Forfeited Land Sale List (Properties Not Sold)

Notes: This python scripts scrapes PDFs creates a CSV output file with the following items: **Book/Plat/Parcel, Case #, Land, Building, Court Cost, Taxes, Total**. The filename is static and not unique.

PDF

Dusty Rhodes

Hamilton County Auditor

Results of the 2019 Forfeited Land Sale

--- Properties Not Sold* ---

*These properties will be offered again next year's Forfeited Land Sale unless they are redeemed by their current owners.

Item:	Book/Plat/Parcel:	Owner	Property Description	Case #	Land/Bldg	Court Cost/ Taxes/ Total:
8	040-0004-0238-00	COGDELL W JENELLE	ARBOR AVE 3 X 125 DRAKES SS ARBOR 293 FT E OF EDWARDS RD	A07-10469	4,680 0	$567.90 $2,259.77 $2,827.67
9	051-0012-0001-00	MOORE RICHARD	STEWART AVE FT IRR R2- T4- S17 SW MORRIS WARD EST	A16-05655	12,950 0	$886.39 $4,191.65 $5,078.04
12	068-0002-0116-00	DURAN VICTOR	PRESLEY ST 25 X 50 EAST HALF OF LOT 364 STEPHEN KEMPERS SUB	A16-05651	3,600 0	$857.69 $23,982.55 $24,840.24
19	098-0005-0166-00	FRIEDHOF INVESTMENTS LLC	HALSTEAD ST 18.77 X 50.41 IRR PTS LOTS 12-13-14 ISAAC J MILLER SUB	A16-06125	6,910 0	$884.59 $12,098.49 $12,983.08
25	108-0003-0031-00	THOMPSON MELVIN JR	BURTON AVE 100 X 122.76 IRR PT H DAYTON EST SUB	A16-01473	10,240 0	$770.30 $4,798.82 $5,569.12
28	112-0001-0026-00	HOPKINS ELLA L	GREENWOOD AV 27.20X98.43 IRR PT LT 32-35 GREENWOOD PLACE SUB PRS 26-51 CONS	A16-04618	6,060 0	$782.70 $9,737.90 $10,520.60
35	126-0002-0062-00	TURNAGE TINA & ANDRE	S WOODMONT AVE 50 X 135 LOT 53 O MEARA SUB	A17-00956	31,970 0	$724.00 $5,884.07 $6,608.07

Python Code (20190715_ForfLandNotSold.py)

```
# Read and dump PDF file data to text file for properties not sold
# Accumatch
# by Michael Keller
# July 15, 2019
# requires pdfplumber, re, datetime

# Libraries
import pdfplumber
import re
from datetime import datetime

# Start Timer
start = datetime.now()

# Booleans
Parcel_Found = False
Case_Found = False
Dollar_Found = False
parcelFound = False

# Variables
parcel = ''
case = ''
land = ''
court_cost = ''
outputType = '.csv'
outputFile = 'PropertiesNotSold'
filename = ''
count = 0
outString = []
tempString = ''
nospace = ''
comma = ','
dollarSign = '$'
_return = '\r'
```

```
regParcel = '\d{3}[-]\d{4}[-]\d{4}[-]\d{2}'
regCase = '.\d{2}[-]\d{5}'
regDollar = '[$]'
colHeadings = 'Book/Plat/Parcel,Case #,Land,Building,Court Cost,Taxes,Total'

# Messages
messageRun = 'Reading \'input.pdf\' file . . .'
messageSaveBeg = 'Saving \''
messageSaveEnd = '\' file . . .'
messageEOL = '. . . end of line'

# PDF File Open
pdf = pdfplumber.open("input.pdf")

print(messageRun)

for page in pdf.pages:
    p1 = pdf.pages[count]
    count = count + 1
    pltext = p1.extract_text()
    text = pltext.splitlines()

    for item in text:
        # Search for Parcel number in line read from PDF
        Parcel_Found = re.search(regParcel, item)
        # Search for any kind of dollar agmount
        Dollar_Found = re.search(regDollar, item)

        # First line of good data in PDF
        if(Parcel_Found is not None):
            parcelFound = True
            parcel = item[Parcel_Found.start():Parcel_Found.end()]
            Case_Found = re.search(regCase, item)
            case = item[Case_Found.start():Case_Found.end()]
            restOfLine = item[Case_Found.end():len(item)].strip()
            restOfLine = restOfLine.replace(comma,nospace)
```

```
            carp = restOfLine.rsplit(' ',3)
            land = carp[0].strip()
            court_cost = carp[1].strip()
            court_cost = court_cost.replace(dollarSign,nospace)
            tempString = parcel + comma + case + comma + land + comma
            lineCount = 0
        elif((parcelFound) and (Dollar_Found is not None)):
            lineCount = lineCount + 1
            if(lineCount == 1):
                # get remaining amounts from second line
                carp = ' ' + item
                carp1 = carp.rsplit(' ', 2)
                building = carp1[1].strip()
                building = building.replace(comma,nospace)
                taxes = carp1[2].replace(comma,nospace)
                taxes = taxes.replace(dollarSign,nospace)
                tempString = tempString + str(building) + comma + str(court_cost) + comma +
str(taxes)
            else:
                # get total
                carp = item[Dollar_Found.start()+1:len(item)].strip()
                total = carp.replace(comma,nospace)
                outString.append(tempString + comma + total)
                parcelFound = False


# PDF File Close
pdf.close()


filename = outputFile + outputType
print(messageSaveBeg + filename + messageSaveEnd)


# Create and open output file
try:
    file = open(filename,'w')
    # Write out lines from array
    file.write(colHeadings + _return)
    # Now dump string array into file
```

```
    for taxLine in outString:
        file.write(taxLine + _return)
    file.close()
except PermissionError:
    print('ERROR: Unable able to open output file for writing. Is the file currently open?')


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


print(messageEOL)


# EOF
```

