2018 Tax Sale Properties

Notes: This python scripts scrapes all PDFs in a folder from the Iron County Notice of Final Tax Sales and creates a csv output file for each. The following column headers are created and the associated data is collected: **ID, Parcel, Acres, and Total Tax Due**. The PDF filenames are unique and are named based on the year from the PDF file.

PDF

2012 IRON COUNTY NOTICE OF FINAL TAX SALE
Thursday, May 24, 2019

Tax Sale # 1 0011467 ORTON HAL L/DORAINE PO BOX 324 PAROWAN, UT 84761-0324 Parcel: A-0491-0001-0000 LOTS 2 & 5 & W1/2 OF LOTS 1 & 6, BLK 13, PLAT C, PAROWAN CITY SURVEY. Acres: 2.48 Total Due: $14,160.28

Tax Sale # 2 0016474 JONES WALTER EUGENE/NYLA JO ANNE 1742 E MADISON AVE EL CAJON, CA 92019-1051 Parcel: A-0636-0000-0000 S 12 RDS OF W1/2 BLK 34, PLAT B, PARAGONAH TOWN SURVEY; 12 RDS E & W BY 12 RDS N& S. Acres: 0.90 Total Due: $5,396.87 (R)

Tax Sale # 3 0131505 ANB VENTURE L L C 477 SHOUP AVE STE 207 IDAHO FALLS, ID 83402 Parcel: A-0962-0000-0000 S 55 AC OF W1/2W1/2W1/2 OF SEC 1,T35S,R11W, SLM; EXCPT THEREFR ANY PART LYING W/IN EXIST CNTY RDWAY; TOG W/ INT IN 3 WTR WELLS DESC REC BK 1000/124. (ANNEX WAS: D-483) SUBJ TO UTIL EASE DESC REC BK 1185/1476; SUBJ TO SEWER LN EASE DESC REC BK 1185/1478. Acres: 55.00 Total Due: $82,636.59

Tax Sale # 4 0131521 ANB VENTURE L L C 477 SHOUP AVE STE 207 IDAHO FALLS, ID 83402 Parcel: A-0962-0001-0000 BEG S 20.11 FT & E 4.46 FT FR NW COR OF SEC 1,T35S,R11W, SLM; SD COR BE LOC S89*29'22"E ALG TWNSHIP LN 2635.89 FT FR N1/4 COR OF SEC 2,T35S,R11W, SLM; S89*18'39"E 654.14 FT; S1*00'31"W 668.36 FT; N89*18'39'W 650.95 FT TO E LN OF W1/4NW1/4 OF SD SEC 1; N0*44'06"E 668.35 FT TO POB; LESS A-962-3 (ANNEX WAS: D-483-1) (LOC SEC 1,T35S,R11W, SLM) SUBJ TO UTIL EASE DESC REC BK 1185/1476; SUBJ TO SEWER LN EASE DESC REC BK 1185/1477. Acres: 9.84 Total Due: $15,059.86

Tax Sale # 5 0131547 ANB VENTURE L L C 477 SHOUP AVE STE 207 IDAHO FALLS, ID 83402 Parcel: A-0962-0001-0001 BEG AT PT ON E R/W LN OF UT HWY-130; SD PT BE S1*08'31"W 688.43 FT ALG SEC LN &S89*18'39"E 9.61 FT FR NW COR OF SEC 1, T35S,R11W, SLM; S89*18'39"E 650.95 FT TO E LN OF W1/4NW1/4 OF SD SEC 1; S1*00'31"W 671.66 FT ALG SD 1/64 LN; N89*18'39"W 647.74 FT TO SD E LN OF HWY; N0*44'06"E 671.65 FT ALG SD E LN OF HWY TO POB. (ANNEX WAS D-483-1-1) SUBJ TO UTIL EASE DESC REC BK 1185/1476. Acres: 10.01 Total Due: $15,244.36

Tax Sale # 6 0261401 ANB VENTURE L L C 477 SHOUP AVE STE 207 IDAHO FALLS, ID 83402 Parcel: A-0962-0002-0000 COM AT NW COR SEC 1,T35S,R11W, SLM; S 1360.00 FT; E 12.72 FT TO TRUE POB; SD PT BE ON E R/W EXIST RDWAY; S89*18'39"E 647.74 FT; S1*00'31"W 675.20 FT; N89*18'39"W 644.52 FT TO E R/W EXIST RDWAY; N0*44'06"E ALG SD RD R/W 675.19 FT TO POB. (ANNEX WAS: D-483-2) SUBJ TO UTIL EASE DESC REC BK 1185/1476. Acres: 10.02 Total Due: $15,259.32

