Geographic Information Systems in Water Resources
Python Scripting and Programming in ArcGIS to evaluate Height Above the Nearest Drainage and Flood Inundation

CUAHSI Virtual University, 2019
David Tarboton

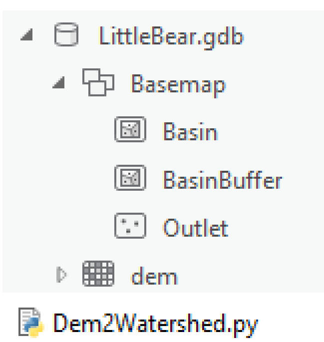
Purpose

The purpose of this exercise is to learn how to use Python Scripting and Programming in ArcGIS. The application is to the calculation of height above the nearest drainage (HAND) from a digital elevation model, that has application for calculating stream reach hydraulic properties, flood inundation depth and flood inundation mapping.

Data

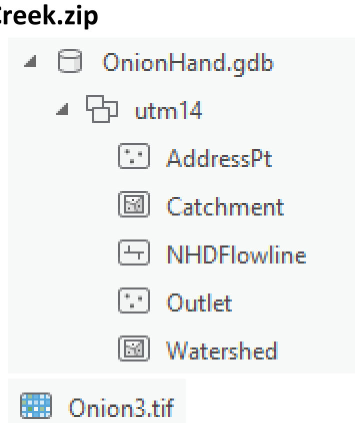
The following data is provided

LittleBear.zip



- **dem**. A digital elevation model covering the Little Bear River
- **Basemap**. A feature dataset with basin boundary buffered basin boundary and outlet

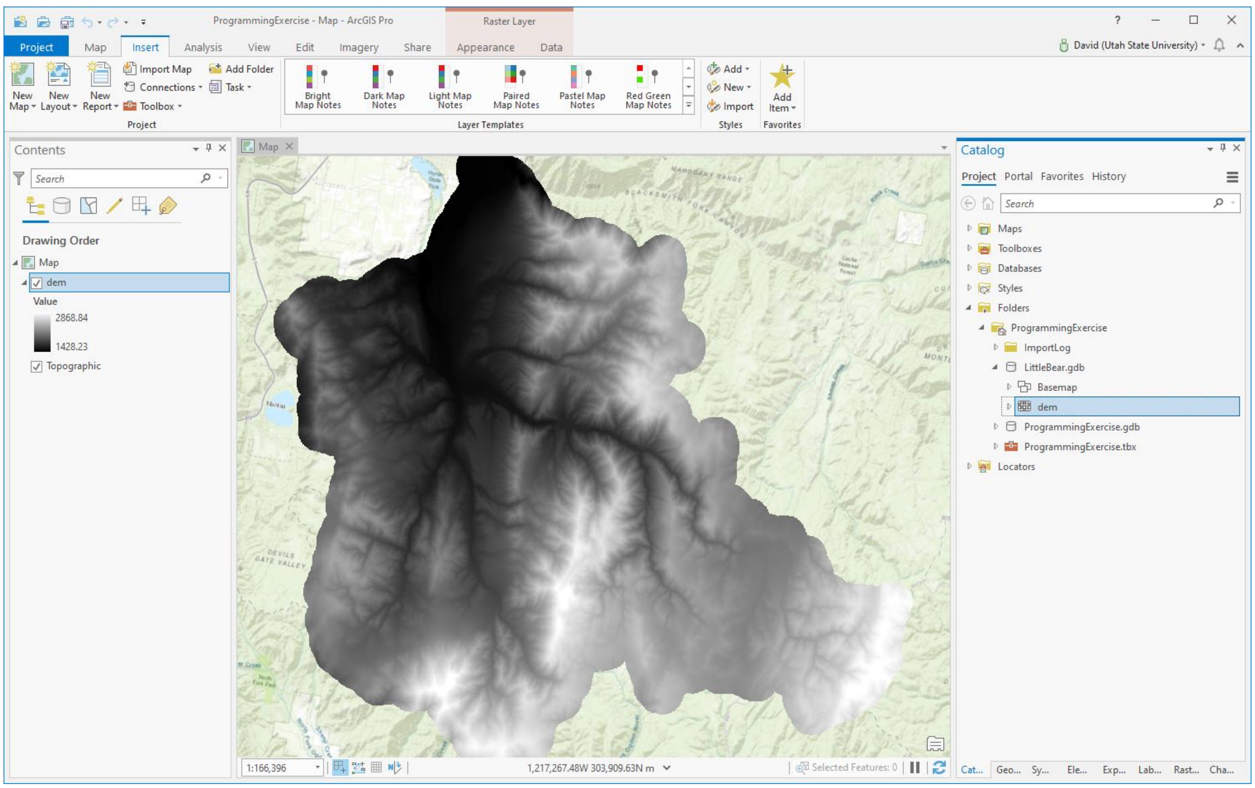
OnionCreek.zip



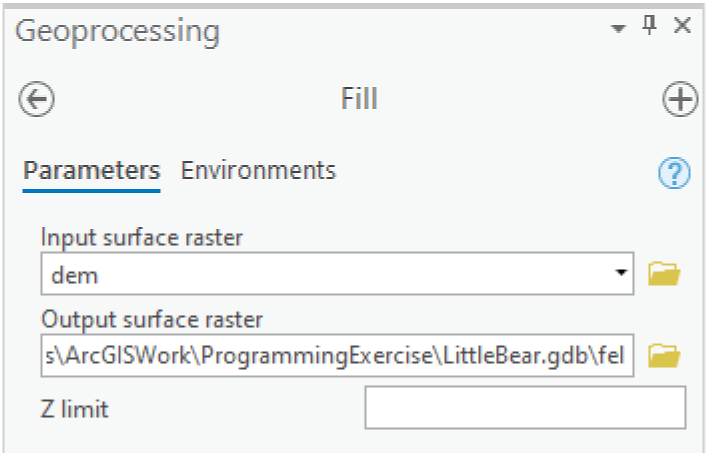
- **OnionHand.gdb**. A geodatabase containing NHDPlus flowlines and catchments for Onion Creek. This also contains address points that will be used in assessing addresses vulnerable to flooding. These have been projected to a UTM zone 14 coordinate system for consistency. Outlet is the watershed outlet
- **Onion3.tif**. A 10 m digital elevation model from the National Elevation dataset. This DEM was obtained from the National Map, extracted to a 2 km buffer around the watershed and projected to the same UTM zone 14 coordinate system as the vector data, for consistency. It has also had a flow direction conditioning procedure applied to it using TauDEM to remove barriers along high resolution NHD flowlines.

