GEOG503A Final Exam

Input data folder: **/data/**

Results folder: **/results/**

Submission: lastname_firstname.zip.

Tasks:

1. Write a script that determines if a point is inside of the box bounded by the points (0, 0) and (1, 1). The script should take two required arguments, an x and a y coordinate separated by space. Hint: Use **`raw_input()`** to get user input and then use **`split()`** to get x and y values. (25 pt.)

Input	Output
0.2 0.4	(0.2,0.4) is in the box.
–0.1 0.5	(–0.1,0.5) is outside the box.
0 0	(0.0,0.0) is in the box.

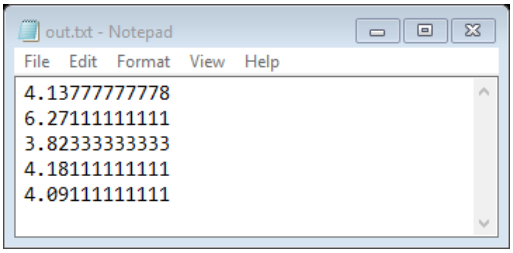
2. Write a script that converts between US dollars and Euros. (Use a rate of 1 US = 0.7 E for this task.) The script should take one numerical required argument and one optional argument (the currency, E or US). If the user gives two arguments, perform the conversion. If the user gives only one argument, a number, assume the number is given in US dollars, warn the user, and perform the conversion. If the user gives no arguments, print a statement explaining how to run it. Examples of test cases and resulting output are shown below: (25 pt.)

Input	Output
45.32 E	45.32 Euros is equivalent to 64.74 US Dollars
55	Since you didn't specify a currency, I'm assuming US to Euros. 55 US Dollars is equivalent to 38.5 Euros
100 US	100.00 US Dollars is equivalent to 70.00 Euros
(No arguments)	Usage: number {currency (US or E)} Example: 100 US

3. Write a script that uses nested FOR -loops to print the output shown below. Hint: Use the **`range()`** function for the numerical (outer) loop and use a hard-coded list [‘a’, ‘b’, ‘c’] for the inner loop. Prefix the inner loop print statements with a tab (‘\t’) to indent them as shown. (25 pt.)

```
1) Hehe
    a) Hoho
    b) Hoho
    c) Hoho
2) Hehe
    a) Hoho
    b) Hoho
    c) Hoho

3) Hehe
    a) Hoho
    b) Hoho
    c) Hoho
4) Hehe
    a) Hoho
    b) Hoho
    c) Hoho
```

4. Write a script that reads a file (‘**`data/crop_yield.txt`**’) with lines of tab separated numeric values, finds the average of each line, and prints these averages to an output file named ‘**`results/out.txt`**’. The lines of the ‘**`out.txt`**’ will appear as follows: (25 pt.)



5. Get a list of the shapefiles in the **`data`** folder. Then for any shapefiles whose name contains the word ‘out’, case-insensitive, use the Get Count (Data Management) tool to determine the

number of records in the attribute table. Report the results as shown in the example. (25 pt.)

```
linesOUT.shp has 530 entries.

outData.shp has 100 entries.

parkOutput.shp has 426 entries.
```

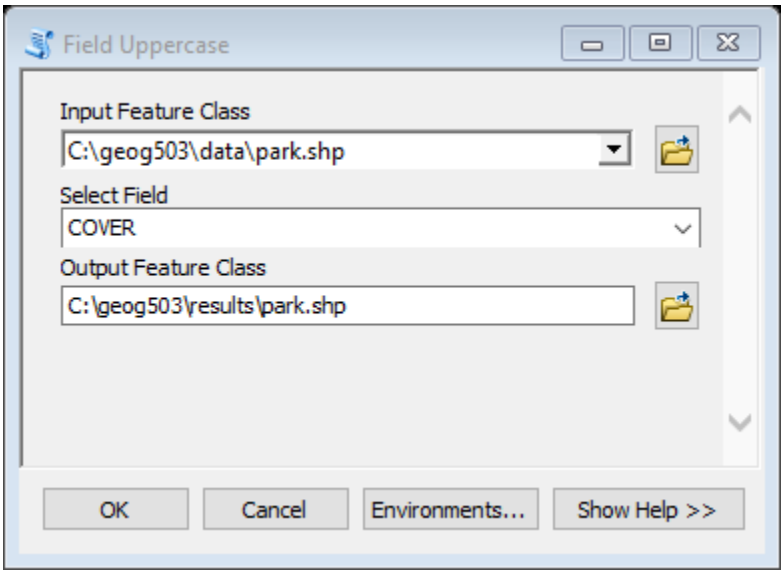
6. Write a script that takes the full path file name of a map document (e.g., '**data/states.mxd**') and prints the '.mxd' file path as well as its data frame and layer names of each layer. Use nested looping and format the output with tabs to achieve the indentation shown in the example below. (25 pt.)

