Getting Started with Python in ArcGIS Pro

```python
import arcpy
from arcpy import env
env.overwriteOutput = True
arcpy.env.workspace = "C:/Bits_bytes/PRJ/
outclip = arcpy.CreateFolder_management("C:/Bits_bytes/","CLIPPED")
fclist = arcpy.ListFeatureClasses()
for fc in fcclist:
in_feat = fc
  if in_feat != "Wards.shp" and in_feat != "Clip.shp":
    print ("C:/Bits_bytes/CLIPPED/" + fc, " will be clipped \n")
    out_feat = "C:/Bits_bytes/CLIPPED/" + fc
    print(out_feat)
    clipper = "C:/Bits_bytes/PRJ/Clip.shp"
    arcpy.Clip_analysis("C:/Bits_bytes/PRJ/" + fc,clipper,out_feat)
```
What is ArcPy?

ArcPy is a Python module that interacts with the tools in arctoolbox which are part of ArcGIS Pro and ArcGIS Desktop. This module allows the user to access the geoprocessing tools available in ArcGIS Pro and Desktop.
Python in ArcGIS Pro

- **Python Notebook**: Open a new notebook.
- **Python Window**: Show python window.
The Python window in ArcGIS Pro
Printing text

print('Welcome to python')

Using a variable
Print text with a variable
text = ('Welcome to python')
print(text)
Open the attribute table to see how many records are in the roads
We get the same information by using the Get Count tool from arctoolbox into the python window:

```python
arcpy.management.GetCount('centreline')
```

<Result '67769'>
Assign and Print a Variable

count = arcpy.management.GetCount('centreline')
print(count)
ArcGIS Toolbox Help

Get Count (Data Management Tools)
Returns the total number of rows for a table.

Input Rows (Required)
The input table view or raster layer. If a selection is defined on the input, the count of the selected rows is returned.
When using tools help is available at the command line
Get Count (Data Management)

Summary

Returns the total number of rows for a table.

Usage

- If the input is a layer or table view containing a selected set of records, only the selected records will be counted.

- This tool honors the Extent environment. Only those features that are within or intersect the Extent environment setting will be counted.

- You can view the returned row count in Geoprocessing history.

- In ModelBuilder, Get Count can be used to set up a precondition, as illustrated below. In this model, Get Count counts the number of records returned by the Select tool. If the count is zero, Buffer will not run due to the precondition.
GetCount example 1 (Python window)
The following Python Window script demonstrates how to use the GetCount function in immediate mode.

```python
import arcpy
arcpy.env.workspace = "C:/data/data.gdb"
arcpy.GetCount_management("roads")
```

GetCount example 2 (stand-alone script)
The following stand-alone script is an example of how to use the GetCount function in a scripting environment.

```python
# Name: fcCount.py
# Purpose: calculate the number of features in a feature class

# Import system modules
import arcpy

lyrfile = r"C:\data\streets.lyr"
result = arcpy.GetCount_management(lyrfile)
print('{} has {} records'.format(lyrfile, result[0]))
```
List feature classes in a Directory

```python
arcpy.env.workspace = "C:/Bits_Bytes/"
fclist = arcpy.ListFeatureClasses()
print(fclist)
```

Python window code and output

```
['centreline.shp', 'green_space.shp', 'Schools.shp', 'TTC_Subway.shp', 'Wards.shp']
```
Python directory for stand alone programs

Python Package Manager

Project Environment

installed Package [C:\Program Files\ArcGIS\Pro\bin\Python\envs\arcgispro-py3]

Manage Environments

Installed Packages

The following list of Python packages are installed:

Note: Cannot modify the default Python environment.
Setup Python version in PyScripter

C:\Program Files\ArcGIS\Pro\bin\Python\envs\arcgispro-py3
import arcpy
from arcpy import env
dev.workspace = "C:/Bits_Bytes/"
prjfile = "C:/Bits_Bytes/Python/Schools.prj"
spatial_ref = arcpy.SpatialReference(prjfile)
print ("the projection is ", spatial_ref.name, " and will be used to check the projection of the shapefiles in this directory 
")
fclass = arcpy.ListFeatureClasses()
print ("These shapefiles do not have a UTM projection")
for fc in fclass:
    spatial_ref = arcpy.Describe(fc).spatialReference
    if spatial_ref.name != "NAD_1983_UTM_Zone_17N":
        print (fc)
AMD64)] on win32. ***

>>> *** Remote Interpreter Reinitialized ***
the projection is  NAD_1983_UTM_Zone_17N
and will be used to check the projection of
the shapefiles in this directory

| These shapefiles do not have a UTM projection
centreline.shp
green_space.shp
TTC_Subway.shp
Wards.shp

