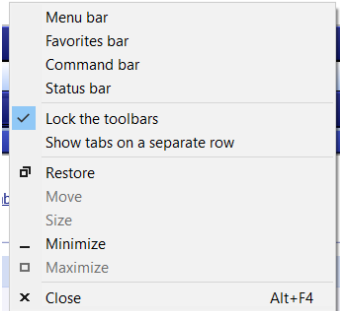


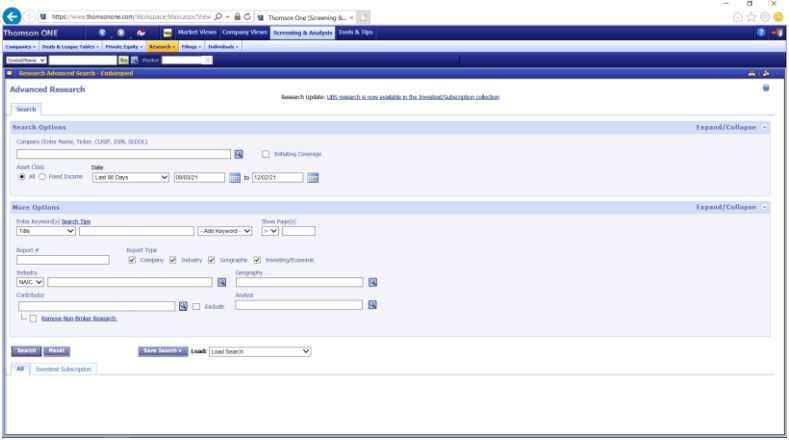
Summary of Main Processing Pipeline (20211202)

	Task	Codes	Input	Output
1	Download Raw Data			
1.1	Download Thomson One's Conference Calls [L]	mouse_key_recorder.py automatic_download .py	-	O1: [yyyymmdd-yyyymmdd].pdf O2: [yyyymmdd-yyyymmdd].xls
2	PDF Processing			
2.1	Convert Conference Calls from .pdf to .txt [M]	pdftransfer.sh	O1	O3: [yyyymmdd-yyyymmdd].txt
2.2	Split Conference Call .txt files to separate out individual conference calls, and combine with report information from .xls files [M, L]	ParseCCpdf.jl	O3, O2	O4: [yyyymmdd-yyyymmdd].csv
3	Firm Identification (Firm Name Matching)			
3.1	Download Compustat datasets [L]	-	-	O5: ciqcompany.sas7bdat O6: ciqcountrygeo.sas7bdat O7: wrds_gvkey.sas7bdat
3.2	Download Hassan dataset [L]	-	-	O8: Hassanfile_raw_updated20219030.csv
3.3	Process Compustat and Hassan datasets into usable and truncated .csv files. [L]	convert_sas7bdatto csv.py hassanfilecsv_viewable_truncate.py	O5, O6, O7, O8	O9: ciqcompany.csv O10: ciqcountrygeo.csv O11: wrds_gvkey.csv O12: Hassanfile_raw_updated20219030_truncated.csv
3.4	Match titles in conference calls with firm names in Hassan and Compustat datasets, with both exact and fuzzy matching [M, L]	Based on the current flow of work, stage 3 is technically done after stage 4 since it used entryfiles_combined, though the original codes were written as stage 4 after stage 3.	O9, O10, O11, O12	O13: CC_List[yyyy].csv O14: CC_List_2020-2021.csv
4	Keyword Identification			
4.1	Make a folder structure with x groups (default: x = 50) [M, L]	mkdir.py dividefilesequallyinto folders.py	-	-
4.2	Make a list of keywords and template entry file [L]	-	-	O15: keyterms.txt O16: Entry mask.xlsx