CSV Output

	A	B	C	D	E	F	G
1	Book/Plat/Parcel	Case #	Land	Building	Court Cost	Taxes	Total
2	040-0004-0238-00	A07-10469	4680	0	567.9	2259.77	2827.67
3	051-0012-0001-00	A16-05655	12950	0	886.39	4191.65	5078.04
4	068-0002-0116-00	A16-05651	3600	0	857.69	23982.55	24840.24
5	098-0005-0166-00	A16-06125	6910	0	884.59	12098.49	12983.08
6	108-0003-0031-00	A16-01473	10240	0	770.3	4798.82	5569.12
7	112-0001-0026-00	A16-04618	6060	0	782.7	9737.9	10520.6
8	126-0002-0062-00	A17-00956	31970	0	724	5884.07	6608.07
9	149-0015-0030-00	A12-00214	1540	0	1572	19059.55	20631.55
10	153-0005-0057-00	A15-02272	7820	0	1186.04	27352.3	28538.34
11	153-0005-0062-00	A16-06564	8520	0	945.98	17947.74	18893.72
12	159-0069-0111-00	A17-04163	4100	0	924.18	14459.61	15383.79
13	168-0002-0044-00	A18-00548	3180	0	776.4	16604.98	17381.38
14	170-0009-0082-00	A17-05414	4990	0	732.9	9796.83	10529.73
15	172-0016-0110-00	A08-10943	830	0	775.9	811.82	1587.72
16	173-0002-0071-00	A16-02551	9020	36710	878.1	11523.03	12401.13
17	173-0005-0015-00	A18-02166	2680	0	762.3	3794.07	4556.37
18	174-0005-0167-00	A17-03670	15700	0	902.69	18513.55	19416.24
19	174-0006-0033-00	A16-03581	6720	0	871.19	19208.67	20079.86
20	174-0007-0004-00	A14-03628	2750	0	750.5	33219.09	33969.59
21	174-0008-0019-00	A16-05972	1500	0	748.7	23029.85	23778.55
22	174-0008-0034-00	A15-01407	2990	0	796.8	27115.01	27911.81
23	176-0019-0112-00	A16-02318	2780	0	1033.67	23839.6	24873.27
24	176-0020-0167-00	A16-05545	2240	0	910.28	8635.11	9545.39
25	176-0022-0103-00	A16-05414	2590	0	723.6	26021.01	26744.61
26	177-0035-0029-00	A17-05460	1950	0	1069.48	19092.78	20162.26
27	177-0035-0060-00	A16-05159	2990	0	940.68	26869.21	27809.89
28	178-0026-0085-00	A16-06651	2830	0	869.59	21119.32	21988.91

Michael Keller
Accumatch
20190715_ForfLandNotSold
July 17, 2019