Developing the steps of a program

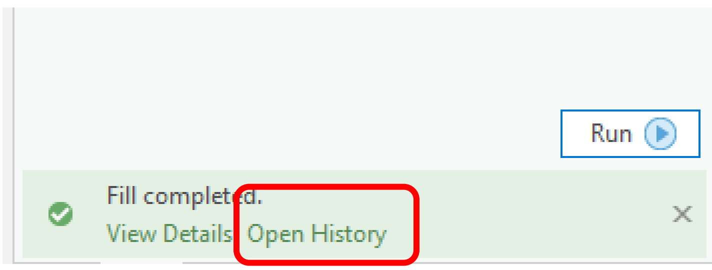
1. Open ArcGIS Pro and create a new map project. I used the folder C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise to hold this project.
2. Copy the LittleBear.gdb geodatabase from LittleBear.zip into the project folder
3. Add **dem** raster to the map.



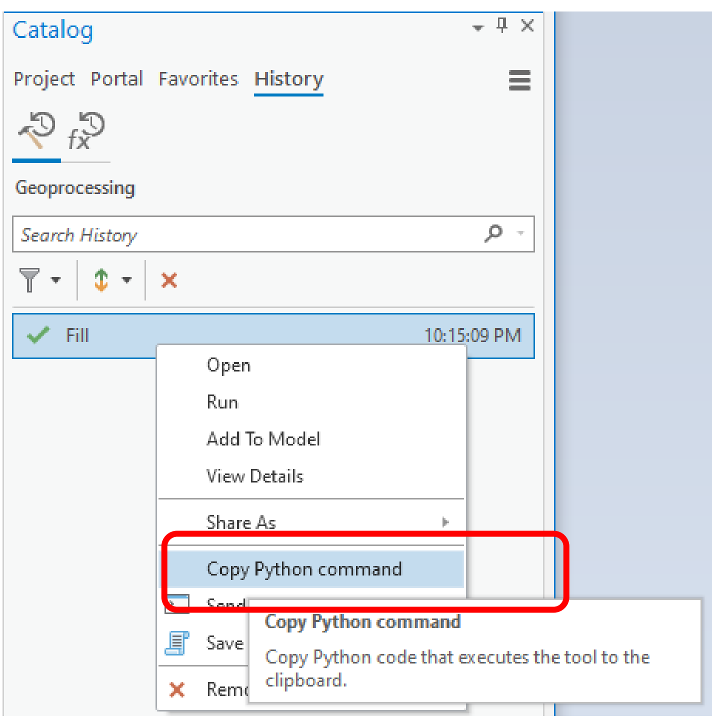
4. Use the Fill Geoprocessing tool to Fill pits



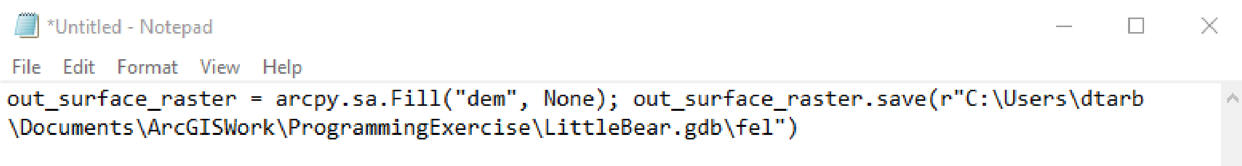
5. Open History at the bottom of the Geoprocessing pane



6. Copy Python Command

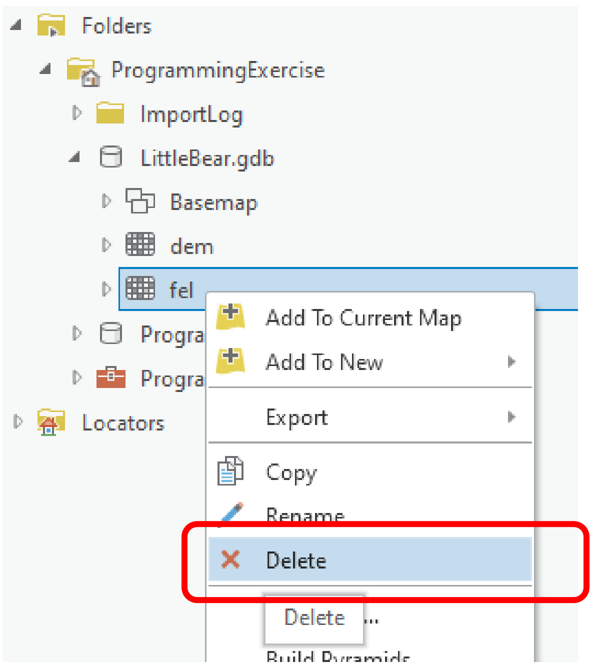


7. Open Notepad (or any text editor and paste)

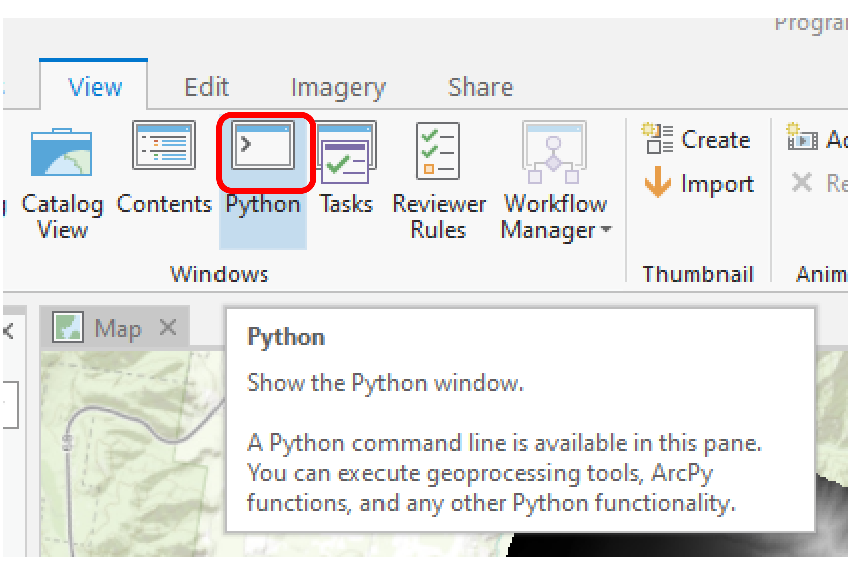


You should see the Python Commands for the fill function.