4.3	Identify keywords in each conference call [M, L]	keyword_ident_1.py keyword_ident_1.sh	O15, O4	O17: FR5.csv
4.4	Extract all paragraphs from each conference call that contains a specific keyword [M, L]	keyword_ident_2.py keyword_ident_2.sh	O17	O18: TotalCircnew.xlsx
4.5	Cleans the identified matches and merges with gvkey dataset [L]	mergeclean.do	O18, O14	O19: cric1_newtotal.xlsx
4.6	Make a paragraph record file that splits the number of entries into groups of 500 [L]	make_paragraphrecord.py	O19	O20: paragraphrecord.xlsx
4.7	Bold the keywords and separate file into "entryfiles", each containing 500 entries. [M, L]	makeentryfiles.py makeentryfiles.sh	O19, O20, O15, O16	O21: [i].xlsx
4.8	Combine entry files [L]	combine_entryfilesjson.py combine_entryfilessixun.py combine_sixunand jasonentryfiles.py	O21	O22: entryfiles_combined.xlsx
5	Get Front Page Descriptions			
5.1	Extract front page descriptions from conference calls [M, L]	extractdescriptioninfront page.py extractdescriptioninfront page.sh copyfiles.py copyfiles.sh	O21, O2, O3	O23: [yyyymmdd-yyyymmdd]_withfront pagedesc.xlsx
5.2	Manually check through error cases and correct accordingly [L]	-	O23	O24: [yyyymmdd-yyyymmdd]_withfront pagedesc.xlsx
5.3	Combine xls files [L]	combine_xlsfiles_with description.py	O24	O25: xlscombined_with frontpagedescription .xlsx
5.4	Match and add front page descriptions to combined entry files [L]	-	O25, O22	O26: entryfiles_combined.xlsx (updated)

* M = Mercury, L = Local. [M] / [L] means this stage can be run on Mercury / locally (on your Booth Windows laptop) respectively. [M, L] means this stage can be run on both Mercury and your local laptop, where Mercury is preferred for large datasets and local is preferred for initial testing, debugging and small datasets.

	Task	Codes
1	Download Raw Data	
1.1	Download Thomson One's Conference Calls [L]	<p>mouse_key_recorder.py: Works, but screen co-ordinates returned don't necessarily match up with the screen for some reason.</p> <p>automatic_download.py: Main loop works, but screen co-ordinates depends on screen size. Current code works for the Booth laptop, Lenono Thinkpad X1 Extreme (Windows 10) that is not connected to a HDMI screen. If on another screen, the co-ordinates will be off. The other parts of the code don't work (error catching) don't work as well.</p> <p>The screen resolution details (https://whatsmyscreensize.com/):</p> <ul style="list-style-type: none"> • Screen Resolution • Width: 1920 • Height: 1080 • Device Pixel Ratio: 1.25 • Display Dimensions (width x height): 16.0" x 9.0" • Screen Diagonal: 18.4" Screen <p>Additional specific settings when running code – goal is to hide away extraneous elements on the screen, so that no scrolling is needed to be able to click on all co-ordinates.</p> <ul style="list-style-type: none"> • Windows Taskbar: "Automatically hide the taskbar in desktop mode" is turned on • Internet Explorer settings:  <p>The screenshot shows the 'Internet Options' dialog box in Windows. The 'Content Advisor' tab is selected. Under 'Content Advisor', the 'Lock the toolbars' checkbox is checked. Other options visible include 'Show tabs on a separate row', 'Restore', 'Move', 'Size', 'Minimize', 'Maximize', and 'Close'. The 'Close' button has the keyboard shortcut 'Alt+F4' next to it.</p> <ul style="list-style-type: none"> • Screen looks like this:

