```
//**************************************
//        MACRO: PLEADING.WCM
//        PURPOSE: Creates/modifies pleading paper.
//**************************************
Application (A1; "WordPerfect"; Default; "EN")

Call InitializeVariables ()
Call GetSettings ()
Call SetEnvironment ()
Call EditInitialStyle ()
Call DefineMainDlg ()
Call DisplayMainDlg ()
Call DismissMainDlg ()
Call CloseInitialStyle (1)
Call CalcLineHeight ()
Call EditInitialStyle ()
Call InsertWatermark ()
Call CloseInitialStyle (1)
Call SaveSettings ()
BookmarkFind (BookmarkName)
BookmarkDelete (BookmarkName)
Quit
```

```
//**************************************
//        ROUTINE: InitializeVariables
//        PURPOSE: Initializes macro variables.
//**************************************
LABEL (InitializeVariables)
StartNum := 1                       // Default starting line number.
StopNum := 28                       // Default ending line number.
NumbersOn := 1                      // Default line numbering (0 = Off, 1 = On).
LinesLeft := 2                      // Default lines on left side.
LinesRight := 1                     // Default lines on right side.
MaxLines := 48                      // Maximum number of pleading lines.
NumToLineSpace := 0.094"            // Spacing between numbers and left vertical line.
LineToTextSpace := 0.094"           // Spacing between vertical lines and text.
VertLineInset := .188"          // Vertical line inset from top and bottom of page.
BookmarkName := "_Pleading"
RegistryKey := "Software\PerfectOffice\WordPerfect\9\Template Experts\Pleading"
TextSpacing := 2
RETURN

//**************************************
//        ROUTINE: SetEnvironment
//        PURPOSE: Sets state of WPWin before beginning macro.
//**************************************
```

```
LABEL (SetEnvironment)
While (?Substructure)
        SubstructureExit ()
EndWhile
If (?BlockActive <>0)
        SelectMode (Off!)
EndIf
BookmarkCreate (BookmarkName)
RETURN

//**************************************
//      ROUTINE: GetSettings
//      PURPOSE: Restores pleading settings.
//**************************************
LABEL (GetSettings)
hKey := RegistryOpenKey (CurrentUser!; RegistryKey)
If (hKey)
        StartNum := RegistryQueryValue (hKey; "Start number")
        StopNum := RegistryQueryValue (hKey; "Stop number")
        NumbersOn := RegistryQueryValue (hKey; "Print line numbers")
        LinesLeft := RegistryQueryValue (hKey; "Lines on left side")
        LinesRight := RegistryQueryValue (hKey; "Lines on right side")
EndIf
```

```
RegistryCloseKey (hKey)
RETURN

//*************************************
//        ROUTINE: DefineMainDlg
//        PURPOSE: Defines main dialog.
//*************************************
LABEL (DefineMainDlg)
Justify := GetJustification ()
DlgWidth := 190
DlgHeight := 149
ButtonWidth := 55
ButtonHPos := DlgWidth-ButtonWidth-8
GroupBoxWidth := DlgWidth-ButtonWidth-24
CounterWidth := 34
CounterHPos := ButtonHPos-CounterWidth-18
DialogDefine ("Pleading"; 50; 50; DlgWidth; DlgHeight; Percent!; "Pleading Paper")
DialogAddPushButton ("Pleading"; "OKBttn"; ButtonHPos; 8; ButtonWidth; 14; OKBttn!|
DefaultBttn!; "OK")
DialogAddPushButton ("Pleading"; "CancelBttn"; ButtonHPos; 26; ButtonWidth; 14;
CancelBttn!|NonDefaultBttn!; "Cancel")
DialogAddPushButton ("Pleading"; "Margins"; ButtonHPos; 44; ButtonWidth; 14; 0;
"&Margins...")
```

DialogAddPushButton ("Pleading"; "Font"; ButtonHPos; 62; ButtonWidth; 14; 0; "&Font...")

DialogAddPushButton ("Pleading"; "Numbering"; ButtonHPos; 80; ButtonWidth; 14; 0; "Pg &Numbers...")