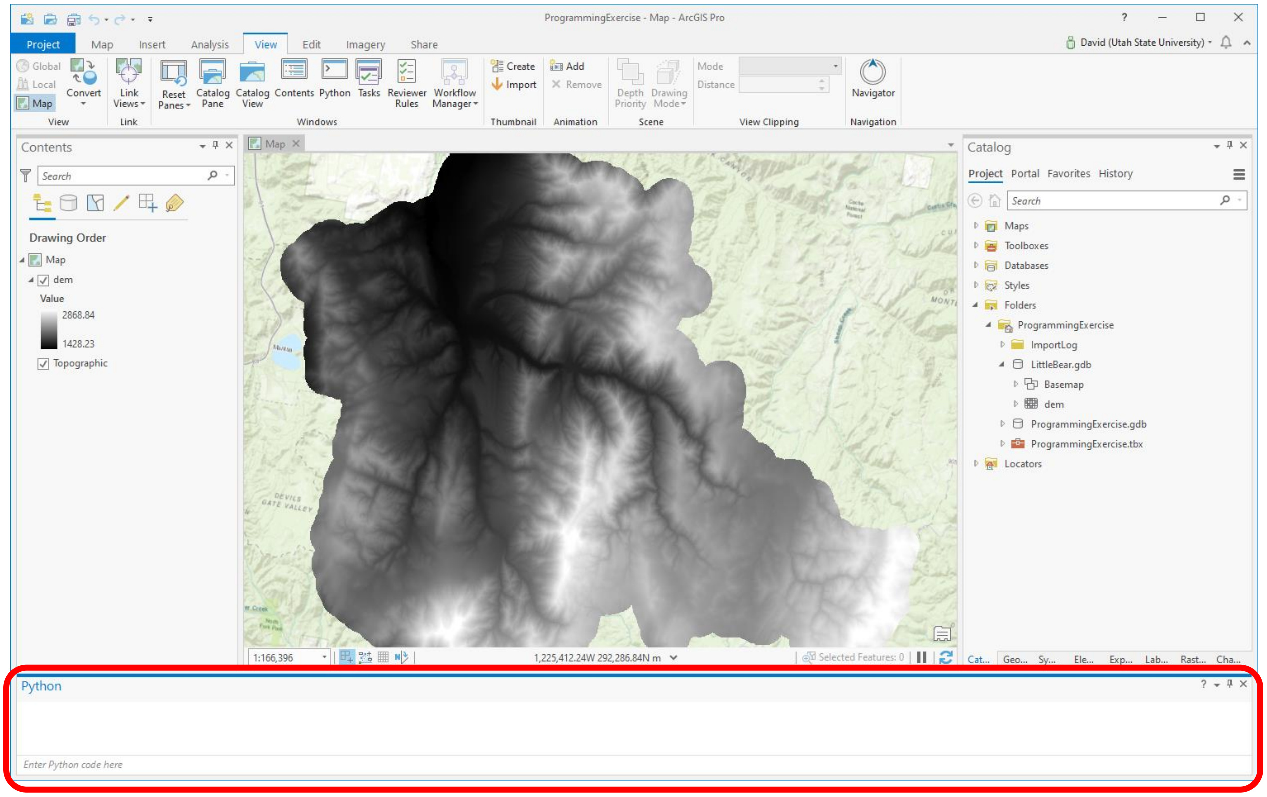
8. Delete **fel** from your Project



9. From the View tab show the python Window



You should see a Python window at the bottom of your map. This is where you can enter Python commands.

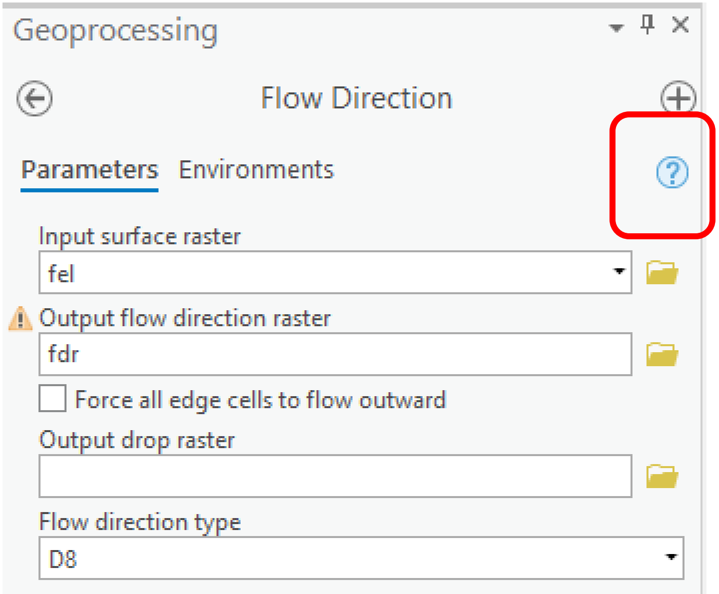


10. Type (or copy and paste) the line
    out_surface_raster = arcpy.sa.Fill("dem", None)
    into the Python window and press **Enter**
    The Fill function should execute. This is how ArcGIS functions can be executed from Python, a script, or command alternative to the Geoprocessing tool interface

11. Type (or copy and paste) the line
    out_surface_raster.save(r"C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise\LittleBear.gdb\fel")
    into the Python window and press Enter. This saves the result from its initial name to a raster with name **fel**.

12. Now execute the following additional steps involved in Watershed Delineation
    Flow Direction
    Flow Accumulation

    You can learn their Python commands from the history, as you did for Fill, or from Geoprocessing Help Code Sample

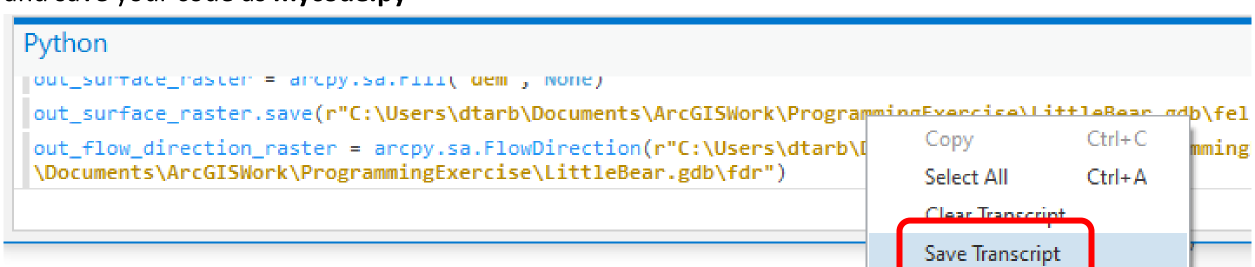


FlowDirection example 1 (Python window)

