import arcpy

arcpy.overwriteOutput = True

mxd = arcpy.mapping.MapDocument('CURRENT')

############################################################

##########bscale scenarios

#set text heading variables

landuse\_now="Modelled Land Use 2005"

landuse\_change="Modelled Land Use 2050"

#refresh scenarios text

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s1":

elm.text = str("Markets \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s2":

elm.text = str("Policy \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s3":

elm.text = str("Security \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s4":

elm.text = str("Sustainability \nFirst \nScenario")

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 8.924

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

###main text refresh pt1

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "Future text":

elm.text = str("%s")% (landuse\_change)

Future\_text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","Future text")[0]

Future\_text.elementPositionX, Future\_text.elementPositionY = 8.2 , 13.3

#main text refresh pt2

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "Current text":

elm.text = str("%s")% (landuse\_now)

Current\_text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","Current text")[0]

Current\_text .elementPositionX, Current\_text.elementPositionY = 0.89 ,9.385

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

## #glr lc

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Modelled Land Use")

for layer in layers:

layer.visible = True

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 8.924

s1text.elementPositionX, s1text.elementPositionY = 5.3, 12.35

s2text.elementPositionX, s2text.elementPositionY = 10.29, 12.35

s3text.elementPositionX, s3text.elementPositionY = 5.3, 5.66

s4text.elementPositionX, s4text.elementPositionY = 10.29, 5.66

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2005GLR\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050GLR\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050GLR\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050GLR\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050GLR\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_lu\_glr.png",resolution=600)

## #mek lc

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 2.899

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

#--turn off all layers

mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Modelled Land Use")

for layer in layers:

layer.visible = True

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2005MEK\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050MEK\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050MEK\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050MEK\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050MEK\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_lu\_mek.png" ,resolution=600)

## #wan lc

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 2.899

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

#--turn off all layers

mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Modelled Land Use")

for layer in layers:

layer.visible = True

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2005And\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050And\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050And\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050And\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*99reclassLUT2050And\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd,r"C:\Data\pdf\_map\_outputs\bscale\_lu\_wan.png",resolution=600)

############################################################

##########bscale scenarios bd

#set text heading variables

biodiv\_change="Biodiversity: change in importance 2005 - 2050"

biodiv\_now="Biodiversity importance: \nbaseline 2005"

#refresh scenarios text

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s1":

elm.text = str("Markets \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s2":

elm.text = str("Policy \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s3":

elm.text = str("Security \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s4":

elm.text = str("Sustainability \nFirst \nScenario")

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29,8.924

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

###main text refresh pt1

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "Future text":

elm.text = str("%s")% (biodiv\_change)

Future\_text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","Future text")[0]

Future\_text.elementPositionX, Future\_text.elementPositionY = 7.2 , 13.3

#main text refresh pt2

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "Current text":

elm.text = str("%s")% (biodiv\_now)

Current\_text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","Current text")[0]

Current\_text .elementPositionX, Current\_text.elementPositionY = 0.89 ,9.385

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

## #glr bd

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29, 8.924

s1text.elementPositionX, s1text.elementPositionY = 5.3, 12.35

s2text.elementPositionX, s2text.elementPositionY = 10.29, 12.35

s3text.elementPositionX, s3text.elementPositionY = 5.3, 5.66

s4text.elementPositionX, s4text.elementPositionY = 10.29, 5.66

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

#--turn off all layers

#mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*lakes\_bscale\*")

for layer in layers:

layer.visible = True

dflegend = arcpy.mapping.ListDataFrames(mxd, "Legend\_current")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*lakes\*', dflegend)[0]

lyr.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Watershed boundaries\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_future\_change\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*legend\_current\_importance\*")

for layer in layers:

layer.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_bd\_value\_imp\_bas\_cons\_allsp\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_bd\_value\_change\_mf2050\_cons\_allsp\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_bd\_value\_change\_polf2050\_cons\_allsp\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_bd\_value\_change\_secf2050\_cons\_allsp\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_bd\_value\_change\_susf2050\_cons\_allsp\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_bd\_glr.png",resolution=600)

## #mek bd

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 2.899

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

#--turn off all layers

#mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*lakes\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Watershed boundaries\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_future\_change\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*legend\_current\_importance\*")

for layer in layers:

layer.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_bd\_value\_imp\_bas\_cons\_allsp\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_bd\_value\_change\_mf2050\_cons\_allsp\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_bd\_value\_change\_polf2050\_cons\_allsp\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_bd\_value\_change\_secf2050\_cons\_allsp\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_bd\_value\_change\_susf2050\_cons\_allsp\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_bd\_mek.png",resolution=600)

## #wan bd

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 2.899

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

#--turn off all layers

#mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Andes\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*lakes\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Watershed boundaries\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_future\_change\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*legend\_current\_importance\*")

for layer in layers:

layer.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_bd\_value\_imp\_bas\_cons\_allsp\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_bd\_value\_change\_mf2050\_cons\_allsp\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_bd\_value\_change\_polf2050\_cons\_allsp\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_bd\_value\_change\_secf2050\_cons\_allsp\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_bd\_value\_change\_susf2050\_cons\_allsp\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_bd\_wan.png",resolution=600)

#################################################################

##################################################################

#######################

## #bscale scenarios ef

#set text heading variables

ef\_change="Ecosystem Functions: change in provision 2005 - 2050"

ef\_now=" Ecosystem Functions: \nbaseline provision: 2005"

#refresh scenarios text

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s1":

elm.text = str("Markets \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s2":

elm.text = str("Policy \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s3":

elm.text = str("Security \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s4":

elm.text = str("Sustainability \nFirst \nScenario")

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 2.899

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

###main text refresh pt1

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "Future text":

elm.text = str("%s")% (ef\_change)