>>>
Project Shapefiles

Project_management (in_dataset, out_dataset, out_coord_system, {transform_method}, {in_coord_system})

• in_dataset in_feat this is the input shapefile which will be projected
• out_dataset out_feat this is the output shapefile which will be the projected shapefile
• out_coord_system out_coord is the projection that will be assigned to the new shapefile. This is defined by using the SpatialReference function where the new projection will be Nad 1983 UTM Zone 17
There are many ways to define projections in ArcGIS. The spatial reference can be defined by using the prj file from a shapefile, from typing out the actual projection such as 'NAD 1983 UTM Zone 17N' or by specifying an EPSG number such as 26917.
Project Shapefile

In ArcGIS Pro type the code below in the python window. This will project the shapefile by assigning the SpatialReference command a projection definition and then using project_management to project the shapefile.

```python
import arcpy
from arcpy import env
env.overwriteOutput = True
in_feat = "C:/Bits_Bytes/Wards.shp"
out_feat = "C:/Bits_Bytes/PRJ/Wards.shp"
out_coord = arcpy.SpatialReference('Nad 1983 UTM Zone 17N')
arcpy.Project_management(in_feat, out_feat, out_coord)
```
import arcpy
from arcpy import env
ever.writeOutput = True
env.workspace = "C:/Bits_Bytes/"
prjfile = "C:/Bits_Bytes/Schools.prj"
spatial_ref = arcpy.SpatialReference(prjfile)
print ("the projection for schools.shp is ", spatial_ref.name)
fclass = arcpy.ListFeatureClasses()
for fc in fclass:
    spatial_ref = arcpy.Describe(fc).spatialReference
    print (spatial_ref.name, " is the projection for ", fc)
    if spatial_ref.name == "NAD_1983_UTM_Zone_17N":
        print (fc, "... will not be projected 
"")
    else:
        print (fc, "...is being projected")
out_dir = "C:/Bits_Bytes/PRJ/"
out_coord = arcpy.SpatialReference('NAD 1983 UTM Zone 17N')
in_feat = fc
out_feat = out_dir + fc
arcpy.Project_management(in_feat, out_feat, out_coord)
Select data for clipping

Using the select tool we can see this sql code

This will not work in python!
Use Model Builder to get python Code
```python
Wards_Select1 = "C:\Users\User\AppData\Local\Temp\ArcGISProTemp14964\ef00c718-76f5-45b3-989d-2efeda70a73a\Default.gdb\Wards_Select1"

with arcpy.EnvManager(scratchWorkspace="C:\Users\User\AppData\Local\Temp\ArcGISProTemp14964\ef00c718-76f5-45b3-989d-2efeda70a73a\Default.gdb", workspace="C:\Users\User\AppData\Local\Temp\ArcGISProTemp14964\ef00c718-76f5-45b3-989d-2efeda70a73a\Default.gdb"):
    arcpy.analysis.Select(in_features=Wards, out_feature_class=Wards_Select1, where_clause="NAME = 'St. Paul''s (21')")

if __name__ == '__main__':
    Model()
```
From select tool

NAME = 'St. Paul''s (21)'

From Model builder

"NAME = 'St. Paul''s (21)''"
SQL Syntax from Model Builder

arcpy.analysis.Select(in_features=Wards, out_feature_class=clip_shp, where_clause="NAME = 'St. Paul's (21)""")
Select_analysis extracts features from an input feature class or input feature layer, typically using a select or Structured Query Language (SQL) expression and stores them in an output feature class

```
Select_analysis (in_features, out_feature_class, {where_clause})
Inf_features     Wards.shp
Out_feature_class Clip.shp
Where clause     "NAME = 'St. Paul's (21)"
```

This will be done in ArcGIS Pro

```
import arcpy
from arcpy import env
env.overwriteOutput = True
env.workspace = "C:/Bits_bytes/PRJ"
arcpy.Select_analysis('Wards','C:/Bits_bytes/PRJ/Clip.shp','"NAME = 'St. Paul's(21)'")
```
Clip data

To be run in PyScripter

```python
import arcpy
from arcpy import env
env.overwriteOutput = True
arcpy.env.workspace = "C:/Bits_bytes/PRJ/"
outclip = arcpy.CreateFolder_management("C:/Bits_bytes/","CLIPPED")
fclist = arcpy.ListFeatureClasses()
for fc in fclist:
in_feat = fc
if in_feat != "Wards.shp" and in_feat != "Clip.shp":
    print ("C:/Bits_bytes/CLIPPED/" + fc, " will be clipped \n")
out_feat = "C:/Bits_bytes/CLIPPED/" + fc
print(out_feat)
clipper = "C:/Bits_bytes/PRJ/Clip.shp"
arcpy.Clip_analysis("C:/Bits_bytes/PRJ/" + fc,clipper,out_feat)
```