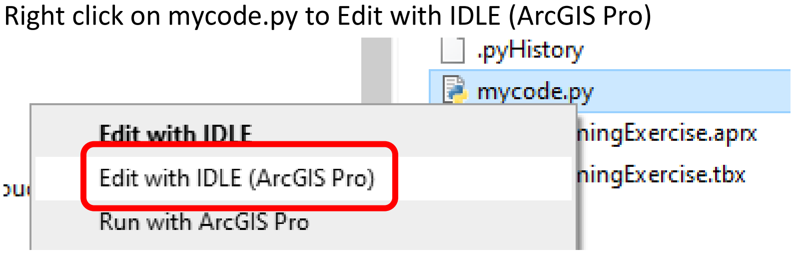
This example creates a D8 flow direction raster from an input Grid elevation surface raster.

```
import arcpy
from arcpy import env
from arcpy.sa import *
env.workspace = "C:/sapyexamples/data"
outFlowDirection = FlowDirection("elevation", "NORMAL")
outFlowDirection.save("C:/sapyexamples/output/outflowdir01")
```

13. After executing a few Python Commands in the Python Window right click to Save Transcript and save your code as **mycode.py**



14. Right click on mycode.py to Edit with IDLE (ArcGIS Pro)



IDLE is Python's simple program editor. You should get a window similar to the below that has the commands you used.

mycode.py - C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise\mycode.py (3.6.8)

File Edit Format Run Options Window Help

```
# coding: utf-8
out_surface_raster = arcpy.sa.Fill("dem", None)
out_surface_raster.save(r"C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise\LittleBear.gdb\fel")
out_flow_direction_raster = arcpy.sa.FlowDirection(r"C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise\LittleBear.gdb\fe
```

Ln: 1 Col: 0

Note that if you are doing extensive Python programming I would recommend a more sophisticated editor such as Pycharm, but for this exercise it is OK to persist with IDLE.

15. Edit the text in the window to the following (except using your file locations)

mycode.py - C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise\mycode.py (3.6.8)

File Edit Format Run Options Window Help

```
# Watershed delineation program
# David Tarboton
# 9/23/19

import arcpy
from arcpy import env
from arcpy.sa import *

# Set inputs
outDir=r"C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise\LittleBear.gdb"
DEM="dem"

# Set workspace environment
env.workspace = outDir
env.scratchWorkspace = outDir
env.overwriteOutput = True

outFill = Fill(DEM, None)
outFill.save("fel"); print("DEM Filled")

outFlowDirection = FlowDirection("fel", "NORMAL")
outFlowDirection.save("fdr"); print("Flow directions computed")

outFlowAccumulation = FlowAccumulation("fdr")
outFlowAccumulation.save("fac"); print("Flow accumulation computed")
```

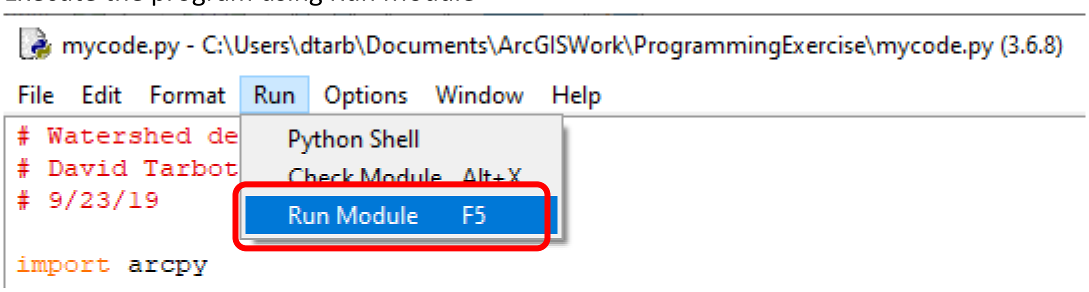
Ln: 26 Col: 0

```
# Watershed delineation program
# David Tarboton
# 9/23/19
```

```
import arcpy
from arcpy import env
from arcpy.sa import *

# Set inputs
outDir=r"C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise\LittleBear.gdb"
DEM="dem"

# Set workspace environment
env.workspace = outDir
env.scratchWorkspace = outDir
env.overwriteOutput = True

outFill = Fill(DEM, None)
outFill.save("fel"); print("DEM Filled")

outFlowDirection = FlowDirection("fel", "NORMAL")
outFlowDirection.save("fdr"); print("Flow directions computed")

outFlowAccumulation = FlowAccumulation("fdr")
outFlowAccumulation.save("fac"); print("Flow accumulation computed")
```

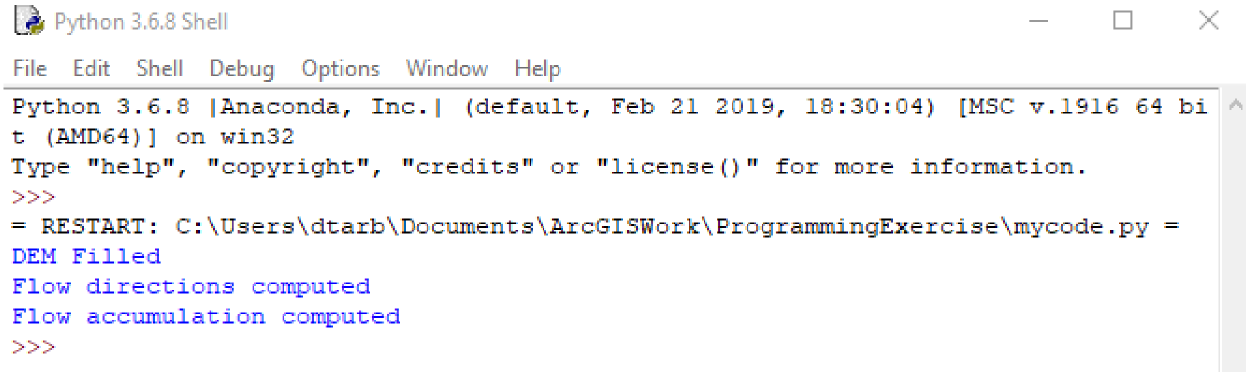
Note that the code above was 'learned' by trial and error and reading the help files.

16. Execute the program using Run Module

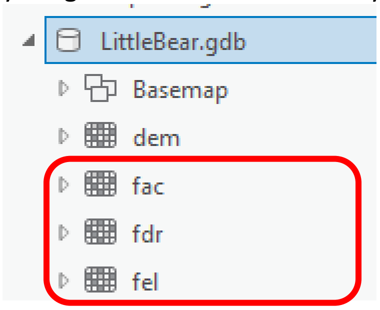


Note that it is not uncommon to get errors. This is just part of programming. With ArcGIS many errors are related to attempting to overwrite existing inputs, or incorrect paths to the data being used.