DialogAddPushButton ("Pleading"; "Spacing"; ButtonHPos; 98; ButtonWidth; 14; 0; "Line &Spacing...")

DialogAddText ("Pleading"; "Justification"; ButtonHPos; 116; ButtonWidth; 10; Left!; "&Justification:")

DialogAddComboBox ("Pleading"; "JustifyCB"; ButtonHPos; 127; ButtonWidth; 60; DropList!; Justify)

DialogAddListItem ("Pleading"; "JustifyCB"; "Left")

DialogAddListItem ("Pleading"; "JustifyCB"; "Right")

DialogAddListItem ("Pleading"; "JustifyCB"; "Center")

DialogAddListItem ("Pleading"; "JustifyCB"; "Full")

DialogAddListItem ("Pleading"; "JustifyCB"; "All")

DialogAddGroupBox ("Pleading"; "NumGrp"; 8; 5; GroupBoxWidth; 73; "Line numbering")

DialogAddCheckBox ("Pleading"; "Numbers"; 18; 18; 85; 11; "&Display line numbers"; NumbersOn)

DialogAddText ("Pleading"; "Start"; 18; 38; 50; 10; 1; "Star&t number:")

DialogAddCounter ("Pleading"; "StartCount"; CounterHPos; 36; CounterWidth; 15; DisplayNormal!|AutoValidate!; StartNum; 1; MaxLines; 1)

DialogAddText ("Pleading"; "Stop"; 18; 56; 50; 10; 1; "Sto&p number:")

