Talking about PowerPoint, you would use a VBA Macro to do the job, something like

Sub Pull()

Dim SrcDir As String, SrcFile As String

 SrcDir = PickDir()

 If SrcDir = "" Then Exit Sub

 SrcFile = Dir(SrcDir & "\\*.ppt")

 Do While SrcFile <> ""

 ImportFromPPT SrcDir + "\" + SrcFile, 1, 2

 SrcFile = Dir()

 Loop

End Sub

Selecting your source directory you can use this function

Private Function PickDir() As String

Dim FD As FileDialog

 PickDir = ""

 Set FD = Application.FileDialog(msoFileDialogFolderPicker)

 With FD

 .Title = "Pick a directory to work on"

 .AllowMultiSelect = False

 .Show

 If .SelectedItems.Count <> 0 Then

 PickDir = .SelectedItems(1)

 End If

 End With

End Function

Now - the main point is inserting slides from another PPT while **preserving the source format**. This is a tricky thing, as the PPT VBA InsertFromFile method is of no good use. Microsoft gave us good time to figure it out the hard way in countless 20hrs debuging sessions :-) and you need to type a lot of code to get it done correctly - far more complicated than using the dialogue manually, in particular if your source slide deviates from your source master slide.

If your PPT's are sticking to their masters, you can safely omit all code between the ">>>>"

Private Sub ImportFromPPT(FileName As String, SlideFrom As Long, SlideTo As Long)

Dim SrcPPT As Presentation, SrcSld As Slide, Idx As Long, SldCnt As Long

 Set SrcPPT = Presentations.Open(FileName, , , msoFalse)

 SldCnt = SrcPPT.Slides.Count

 If SlideFrom > SldCnt Then Exit Sub

 If SlideTo > SldCnt Then SlideTo = SldCnt

 For Idx = SlideFrom To SlideTo Step 1

 Set SrcSld = SrcPPT.Slides(Idx)

 SrcSld.Copy

 With ActivePresentation.Slides.Paste

 .Design = SrcSld.Design

 .ColorScheme = SrcSld.ColorScheme

 ' if slide is not following its master (design, color scheme)

 ' we must collect all bits & pieces from the slide itself

 ' >>>>>>>>>>>>>>>>>>>>

 If SrcSld.FollowMasterBackground = False Then

 .FollowMasterBackground = False

 .Background.Fill.Visible = SrcSld.Background.Fill.Visible

 .Background.Fill.ForeColor = SrcSld.Background.Fill.ForeColor

 .Background.Fill.BackColor = SrcSld.Background.Fill.BackColor

 ' inspect the FillType object

 Select Case SrcSld.Background.Fill.Type

 Case Is = msoFillTextured

 Select Case SrcSld.Background.Fill.TextureType

 Case Is = msoTexturePreset

 .Background.Fill.PresetTextured (SrcSld.Background.Fill.PresetTexture)

 Case Is = msoTextureUserDefined

 ' TextureName gives a filename w/o path

 ' not implemented, see picture handling

 End Select

 Case Is = msoFillSolid

 .Background.Fill.Transparency = 0#

 .Background.Fill.Solid

 Case Is = msoFillPicture

 ' picture cannot be copied directly, need to export and re-import slide image

 If SrcSld.Shapes.Count > 0 Then SrcSld.Shapes.Range.Visible = False

 bMasterShapes = SrcSld.DisplayMasterShapes

 SrcSld.DisplayMasterShapes = False

 SrcSld.Export SrcPPT.Path & SrcSld.SlideID & ".png", "PNG"

 .Background.Fill.UserPicture SrcPPT.Path & SrcSld.SlideID & ".png"

 Kill (SrcPPT.Path & SrcSld.SlideID & ".png")

 SrcSld.DisplayMasterShapes = bMasterShapes

 If SrcSld.Shapes.Count > 0 Then SrcSld.Shapes.Range.Visible = True

 Case Is = msoFillPatterned

 .Background.Fill.Patterned (SrcSld.Background.Fill.Pattern)

 Case Is = msoFillGradient

 ' inspect gradient type

 Select Case SrcSld.Background.Fill.GradientColorType

 Case Is = msoGradientTwoColors

 .Background.Fill.TwoColorGradient

 SrcSld.Background.Fill.GradientStyle , \_

 SrcSld.Background.Fill.GradientVariant

 Case Is = msoGradientPresetColors

 .Background.Fill.PresetGradient \_

 SrcSld.Background.Fill.GradientStyle, \_

 SrcSld.Background.Fill.GradientVariant, \_

 SrcSld.Background.Fill.PresetGradientType

 Case Is = msoGradientOneColor

 .Background.Fill.OneColorGradient \_

 SrcSld.Background.Fill.GradientStyle, \_

 SrcSld.Background.Fill.GradientVariant, \_

 SrcSld.Background.Fill.GradientDegree

 End Select

 Case Is = msoFillBackground

 ' Only shapes - we shouldn't come here

 End Select

 End If

 ' >>>>>>>>>>>>>>>>>>>>

 End With

 Next Idx

End Sub

The code doesn't check for read-only or password protected files and will crash on them. Also be careful not to run over the collector file itself.