Solutions

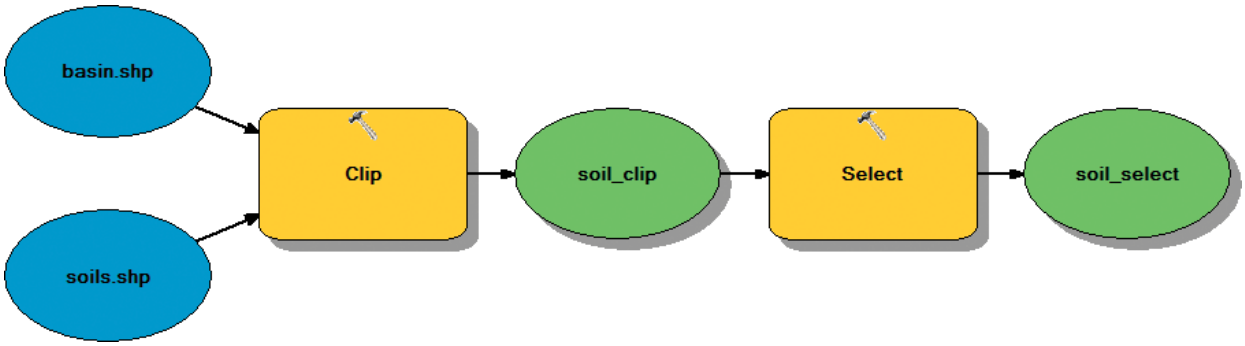
Exercise 1

No challenge exercises.

Exercise 2

Challenge 1

The ModelBuilder result is shown in the figure.



Python script converted from ModelBuilder:

```
# -*- coding: utf-8 -*-
# ---------------------------------------------------------------------------
# soil.py
# Created on: 2012-03-13 11:53:44.00000
# (generated by ArcGIS/ModelBuilder)
# Description:
# ---------------------------------------------------------------------------

# Import arcpy module
import arcpy

# Local variables:
soils_shp = "C:\\EsriPress\\Python\\Data\\Exercise02\\soils.shp"
basin_shp = "C:\\EsriPress\\Python\\Data\\Exercise02\\basin.shp"
soil_clip_shp = "C:\\EsriPress\\Python\\Data\\Exercise02\\Results\\
soil_clip.shp"
soil_select_shp = "C:\\EsriPress\\Python\\Data\\Exercise02\\Results\\
soil_select.shp"

# Process: Clip
arcpy.Clip_analysis(soils_shp, basin_shp, soil_clip_shp, "")

# Process: Select
arcpy.Select_analysis(soil_clip_shp, soil_select_shp, "FARMLNDCL =
'Not prime farmland'")
```

Exercise 3

No challenge exercises.

Exercise 4

Challenge 1

```
mytext = "Geographic Information Systems"
result = mytext.find("Z")
if result == -1:
    print "No"
else:
    print "Yes"
```

Challenge 2

```
mylist = [2, 8, 64, 16, 32, 4]
mylist.sort()
mylist[-2]
```

Challenge 3

```
mylist = [2, 8, 64, 16, 32, 4, 16, 8]
for number in mylist:
    count = mylist.count(number)
    if count <> 1:
        result = "The list provided contains duplicate values."
        break
    else:
        result = "The list provided does not contain duplicate
values."
print result
```

Optional addition to remove duplicates from the list:

```
mylist = [2, 8, 64, 16, 32, 4, 8, 16]
mylist.sort()
for number in mylist:
    count = mylist.count(number)
    if count <> 1:
        mylist.remove(number)
print mylist
```

Challenge 4

a) `5`
b) `'Cairo'`
c) `['Barcelona', 'Cairo', 'Florence', 'Helsinki']`
d) `'Helsinki'`
e) `2`
f) `'Barcelona'`; new list: `['Athens', 'Cairo', 'Florence', 'Helsinki']`
g) No result prints; new list: `['Helsinki', 'Florence', 'Cairo', 'Barcelona', 'Athens']`
h) No result prints; new list: `['Athens', 'Barcelona', 'Cairo', 'Florence', 'Helsinki', 'Berlin']`

Exercise 5

Challenge 1

For Add XY Coordinates tool:

Syntax: `AddXY_management(in_features)`

Required parameter: `in_features` (feature layer, geometry type point)

Optional parameters: none

For Dissolve tool:

Syntax: `Dissolve_management(in_features, out_feature_class, {dissolve_field}, {statistics_fields}, {multi_part}, {unsplit_lines})`

Required parameters:

- `in_features` (feature layer)
- `out_feature_class` (feature class)

Optional parameters:

- `dissolve_field` (field or fields; default: no fields selected)
- `statistics_fields` (field or fields; default: no fields selected)
- `multi_part` (Boolean value; default: multipart features allowed)
- `unsplit_lines` (Boolean value; default: lines dissolved)

Challenge 2

```
import arcpy
from arcpy import env
env.workspace = "C:/Data"
arcpy.AddXY_management("hospitals.shp")
```