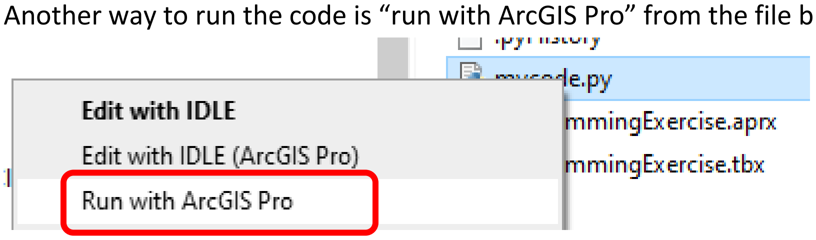
If all goes well you should see



And you should have the filled elevation, flow direction and flow accumulation results added to your geodatabase. Close the Python Shell window.



17. Another way to run the code is "run with ArcGIS Pro" from the file browser



To turn in: A screen shot from IDLE with your mycode.py program
A screen shot from Python that shows the completion of mycode.py execution
Screen shots from ArcGIS showing flow direction and flow accumulation (symbolize these pleasantly) for the Little Bear DEM provided.

Using with different data

Locate the logan.tif file from the Watershed delineation exercise and copy this to your ProgrammingExercise folder. Modify mycode.py to work with logan.tif and fill the pits, calculate flow direction and flow accumulation for this dem.

```
# Set inputs
outDir=r"C:\Users\dtarb\Documents\ArcGISWork\ProgrammingExercise"
DEM="logan.tif"
```

To turn in: A screen shot from IDLE with your mycode.py program modified to work with logan.tif
A screen shot from Python that shows the completion of mycode.py execution

Screen shots from ArcGIS showing flow direction and flow accumulation (symbolize these pleasantly) for the logan.tif DEM.

Watershed delineation program

A complete set of steps to achieve watershed delineation is

1. Set Inputs
   - DEM
   - Outlet
   - Threshold
2. Set workspace
3. Fill
4. Flow Direction
5. Flow Accumulation
6. Snap Outlet
7. Watershed
8. Stream Raster
9. Stream Link
10. Catchment
11. Vector Conversion

The following program achieves this (this code is in Dem2Watershed.py in LittleBear.zip)

Dem2Watershed.py - C:\Users\dtarb\Dropbox\Academics\CUAHSIVirtualUniversity\MyWatershed\Dem2Watershed.py (3.6.8)

File Edit Format Run Options Window Help

```python
outDir=r"C:\Users\dtarb\Desktop\MyWatershed\MyWatershed.gdb"
DEM="dem"
gage = "Basemap/Outlet"
threshold = 10000
snapdistance = 100

# Set workspace environment
env.workspace = outDir
env.scratchWorkspace = outDir
env.overwriteOutput = True

outFill = Fill(DEM, None)
outFill.save("fel"); print("DEM Filled")

outFlowDirection = FlowDirection("fel", "NORMAL")
outFlowDirection.save("fdr"); print("Flow directions computed")

outFlowAccumulation = FlowAccumulation("fdr")
outFlowAccumulation.save("fac"); print("Flow accumulation computed")

outSnapPour = SnapPourPoint(gage, "fac", snapdistance)
outSnapPour.save("SnapOutlet"); print("Pour point snapped")

outWatershed = Watershed("fdr", "SnapOutlet")
outWatershed.save("demw"); print("Watershed delineated")

StreamRaster = (Raster("fac") >= threshold) & (Raster("demw") >= 0)
StreamRaster.save("str"); print("Stream raster generated")

outStreamLink = StreamLink("str","fdr")
outStreamLink.save("strlnk"); print("Stream links created")

Catchment = Watershed("fdr", "strlnk")
Catchment.save("CatchmentGrid"); print("Catchment grid created")

StreamToFeature("strlnk", "fdr", "DrainageLine","NO_SIMPLIFY")
arcpy.RasterToPolygon_conversion("CatchmentGrid", "Catchments", "NO_SIMPLIFY","","MULTIPLE_OUTER_PART")
print("Vector conversions done")
```

Ln: 37 Col: 56

Change the inputs(**change outDir line to location you are using**) and execute this program for the Little Bear DEM (dem in LittleBear.zip).

To turn in: A layout showing Catchments (polygons) and DrainageLine (polyline) features nicely symbolized for the Little Bear River.

Applying to Onion Creek.

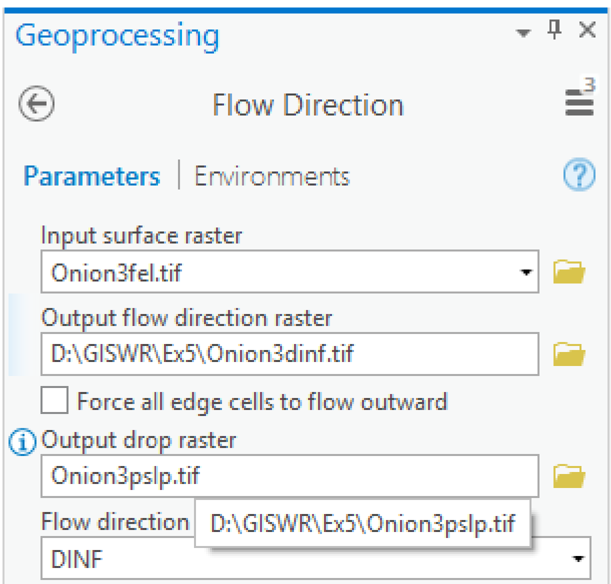
Onion Creek is near Austin Texas and has been the location of significant flooding. The Onion Creek DEM and Outlet are provided in **OnionCreek.zip**.

1. Use Dem2Watershed.py to delineate the Catchments and streams of Onion Creek using a stream delineation threshold of 50000 grid cells.

To turn in: A layout showing Catchments (polygons) and DrainageLine (polyline) features nicely symbolized for Onion Creek.