Python Code (20190827_IronCounty.py)

```
# Read a list of Iron County May Tax Sale PDFs in folder, open, scan, and dump data individual text files

# Accumatch

# by Michael Keller

# Aug 27, 2019

# https://www.ironcounty.net/tax-sale/results/

# requires mwkFunctions, pdfplumber, re, datetime, os


# Libraries

from mwkFunctions import getFiles, printFilename, openFile, writeFile, stripAmount

import pdfplumber

import re

from datetime import datetime

import os


# Start Timer

start = datetime.now()


# Booleans

Parcel_Found = False

Total_Found = False

Acres_Found = False

lineStart = False


# Variables

comma = ','

space = ' '

dollarSign = '$'

outString = []

outFilename = 'IronCounty_'

outputType = '.csv'

colHeadings = 'ID,Parcel,Acres,Total'

dataLineStart = 'Tax'

fileCount = 1

tempString = ''

Parcel = ''
```

```
Total = ''
ID = ''
Acres = ''

# Regular expressions
regParcel = 'Parcel:[ ]?[a-zA-Z0-9-()]{2,18}'
##Parcel:[ ]?[a-zA-Z0-9-()]{2,18}
##Parcel:[ ]?[a-zA-Z0-9-()]*;*[ ]*
regTotal = "[$][,0-9]+['.']\d{2}"
regAcres = "Acres:[ ]?[,.0-9]*;*[ ]*"

# Messages
messageSaveBeg = 'Saving \''
messageSaveEnd = '\' file . . .'
messageEOL = '. . . end of line'

PDFs = getFiles('.pdf',os.getcwd())

for x in PDFs:
    ## Message Run
    print(printFilename(x))

    ## Open PDF file for reading
    pdf = pdfplumber.open(x)

    lastPage = len(pdf.pages)

    ## Set counters
    count = 0

    print('Number of Pages: ' + str(lastPage))

    ## Read year from PDF filename
    getYear = re.search('\d{4}',x)
    year = x[getYear.start():getYear.end()]
```

```
    fileName = outFilename + str(year) + outputType

    openFile(fileName,colHeadings)

    for page in pdf.pages:
        p1 = pdf.pages[count]
        pltext = p1.extract_text()
        text = pltext.splitlines()

##    Break for Testing (Page Count)
##        if(count == 2):
##            break

        for item in text:
            if(tempString == ''):
                blob = item.lstrip()
            else:
                blob = item

            if(blob[0:3].strip() == dataLineStart):
                lineStop = False
                lineStart = True

            if((lineStart == True) and (lineStop == False)):
                tempString = tempString + blob

            Total_Found = re.search(regTotal,tempString)
            if not (Total_Found is None):
                # Total value first
                Total = tempString[Total_Found.start():Total_Found.end()]
                Total = stripAmount(Total,True)
                # Parcel
                Parcel_Found = re.search(regParcel,tempString)
                Parcel = tempString[Parcel_Found.start()+7:Parcel_Found.end()]
                Parcel = Parcel.strip(' :;.')
                # Acres
```

```
                Acres_Found = re.search(regAcres,tempString)
                if (Acres_Found is None):
                    Acres = '0.0'
                else:
                    Acres = tempString[Acres_Found.start()+6:Acres_Found.end()]
                    Acres = Acres.strip(' :;.')
                    if (Acres == '0'):
                        Acres = '0.0'
                # ID
                result = tempString[tempString.find('#')+1:len(tempString)].strip()
                carp = result.split(space,2)
                ID = carp[1].strip()

                outString.append(ID + comma + Parcel + comma + Acres + comma + Total)
                tempString = ''

        count = count + 1

    ## Close PDF
    pdf.close()

    ## Write CSV file
    writeFile(fileName,outString)

    ## Clear variables
    ## Make sure outString is clear for next file's data
    outString.clear()
    tempString = ''

    fileCount = fileCount + 1

##    Break for Testing (File Count)
##    if(fileCount == 2):
##        break

# Print file number
```

```
print('Number of files read: ' + str(fileCount - 1))


#end timer
print('Runtime: ' + str(datetime.now() - start))


print(messageEOL)


# EOF
```

