSoll#(k)*fy#+2, 1
Else
Rect k*20, -2, 10, -aSoll#(k)*fy#+2, 1
EndIf
Next

Origin 30, 300
fy#=100.0
Color 0, 0, 0
Line 0, -10, 0, 10
Line 0, 0, 400, 0
Line -10, -fy#, 0, -fy#
Line -10, fy#, 0, fy#
Line -10, -fy#/2, 0, -fy#/2
Line -10, fy#/2, 0, fy#/2
Color 200, 20, 10
For k=0 To 20

```

```

If bSoll#(k)>=0Then
Rect k*20, -bSoll#(k)*fy#-2, 10, bSoll#(k)*fy#+2, 1
Else
Rect k*20, -2, 10, -bSoll#(k)*fy#+2, 1
EndIf
Next

```

```

Flip 1
Delay 1000
WaitKey()
End Function

```

```

;Labels -----

```

```

.GetValues
Data 20, AuflY/2+10, 2*(MaxNumkField+1)*BreiteSlide, AuflY/2-30

```

```

.DrawFunction
Data 20-DickeRahmen, 15, AuflX-200, AuflY/2-40
Data AuflX-200+20+10, 15, 150, 90

```

```

.ChooseFunction
Data 60, 60, 3, 2

```

```

.ChooseFktButton
Data AuflX-150, AuflY/2+10, 114, 37

```

```

.ResetKoefButton
Data AuflX-150, AuflY/2+100, 114, 37

```

```

.SetKoefZeroButton
Data AuflX-150, AuflY/2+145, 114, 37

```

```

.FreeFktButton
Data AuflX-150, AuflY/2+55, 114, 37

```

```

.EndButton
Data AuflX-150, AuflY/2-60, 114, 37

```

```

.StartAnimButton
Data AuflX-150, AuflY/2-115, 32, 28

```

```

.StopAnimButton
Data AuflX-110, AuflY/2-115, 32, 28

```

```

.SetDelayAnim
Data AuflX-140, AuflY/2-145, 30, 16

```

```

.AddKoefNum
Data AuflX-140, AuflY-50, 30, 16

```