Challenge 3

```
import arcpy
from arcpy import env
env.workspace = "C:/Data"
arcpy.Dissolve_management("parks.shp", "parks_dissolved.shp",
"PARK_TYPE", "", "FALSE")
```

Challenge 4

```
import arcpy
default = "no extensions are available"
if arcpy.CheckExtension("3D") == "Available":
    ext_3D = "3D Analyst "
else:
    ext_3D = ""
if arcpy.CheckExtension("Network") == "Available":
    ext_network = "Network Analyst "
else:
    ext_network = ""
if arcpy.CheckExtension("Spatial") == "Available":
    ext_spatial = "Spatial Analyst "
else:
    ext_spatial = ""
print "The following extensions are available: " + ext_3D + ext_
➔ spatial + ext_network + default
```

Exercise 6

Challenge 1

```
import arcpy
from arcpy import env
env.workspace = "C:/Data"
fc_list = arcpy.ListFeatureClasses()
for fc in fc_list:
    desc = arcpy.Describe(fc)
    print "{0} is a {1} feature class".format(desc.basename, ➔
  ➔ desc.shapeType)
```

Challenge 2

```
import arcpy
from arcpy import env
env.workspace = "C:/Data/study.mdb"
fc_list = arcpy.ListFeatureClasses()
arcpy.CreateFileGDB_management("C:/Data", "newstudy.gdb")
for fc in fc_list:
    desc = arcpy.Describe(fc)
    if desc.shapeType == "Polygon":
        arcpy.Copy_management (fc, "C:/Data/newstudy.gdb/" + fc)
```

Exercise 7

Challenge 1

```
import arcpy
from arcpy import env
env.workspace = "C:/EsriPress/Python/Data/Exercise07"
sql1 = " \"FEATURE\" = 'Airport'"
sql2 = " \"FEATURE\" = 'Seaplane Base'"
arcpy.Select_analysis ("airports.shp", "Results/airports_final.shp", sql1)
arcpy.Select_analysis ("airports.shp", "Results/seaports.shp", sql2)
arcpy.Buffer_analysis("Results/airports_final.shp", "Results/aiports_buffers.shp", "15000 METERS")
arcpy.Buffer_analysis("Results/seaports.shp", "Results/seaports_buffers.shp", "7500 METERS")
```

Challenge 2

```
import arcpy
from arcpy import env
env.workspace = "C:/EsriPress/Python/Data/Exercise07"
fc = "roads.shp"
arcpy.AddField_management(fc, "FERRY", "TEXT", "", "", 20)
cursor = arcpy.da.UpdateCursor(fc, ["FEATURE", "FERRY"])
for row in cursor:
    if row[0] == "Ferry Crossing":
        row[1] = "YES"
    else:
        row[1]= "NO"
    cursor.updateRow(row)
```

Exercise 8

Challenge 1

```
import arcpy
from arcpy import env
env.workspace = "C:/Data"
fc = "newpoly2.shp"
arcpy.CreateFeatureclass_management("C:/Data", fc, "Polygon")
cursor = arcpy.da.InsertCursor(fc, ["SHAPE@"])
array = arcpy.Array()
coordlist =[[0, 0], [0, 1000], [1000, 1000], [1000, 0]]
for x, y in coordlist:
    point = arcpy.Point(x,y)
    array.append(point)
polygon = arcpy.Polygon(array)
cursor.insertRow([polygon])
del cursor
```

Challenge 2

```
import arcpy
from arcpy import env
env.workspace = "C:/EsriPress/Python/Data/Exercise08"
fc = "Hawaii.shp"
newfc = "Results/Hawaii_single.shp"
arcpy.MultipartToSinglepart_management(fc, newfc)
spatialref = arcpy.Describe(newfc).spatialReference
unit = spatialref.linearUnitName
cursor = arcpy.da.SearchCursor(newfc, ["SHAPE@"])
for row in cursor:
    print ("{0} square {1}".format(row[0].area, unit))
```

Challenge 3

```
import arcpy
from arcpy import env
env.workspace = "C:/EsriPress/Python/Data/Exercise08"
fc = "Hawaii.shp"
newfc = "envelope8.shp"
desc = arcpy.Describe(fc)
spatialref = desc.spatialReference
extent = desc.extent
arcpy.CreateFeatureclass_management("C:/EsriPress/Python/Data/
Exercise08", newfc, "Polygon", "", "", "", spatialref)
cursor = arcpy.da.InsertCursor(newfc, ["SHAPE@"])
array = arcpy.Array()
array.append(extent.upperLeft)
array.append(extent.upperRight)
array.append(extent.lowerRight)
array.append(extent.lowerLeft)
polygon = arcpy.Polygon(array)
cursor.insertRow([polygon])
del cursor
```