Python Code (mwkFunctions.py)

```
## Michael Keller's function module
import os
import re

# Local Variables
_return = '\r'

def getCurrentDirectory():
    os.chdir(os.path.dirname(__file__))
    return os.getcwd()

def getFiles(ext):
    for(dirpath,dirnames,filenames) in os.walk(getCurrentDirectory()):
        return (f for f in filenames if f.endswith(ext))

def printFilename(fn):
    return 'Reading ' + fn + ' file . . .'

def printNumberOfPages(np):
    return 'Number of Pages: ' + str(np)

def readFile(fn):
    # Read existing file
    try:
        file = open(fn,'r')
        print('Opening ' + fn + ' file . . .')
    except FileNotFoundError:
        sys.exit('File: ' + fn + ' was not found')
    except PermissionError:
        sys.exit('Unable able to open file for reading. Do you have permission to open file?')

def openFile(fn,ch):
    # Create output file
    try:
        file = open(fn,'w+')
```

```
        # Write out lines from array
        file.write(ch + _return)
        file.close()
        print('Opening ' + fn + ' file . . .')
    except FileNotFoundError:
        sys.exit('File: ' + fn + ' does not exist')
    except PermissionError:
        sys.exit('Unable able to open output file for writing. Is the file currently open?')

def writeFile(fn,os):
    # Create and append to output file
    try:
        file = open(fn,'a+')
        for taxLine in os:
            file.write(taxLine + _return)
        file.close()
        print('Saving ' + fn + ' file . . .')
    except FileNotFoundError:
        sys.exit('File: ' + fn + ' does not exist')
    except PermissionError:
        sys.exit('Unable able to open output file for writing. Is the file currently open?')

def stripAmount(amt,zero):
    ## Strip dollar sign, commas, and blanks from value
    ## if nothing left after stripping, change value to 0.00
    ## if zero is True, all amounts that are either blank, have
    ## dashes, do not have values, etc. will be converted to '0.00'
    regAmount = "[,0-9]+['.']\d{2}"
    comma = ','
    dollarSign = '$'
    space = ' '
    nospace = ''
    Zero_Value = False

    amt = amt.strip()
    amt = amt.replace(comma,nospace)
```

```
    amt = amt.replace(dollarSign,nospace)
    amt = amt.replace(space,nospace)
    if(zero == True):
        Zero_Value = re.search(regAmount,amt)
        if ((Zero_Value is None) or (len(amt) == 0)):
            amt = '0.00'
    return amt


# EOF
```

CSV Output

	A	B	C	D
1	ID	Parcel	Acres	Total
2	11467	A-0491-0001-0000	2.48	14160.28
3	16474	A-0636-0000-0000	0.9	5396.87
4	131505	A-0962-0000-0000	55	82636.59
5	131521	A-0962-0001-0000	9.84	15059.86
6	131547	A-0962-0001-0001	10.01	15244.36
7	261401	A-0962-0002-0000	10.02	15259.32
8	492572	A-0962-0003-0000	0	433.68
9	461878	A-0963-0001-0000	12	18225.26
10	35409	A-1150-0002-0002-0	0	5446.05
11	314838	A-1165-0003-0001-0	0.27	7021.24
12	242476	A-1208-0008-0000	0.34	1786.89
13	103785	A-1209-0046-0000	0.51	2078.15
14	103751	A-1209-0069-0000	0.55	2078.15
15	106341	A-1211-0053-0000	0.3	2283.33
16	106846	A-1212-0045-0000	1.71	4042.92
17	106903	A-1213-000A-0004	0.45	4083.76
18	457322	A-2059-0012-0000	0.52	3225.84
19	60811	B-0587-0000-0000	0.6	19846.73
20	61975	B-1021-0001-0001	0.29	2658.27
21	73608	B-1135-0028-0000	2.29	29997.06
22	75017	B-1142-0001-0000	0.03	394.83
23	462140	B-1253-0013-0000	3	20760.2
24	336708	B-1379-0013-0000	0.18	3412.97
25	443959	B-1780-0009-0000	0.39	32680.27
26	458841	B-1812-0004-0000	0.73	31002.96
27	139185	B-1825-0000-0000	8.48	4168.58
28	475696	B-1866-0046-0000	0.24	3647.51

Michael Keller
Accumatch
20190827_IronCounty
Aug 27, 2019