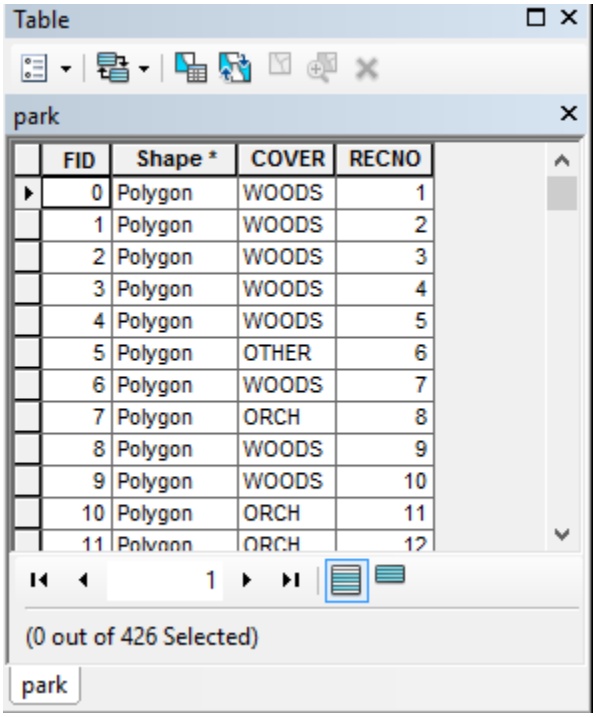
```
Map: data/states.mxd

Frame 0: Northeast
         Layer 0: VT C:\GEOG503\data\USstates\VT.shp
         Layer 1: RI C:\GEOG503\data\USstates\RI.shp
         Layer 2: CT C:\GEOG503\data\USstates\CT.shp
Frame 1: Southeast
         Layer 0: VA C:\GEOG503\data\USstates\VA.shp
         Layer 1: NC C:\GEOG503\data\USstates\NC.shp
```

7. Write a script that lists the rasters in a workspace ('**data/rastTester.gdb**') whose name contain a specified substring ('***cover***'). Also list the field names for each raster. Indent the field names using a tab ('\t') to match the example output shown below. (25 pt.)

```
getty_cover
        OBJECTID
        VALUE
        COUNT
CoverMinus
        OBJECTID
        VALUE
        COUNT
        Location
TimesCOVER
        OBJECTID
        Value
        Count
        blades
```

8. Create a tool that can change all characters in a string field of a shapefile to uppercase. Hint: Use the `Copy` (Data Management) tool to make a copy of the input file in '**`data/park.shp`**' and then use an update cursor to modify the string field input by the user so that all characters in that field are uppercased. In the example below, the output 'COVER' field values become WOODS, ORCH, and OTHER. (25 pt.)





FID	Shape *	COVER	RECNO
0	Polygon	WOODS	1
1	Polygon	WOODS	2
2	Polygon	WOODS	3
3	Polygon	WOODS	4
4	Polygon	WOODS	5
5	Polygon	OTHER	6
6	Polygon	WOODS	7
7	Polygon	ORCH	8
8	Polygon	WOODS	9
9	Polygon	WOODS	10
10	Polygon	ORCH	11
11	Polygon	ORCH	12