Future\_text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","Future text")[0]

Future\_text.elementPositionX, Future\_text.elementPositionY = 6.5 , 13.3

#main text refresh pt2

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "Current text":

elm.text = str("%s")% (ef\_now)

Current\_text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","Current text")[0]

Current\_text .elementPositionX, Current\_text.elementPositionY = 0.89 ,9.385

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

## #glr ef

#--turn off all layers

mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*lakes\_bscale\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Watershed boundaries\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_future\_ef\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\* legend\_ef\_current\*")

for layer in layers:

layer.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_ef\_glr.png",resolution=600)

##########

## ##########bscale scenarios ef

#set text heading variables

ef\_change="Ecosystem Functions: change in provision 2005 - 2050"

ef\_now=" Ecosystem Functions: \nbaseline provision: 2005"

#refresh scenarios text

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s1":

elm.text = str("Markets \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s2":

elm.text = str("Policy \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s3":

elm.text = str("Security \nFirst \nScenario")

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "s4":

elm.text = str("Sustainability \nFirst \nScenario")

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 2.899

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

###main text refresh pt1

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "Future text":

elm.text = str("%s")% (ef\_change)

Future\_text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","Future text")[0]

Future\_text.elementPositionX, Future\_text.elementPositionY = 6.5 , 13.3

#main text refresh pt2

for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT"):

if elm.name == "Current text":

elm.text = str("%s")% (ef\_now)

Current\_text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","Current text")[0]

Current\_text .elementPositionX, Current\_text.elementPositionY = 0.89 ,9.385

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

## #glr ef

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29, 8.924

s1text.elementPositionX, s1text.elementPositionY = 5.3, 12.35

s2text.elementPositionX, s2text.elementPositionY = 10.29, 12.35

s3text.elementPositionX, s3text.elementPositionY = 5.3, 5.66

s4text.elementPositionX, s4text.elementPositionY = 10.29, 5.66

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

#--turn off all layers

#mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*lakes\_bscale\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Watershed boundaries\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_future\_ef\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_current\_provision\*")

for layer in layers:

layer.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_ef\_value\_imp\_bas\_cons\_allsp\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_ef\_value\_change\_mf2050\_cons\_allsp\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_ef\_value\_change\_polf2050\_cons\_allsp\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_ef\_value\_change\_secf2050\_cons\_allsp\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_glr\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_glr\_ef\_value\_change\_susf2050\_cons\_allsp\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_ef\_glr.png",resolution=600)

## #mek ef

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 2.899

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

#--turn off all layers

#mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*lakes\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Watershed boundaries\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_future\_ef\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_current\_provision\*")

for layer in layers:

layer.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_ef\_value\_imp\_bas\_cons\_allsp\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_ef\_value\_change\_mf2050\_cons\_allsp\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_ef\_value\_change\_polf2050\_cons\_allsp\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_ef\_value\_change\_secf2050\_cons\_allsp\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_mek\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_mek\_ef\_value\_change\_susf2050\_cons\_allsp\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_ef\_mek.png",resolution=600)

## #wan ef

#assigning text to variables

s0text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s0")[0]

s1text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s1")[0]

s2text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s2")[0]

s3text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s3")[0]

s4text = arcpy.mapping.ListLayoutElements(mxd, "TEXT\_ELEMENT","s4")[0]

#positioning text for each scenario

s0text.elementPositionX, s0text.elementPositionY = 0.29 , 2.899

s1text.elementPositionX, s1text.elementPositionY = 5.3, 6.9

s2text.elementPositionX, s2text.elementPositionY = 10.29, 6.9

s3text.elementPositionX, s3text.elementPositionY = 5.3, 0.19

s4text.elementPositionX, s4text.elementPositionY = 10.29, 0.19

#--turn off all layers

#mxd = arcpy.mapping.MapDocument("current")

layers = arcpy.mapping.ListLayers(mxd, "\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Andes\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*lakes\*")

for layer in layers:

layer.visible = False

layers = arcpy.mapping.ListLayers(mxd, "\*Watershed boundaries\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_future\_ef\*")

for layer in layers:

layer.visible = True

layers = arcpy.mapping.ListLayers(mxd, "\*Legend\_current\_provision\*")

for layer in layers:

layer.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

df0 = arcpy.mapping.ListDataFrames(mxd, "Baseline")[0]

df1 = arcpy.mapping.ListDataFrames(mxd, "s1")[0]

df2 = arcpy.mapping.ListDataFrames(mxd, "s2")[0]

df3 = arcpy.mapping.ListDataFrames(mxd, "s3")[0]

df4 = arcpy.mapping.ListDataFrames(mxd, "s4")[0]

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df0)[0]

lyr.visible = False

ext = lyr.getExtent()

df0.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_ef\_value\_imp\_bas\_cons\_allsp\*', df0)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df1)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df1.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_ef\_value\_change\_mf2050\_cons\_allsp\*', df1)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df2)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df2.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_ef\_value\_change\_polf2050\_cons\_allsp\*', df2)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df3)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df3.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_ef\_value\_change\_secf2050\_cons\_allsp\*', df3)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

lyr = arcpy.mapping.ListLayers(mxd, '\*buffer\_extent\_bscale\_wan\*', df4)[0]

ext = lyr.getExtent()

lyr.visible = False

ext = lyr.getExtent()

df4.extent = ext

lyr = arcpy.mapping.ListLayers(mxd, '\*bscale\_wan\_ef\_value\_change\_susf2050\_cons\_allsp\*', df4)[0]

lyr.visible = True

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.RefreshTOC()

arcpy.RefreshActiveView()

arcpy.mapping.ExportToPNG(mxd, r"C:\Data\pdf\_map\_outputs\bscale\_ef\_wan.png",resolution=600)