DialogAddCounter ("Pleading"; "StopCount"; CounterHPos; 54; CounterWidth; 15; DisplayNormal!|AutoValidate!; StopNum; 1; MaxLines; 1)

```
DialogAddGroupBox ("Pleading"; "LineGrp"; 8; 85; GroupBoxWidth; 55; "Vertical lines")
DialogAddText ("Pleading"; "LinesLeft"; 18; 100; 50; 10; 1; "Lines on &left:")
DialogAddCounter ("Pleading"; "LeftCount"; CounterHPos; 98; CounterWidth; 15;
DisplayNormal!|AutoValidate!; LinesLeft; 0; 2; 1)
DialogAddText ("Pleading"; "LinesRight"; 18; 118; 50; 10; 1; "Lines on &right:")
DialogAddCounter ("Pleading"; "RightCount"; CounterHPos; 116; CounterWidth; 15;
DisplayNormal!|AutoValidate!; LinesRight; 0; 2; 1)
RETURN

//**************************************
//      ROUTINE: DisplayMainDlg
//      PURPOSE: Displays main dialog and begins message loop.
//**************************************
LABEL (DisplayMainDlg)
Display (On!)
DialogShow ("Pleading"; "StartCount"; MainMsgLoop)
NotDone := True
While (NotDone)
EndWhile
RETURN

//**************************************
//      ROUTINE: MainMsgLoop
```

```
//          PURPOSE: Processes messages in main dialog.
//*************************************
LABEL (MainMsgLoop)
If ((MainMsgLoop[5] = 274) Or (MainMsgLoop[3] = "CancelBttn"))
        Call CloseInitialStyle (0)
        Call DismissMainDlg ()
        BookmarkFind (BookmarkName)
        BookmarkDelete (BookmarkName)
        Quit
EndIf
Switch (MainMsgLoop[3])
        CaseOf "OKBttn": NotDone := False
        CaseOf "Margins": FormatMarginsDlg ()
        CaseOf "Font": FontDlg ()
        CaseOf "Spacing":
                LineSpacingDlg ()
                TextSpacing := ?LineSpacing
        CaseOf "Numbering":
                PageNumberingDlg ()
                If (?PageNumberingOn)
                        MessageBox (Response; "Suppress Page Number"; "Suppress page
                        number on first page?"; YesNo!|IconQuestion!)
                        If (Response = 6)
```

```
                    If (?PageSuppress <>1)
                        Suppress (PageNumbering!)
                    EndIf
                Else
                    SuppressOff ()
                EndIf
            EndIf
        CaseOf "JustifyCB":
            Justify := RegionGetWindowText ("Pleading.JustifyCB")
            Switch (Justify)
                CaseOf "Left": JustifyLeft ()
                CaseOf "Right": JustifyRight ()
                CaseOf "Center": JustifyCenter ()
                CaseOf "Full": JustifyFull ()
                CaseOf "All": JustifyAll ()
                Default: JustifyFull ()
            EndSwitch
EndSwitch
RETURN

//*************************************
//      ROUTINE: DismissMainDlg
//      PURPOSE: Dismisses main dialog.
```

```
//**************************************

LABEL (DismissMainDlg)

DialogDismiss ("Pleading"; "OKBttn")

DialogDestroy ("Pleading")

RETURN


//**************************************

//        ROUTINE: InsertWatermark

//        PURPOSE: Insert lines and numbers in Watermark B.

//**************************************

LABEL (InsertWatermark)

Advance (AdvanceFromTop!; MinLine)       // Set vertical starting point.

WidowOrphan (Off!)

WatermarkB (Create!)

        DialogDefine ("Wait"; 50; 50; 150; 36; Percent!; "Please Wait")

        DialogAddText ("Wait"; "WaitStatic"; 8; 14; 134; 18; Left!; "Generating pleading...")

        DialogShow ("Wait"; ""; WaitMsgLoop)

        LineHeight (Integer (PrintSpace/LinesNeeded))

        TextShade (100)

        LineSpacing (2)

        LineSpacing (TextSpacing)

        ColumnWidth := MarginLeft()-NumToLineSpace-LineToTextSpace-?

                UnprintableLeftMargin
```

```
        ColumnsDefinition (ColumnsType: Parallel!; VerticalSpacing: 1.0; {Spacing:
                ColumnWidth; SpacingDef: Fixed!; Spacing: NumToLineSpace +
                NumToLineSpace; SpacingDef: Fixed!; Spacing: 5.9"; SpacingDef: NotFixed!})
        MarginLeft (?UnprintableLeftMargin)
        Call InsertVerticalLines ()
        Call InsertLineNumbers ()
        DialogDismiss ("Wait"; "WaitStatic")
        DialogDestroy ("Wait")
SubstructureExit ()
LineSpacing (TextSpacing)
RETURN

LABEL (WaitMsgLoop)
RETURN

//**************************************
//      ROUTINE: InsertVerticalLines
//      PURPOSE: Inserts vertical lines in watermark.
//**************************************
LABEL (InsertVerticalLines)
If (LinesLeft>0)
        GraphicsLineCreate ()
                GraphicsLineType (Vertical!)
```

```
            Switch (LinesLeft)
                CaseOf 1: GraphicsLineStyle (SingleLine!)
                CaseOf 2: GraphicsLineStyle (DoubleLine!)
                Default: GraphicsLineStyle (DoubleLine!)