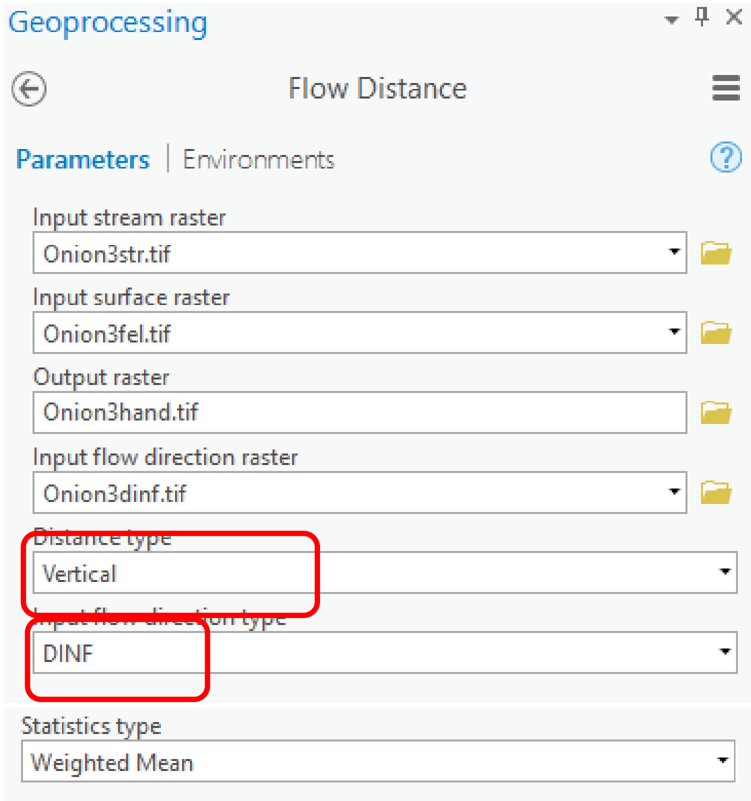
Extend program to calculate Height above Nearest Drainage (HAND)

1. Use Flow direction with the DINF option to calculate Dinfinity flow directions



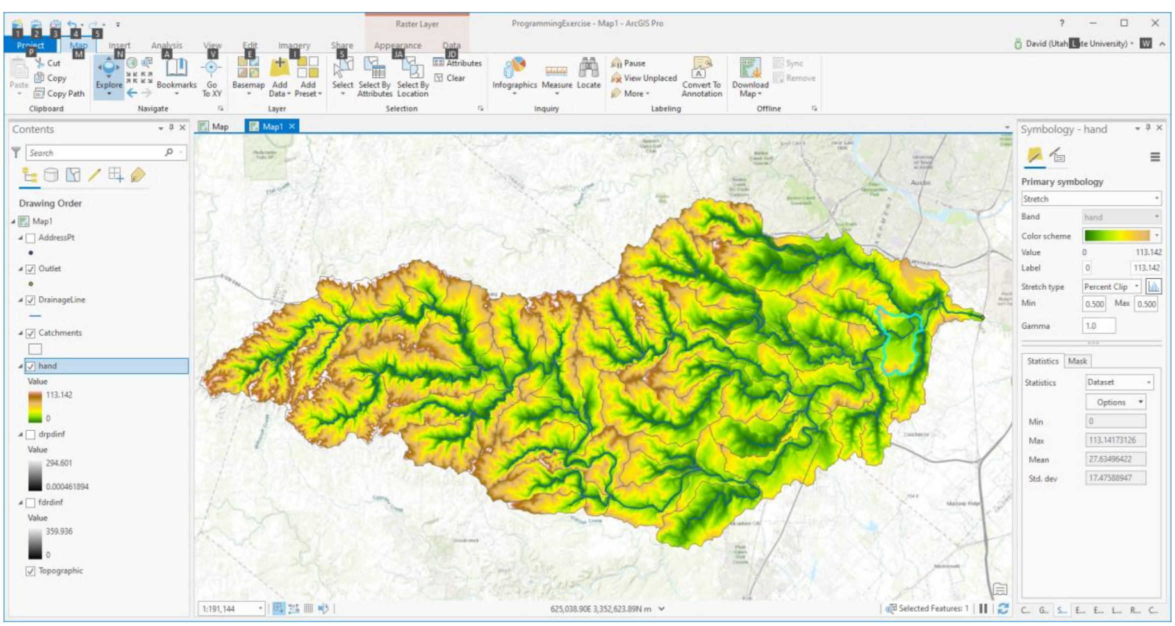
Note that Dinfinity is used for HAND because distances to streams are across hillslopes. Dinfinity is generally better for representing smooth hillslope flow fields, whereas D8 is used in valleys where the topography is convergent and we do not want dispersion of flow.

2. Use Flow distance with the DINF flow direction and vertical distance to calculate Height Above Nearest Drainage for Onion Creek



3. Extend Dem2Watershed.py to also include these functions and execute this for Onion Creek.

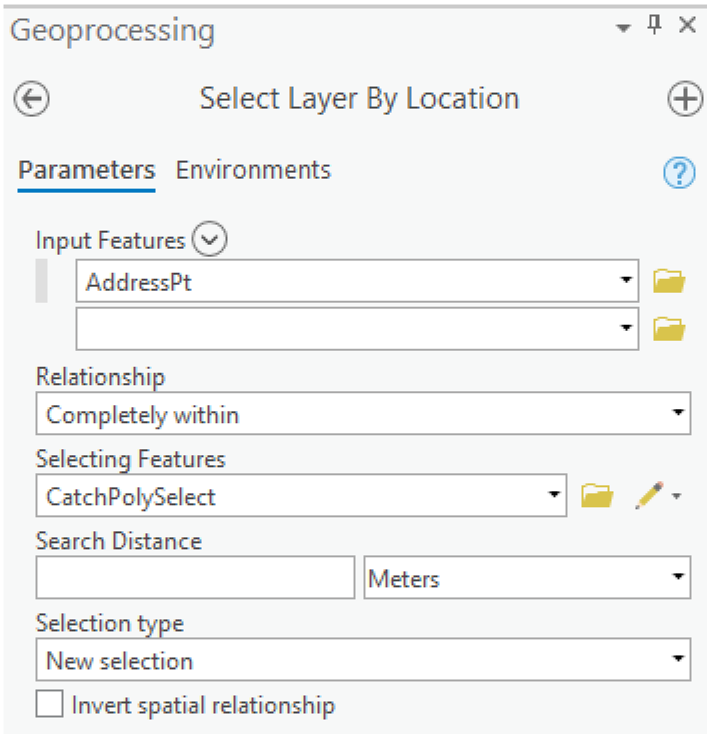
The result should be a map of height above nearest drainage, or relative elevation for Onion Creek, similar to the below



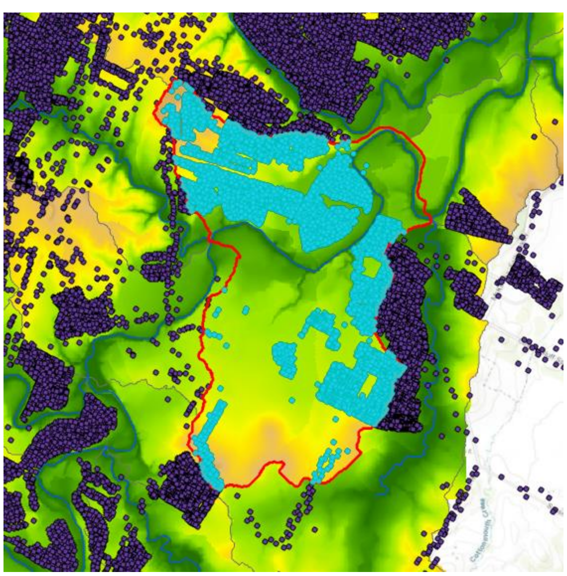
To turn in: Your Dem2Watershed.py script modified to also evaluate HAND.