Exercise 9

Challenge 1

```
import arcpy
from arcpy import env
from arcpy.sa import *
env.workspace = "C:/EsriPress/Python/Data/Exercise09"
if arcpy.CheckExtension("Spatial") == "Available":
    arcpy.CheckOutExtension("Spatial")
elev = arcpy.Raster("elevation")
lc = arcpy.Raster("landcover.tif")
slope = Slope(elev)
aspect = Aspect(elev)
goodslope = ((slope > 5) & (slope < 20))
goodaspect = ((aspect > 150) & (aspect < 270))
goodland = ((lc == 41) | (lc == 42) | (lc ==43))
outraster = (goodslope & goodaspect & goodland)
outraster.save("C:/EsriPress/Python/Data/Exercise09/Results/final")
arcpy.CheckInExtension("Spatial")
```

Challenge 2

```
import arcpy
from arcpy import env
out_path = "C:/EsriPress/Python/Data/Exercise09"
env.workspace = out_path
rasterlist = arcpy.ListRasters()
arcpy.CreatePersonalGDB_management(out_path + "/Results", "myrasters.
gdb")
for raster in rasterlist:
    desc = arcpy.Describe(raster)
    rname = desc.baseName
    outraster = out_path + "/Results/myrasters.gdb/" + rname
    arcpy.CopyRaster_management(raster, outraster)
```

Exercise 10

Challenge 1

```
import arcpy
from arcpy import env
env.workspace = "C:/EsriPress/Python/Data/Exercise10"
mxd = arcpy.mapping.MapDocument("C:/EsriPress/Python/Data/Exercise10/
Austin_TX.mxd")
df = arcpy.mapping.ListDataFrames(mxd, "Parks")[0]
lyr = arcpy.mapping.ListLayers(mxd, "parks", df)[0]
dflist = arcpy.mapping.ListDataFrames(mxd)
for dframe in dflist:
    if dframe.name <> "Parks":
        arcpy.mapping.AddLayer(dframe, lyr)
mxd.save()
del mxd
```

Exercise 11

Challenge 1

The four coding errors are highlighted in yellow as follows:

```
import arcpy
from arcpy import env
env.workspace = "C:/EsriPress/Python/Data/Exercise07"
FC = "airports.shp"
rows = arcpy.SearchCursor(fc)
fields = arcpy.ListFields(fc)
for field in fields:
    if fields.name == "NAME":
        for row in rows:
        ____print "Name = {0}".format(row.getValue(field.name))
```

Challenge 2

The six coding errors are highlighted in yellow as follows:

```
import arcpy
from arcpy import env
env.workspace = "C:/EsriPress/Python/Data\Exercise09"
raster = "landcover.tiff"
desc = arcpy.describe(raster)
x = desc.MeanCellHeight
y = desc.MeanCellWidth
spatialref = desc.spatialReference
units = spatialref.linearUnitName
print "Cells are" + str(x) + " by " + str(y) + " " + units + "."
```

Exercise 12

Challenge 1

For callingscript.py:

```
import arcpy
import mycount
table = "C:/EsriPress/Python/Data/Exercise12/streets.shp"
print mycount.countstringfields(table)
```

For mycount.py:

```
import arcpy
import os
def countstringfields(table):
    fields = arcpy.ListFields(table)
    i = 0
    for field in fields:
        if field.type == "String":
            i += 1
    return i
```

Challenge 2

For parcelclass.py:

```
import arcpy
import parcelclass
from arcpy import env
env.workspace = "C:/EsriPress/Python/Data/Exercise12"
fc = "parcels.shp"
cursor = arcpy.da.SearchCursor(fc, ["FID", "Landuse", "Value"])
for row in cursor:
    myparcel = parcelclass.Parcel(row[1], row[2])
    mytax = myparcel.assessment()
    print "{0}: {1}".format(row[0], mytax)
```

Exercise 13

Challenge 1

```
import arcpy
import random
from arcpy import env
env.overwriteOutput = True
inputfc = arcpy.GetParameterAsText(0)
outputfc = arcpy.GetParameterAsText(1)
percent = int(arcpy.GetParameterAsText(2))
desc = arcpy.Describe(inputfc)
input_count = int(arcpy.GetCount_management(inputfc)[0])
outcount = int(round(input_count * percent * 0.01))
inlist = []
randomlist = []
fldname = desc.OIDFieldName
rows = arcpy.SearchCursor(inputfc)
row = rows.next()
while row:
    id = row.getValue(fldname)
    inlist.append(id)
    row = rows.next()
while len(randomlist) < outcount:
    selitem = random.choice(inlist)
    randomlist.append(selitem)
    inlist.remove(selitem)
length = len(str(randomlist))
sqlexp = '"' + fldname + '"' + " in " + "(" + str(randomlist)
[1:length - 1] + ")"
arcpy.MakeFeatureLayer_management(inputfc, "selection", sqlexp)
arcpy.CopyFeatures_management("selection", outputfc)
```

In the script tool, the parameter "Number of Features" is replaced by "Percent of Features" of type Integer with a range filter from 0 to 100.

Exercise 14

Challenge 1

Results will vary with the tool selected.