            EndSwitch
            GraphicsLineHorizontalPosition (Position: BetweenColumns!; LeftColNum: 1)
            GraphicsLineVerticalPosition (Position: Set!; Where: VertLineInset)
            GraphicsLineLength (LineLength (VertLineInset))
        GraphicsLineEnd (Save!)
EndIf
If (LinesRight>0)
        GraphicsLineCreate ()
            GraphicsLineType (Vertical!)
            Switch (LinesRight)
                CaseOf 1: GraphicsLineStyle (SingleLine!)
                CaseOf 2: GraphicsLineStyle (DoubleLine!)
                Default: GraphicsLineStyle (SingleLine!)
            EndSwitch
            GraphicsLineSpacing (TopSpace: LineToTextSpace)
            GraphicsLineHorizontalPosition (Position: BetweenColumns!; LeftColNum: 2)
            GraphicsLineVerticalPosition (Position: Set!; Where: VertLineInset)
            GraphicsLineLength (LineLength (VertLineInset))
        GraphicsLineEnd (Save!)
```

```
EndIf
RETURN

//*************************************
//      ROUTINE: InsertLineNumbers
//      PURPOSE: Insert line numbers in watermark.
//*************************************
LABEL (InsertLineNumbers)
If (NumbersOn)
        Advance (AdvanceFromTop!; MinLine)
        JustifyRight ()
        If (StartNum <= StopNum)
                For (Count; StartNum; Count < StopNum; Count + 1)
                        Type (Count)
                        HardReturn ()
                EndFor
        Else
                For (Count; StartNum; Count>StopNum; Count-1)
                        Type (Count)
                        HardReturn ()
                EndFor
        EndIf
        Type (StopNum)
```

```
EndIf
RETURN

//**************************************
//      ROUTINE: CalcLineHeight
//      PURPOSE: Calculates and sets line height.
//**************************************
LABEL (CalcLineHeight)
PosDocTop ()
HardPageBreak ()
Advance (AdvanceDown!; 27")
MaxLine := ?Line
PosCharPrevious ()
MinLine := ?Line
DeleteCharNext ()
DeleteCharPrevious ()
PrintSpace := MaxLine-MinLine
If (StartNum <= StopNum)
        LinesNeeded := 2* (StopNum-StartNum) + 1
Else
        LinesNeeded := 2* (StartNum-StopNum) + 1
EndIf
PosDocVeryTop ()
```

```
LineHeight (Integer (PrintSpace/LinesNeeded))
RETURN

//*************************************
//      ROUTINE: SaveSettings
//      PURPOSE: Saves pleading settings.
//*************************************
LABEL (SaveSettings)
hKey := RegistryCreateKey (CurrentUser!; RegistryKey)
RegistrySetValue (hKey; "Start number"; StartNum; DWord!)
RegistrySetValue (hKey; "Stop number"; StopNum; DWord!)
RegistrySetValue (hKey; "Print line numbers"; NumbersOn; DWord!)
RegistrySetValue (hKey; "Lines on left side"; LinesLeft; DWord!)
RegistrySetValue (hKey; "Lines on right side"; LinesRight; DWord!)
RegistryCloseKey (hKey)
RETURN

//*************************************
//      PROCEDURE SuppressOff
//      PURPOSE: Removes any suppress codes in current editing window.
//*************************************
PROCEDURE SuppressOff ()
PosDocVeryTop ()
```

```
SearchString (StrgToLookFor: "[Suppress]")
ReplaceString (RplcStrg: "")
ReplaceForward (SearchMode: Regular!)
ENDPROC

//*************************************
//      PROCEDURE EditInitialStyle
//      PURPOSE: Opens initial style for editing.
//*************************************
PROCEDURE EditInitialStyle ()
StyleEditBegin (DocStyle!; CurrentDoc!)
StyleCodes (WithoutOffCodes!; CurrentDoc!)
PosDocBottom ()
ENDPROC

//*************************************
//      PROCEDURE CloseInitialStyle (Save)
//      PURPOSE: Closes initial style (0 = Cancel changes, 1 = Save changes).
//*************************************
PROCEDURE CloseInitialStyle (Save)
Display (Off!)
SubstructureExit ()
Switch (Save)
```

```
        CaseOf 0: StyleEditEnd (Cancel!)
        CaseOf 1: StyleEditEnd (Save!)
        Default: StyleEditEnd (Save!)
EndSwitch
ENDPROC

//*************************************
//      FUNCTION: GetJustification
//      PURPOSE: Returns name of current justification setting.
//*************************************
FUNCTION GetJustification ()
Switch (?Justification)
        CaseOf 0: Justify := "Left"
        CaseOf 1: Justify := "Full"
        CaseOf 2: Justify := "Center"
        CaseOf 3: Justify := "Right"
        CaseOf 4: Justify := "All"
        Default: Justify := "Full"
EndSwitch
Return (Justify)
ENDFUNC

//*************************************
```

```
//      FUNCTION: LineLength
//      PURPOSE: Returns length of vertical pleading lines.
//**************************************
FUNCTION LineLength (VertLineInset)
LineLength := ?PaperLength- (2*Integer (VertLineInset*1200))
Return (LineLength)
ENDFUNC
```