A layout of HAND for Onion Creek.

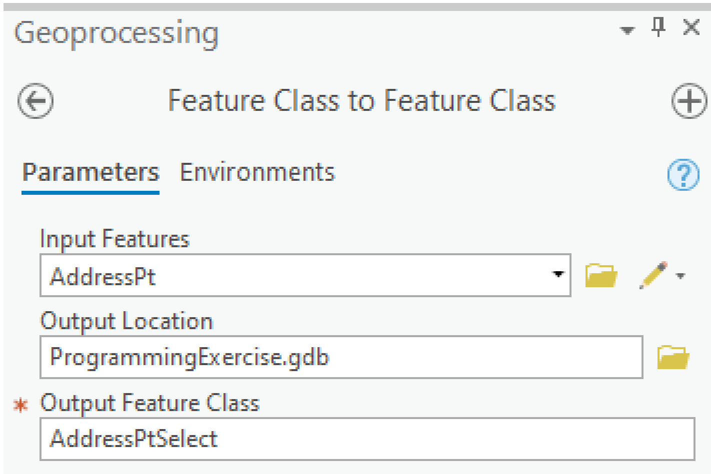
4. Add the AddressPt Feature class from OnionHand.gdb. Zoom in on the catchment illustrated and export it as **CatchPolySelect**.
5. Use **Select by Location** to Create a new feature class that is just address points in **CatchPolySelect**.

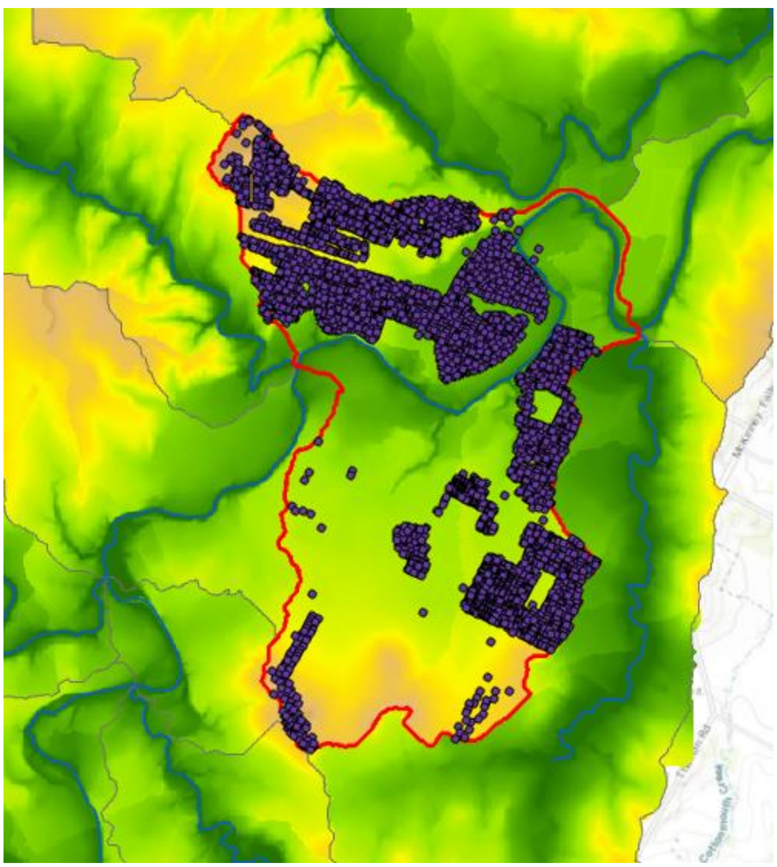


You should see just address points in CatchPolySelect selected.