		
2	PDF Processing	
2.1	Convert Conference Calls from .pdf to .txt [M]	pdftransfer.sh: Works, but better if input and output folders can be selected as an argument. Currently, file needs to be moved
2.2	Split Conference Call .txt files to separate out individual conference calls, and combine with report information from .xls files [M, L]	ParseCCpdf.jl: Works, but there is a specific line of code that doesn't work when the entire file is run, and only when the code is run line by line. Note: This is a different version from the original code, and works only for newer conference calls that have a standardized format.
3	Firm Identification (Firm Name Matching)	
3.1	Download Compustat datasets [L]	-
3.2	Download Hassan dataset [L]	-
3.3	Process Compustat and Hassan datasets into usable and truncated .csv files. [L]	convert_sas7bdattocsv.py: works hassanfilecsv_viewable_truncate.py: works
3.4	Match titles in conference calls with firm names in Hassan and Compustat datasets, with both exact and fuzzy matching [M, L]	Based on the current flow of work, stage 3 is technically done after stage 4 since it used entryfiles_combined, though the original codes were written as stage 4 after stage 3. The current output is cc_list, but we want to generate something of that format without the gvkeys, for 4.5.
4	Keyword Identification	
4.1	Make a folder structure with x groups (default: x = 50) [M, L]	mkdir.py: works dividefilesequallyintofolders.py: works, but can be generalized

4.2	Make a list of keywords and template entry file [L]	-
4.3	Identify keywords in each conference call [M, L]	<p>keyword_ident_1.py keyword_ident_1.sh: works, but the “parallelization” is currently implemented manually (split into 5 code files, each doing 10/50 folders). It’ll be great to do this automatically with some parallelization / batch processing code (i.e. have just 1 code file, with some parameter, then use this to submit say 50 jobs on Mercury, each doing 1/50 folders). Dask arrays would further increase performance.</p> <p>Also, currently files are copied and split over 50 folders, creating duplicates. Ideally, the code runs without having to copy the files over to 50 folders in the first place.</p>
4.4	Extract all paragraphs from each conference call that contains a specific keyword [M, L]	<p>keyword_ident_2.py keyword_ident_2.sh: works. This original code returns the first match and not the remaining matches.</p>
4.5	Cleans the identified matches and merges with gvkey dataset [L]	<p>mergeclean.do: works, but was written with the idea that gvkeys have already been retrieved. Will need to rewrite (possibly in python) to remove that; then with the cc_lists generated without gvkeys, this can be run.</p>
4.6	Make a paragraph record file that splits the number of entries into groups of 500 [L]	<p>make_paragraphrecord.py: works.</p>
4.7	Bold the keywords and separate file into “entryfiles”, each containing 500 entries. [M, L]	<p>makeentryfiles.py, makeentryfiles.sh: works. Renamed from makebold to makeentryfiles because that’s a better description of what it does.</p> <p>Currently it takes in a template excel file and then does some formatting (including bolding keywords). It’s complicated and should be simplified.</p>
4.8	Combine entry files [L]	<p>combine_entryfilesjson.py combine_entryfilessixun.py combine_sixunandjsonentryfiles.py: works; consolidation would be better. Currently the format is slightly different across versions; once the pipeline is fixed, this shouldn’t happen.</p>
5	Get Front Page Descriptions	
5.1	Extract front page descriptions from conference calls [M, L]	<p>extractdescriptioninfrontpage.py extractdescriptioninfrontpage.sh: works, with a caveat.</p>

		<p>The initial process went something like: code v1 -> worked for some files code v2 -> out of remaining files, worked for some files ... code final -> out of remaining files, worked for all except those that need to be handled manually</p> <p>But it's not certain if code final works for all files. Will need to run on all files to confirm.</p> <p>copyfiles.py copyfiles.sh: works</p>		
5.2	Manually check through error cases and correct accordingly [L]	-		
5.3	Combine xls files [L]	combine_xlsfiles_withdescription.py: works		
5.4	Match and add front page descriptions to combined entry files [L]	-		

Other notes:

- Currently, conference calls are grouped by 1 (2001-2020) and 2 (2020-2021) with some overlap, corresponding to 2 pulls of the data. It would be nice to write code that allows you to choose one folder to work with.
- Ideally, there's just one file to update locations, so that a name change won't imply that the folder path in every code needs to be updated.

Other checks:

- After the pipeline is established, run it from start to end and compare against current results.