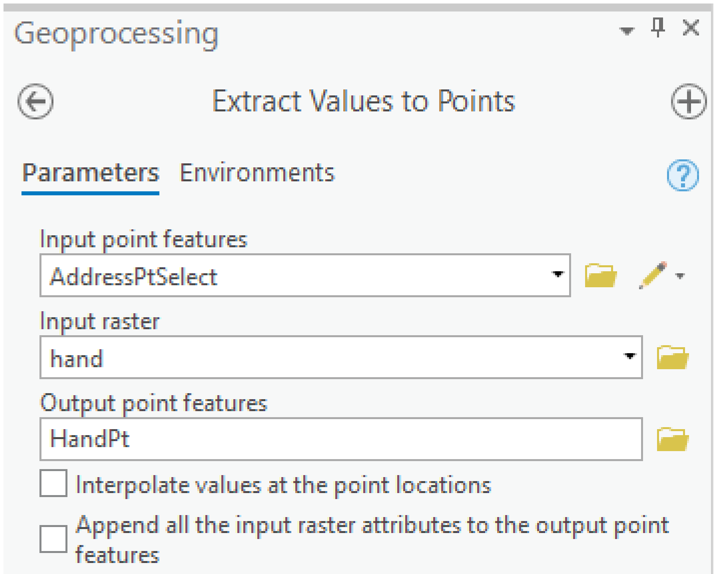


6. Export the selected AddressPt dataset as a feature class **AddressPtSelect.**





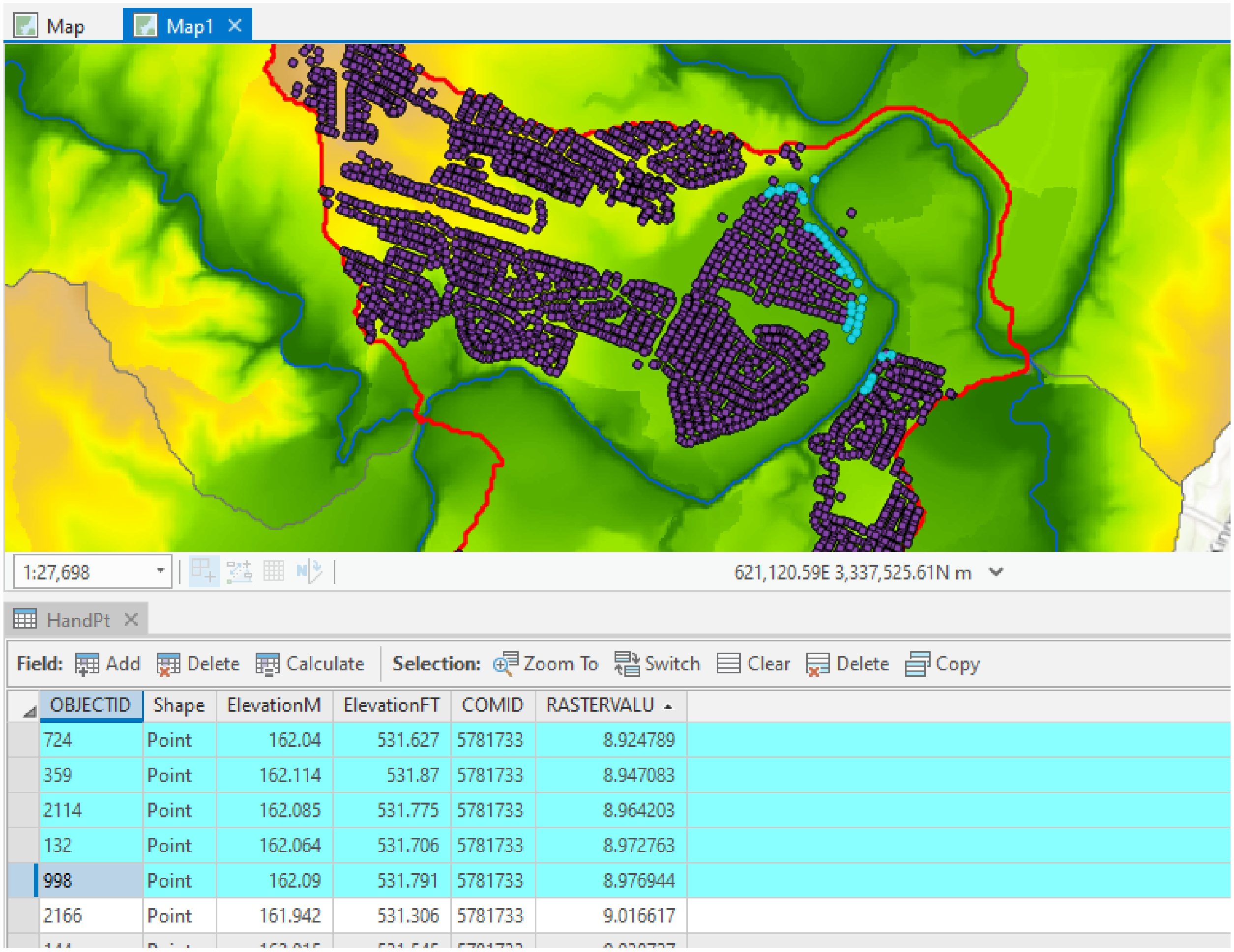
7. Use **Extract Values to Points** to determine HAND from the HAND raster for addresses in this catchment. What you are doing is to overlay the AddressPts on the HAND raster and determine for each AddressPt what the HAND value is for that Address location.



Open the attribute table for HandPt. RASTERVU gives the raster value (HAND) for each address point. Sort Ascending on the RASTERVU column. You will see that the water depth in Onion Creek has to rise to about 8m before many address points start getting flooded in this area.

OBJECTID	Shape	ElevationM	ElevationFT	COMID	RASTERVALU
3711	Point	153.754	504.442	5781733	2.383301
914	Point	159.639	523.75	5781733	7.204514
729	Point	160.928	527.979	5781733	8.202728
3944	Point	161.341	529.334	5781733	8.209214
2007	Point	161.992	531.47	5781733	8.216553
929	Point	161.361	529.4	5781733	8.247982
1284	Point	161.5	529.856	5781733	8.273254
3343	Point	161.2	528.871	5781733	8.350906
1241	Point	161.609	530.213	5781733	8.368011
1236	Point	161.342	529.337	5781733	8.41864
2029	Point	162.302	532.487	5781733	8.436798

Select all address points with RASTERVU less than 9m.



OBJECTID	Shape	ElevationM	ElevationFT	COMID	RASTERVALU
724	Point	162.04	531.627	5781733	8.924789
359	Point	162.114	531.87	5781733	8.947083
2114	Point	162.085	531.775	5781733	8.964203
132	Point	162.064	531.706	5781733	8.972763
998	Point	162.09	531.791	5781733	8.976944
2166	Point	161.942	531.306	5781733	9.016617

Prepare a map that shows addresses where the HAND value is less than 9m. These are addresses subject to flooding if the stage gets to 9 m. Prepare a plot that shows the distribution of inundation depths (as a histogram) for address points within this catchment.

To turn in. A layout showing the catchment that you used for this HAND analysis. On this layout include HAND, potential flood inundation depth based on an assumed flood stage of 9 m (This is 9 m – HAND). Report how many addresses are flooded with a stage of 9 m. Depict flooded address points using a separate symbol. Include your plot that shows the distribution of inundation depths for address points potentially subject to flooding in this catchment.

OK. You are done!

Summary of Items to turn in.

1. *A screen shot from IDLE with your mycode.py program*
2. *A screen shot from Python that shows the completion of mycode.py execution*
3. *Screen shots from ArcGIS showing flow direction and flow accumulation (symbolize these pleasantly) for the Little Bear DEM provided.*
4. *A screen shot from IDLE with your mycode.py program modified to work with logan.tif*
5. *A screen shot from Python that shows the completion of mycode.py execution*
6. *Screen shots from ArcGIS showing flow direction and flow accumulation (symbolize these pleasantly) for the logan.tif DEM.*
7. *A layout showing Catchments (polygons) and DrainageLine (polyline) features nicely symbolized for the Little Bear River.*
8. *A layout showing Catchments (polygons) and DrainageLine (polyline) features nicely symbolized for Onion Creek.*
9. *Your Dem2Watershed.py script modified to also evaluate HAND.*
10. *A layout of HAND for Onion Creek.*
11. *A layout showing the catchment that you used for this HAND analysis. On this layout include HAND, potential flood inundation depth based on an assumed flood stage of 9 m (This is 9 m – HAND). Report how many addresses are flooded with a stage of 9 m. Depict flooded address points using a separate symbol. Include your plot that shows the distribution of inundation depths for address points potentially subject to flooding in this catchment.*