As you begin working more and more with gene data, you may come across annotations known as [Gene Ontology (GO) (Links to an external site.)](http://geneontology.org/docs/ontology-documentation/).  An ontology provides a common vocabulary regarding concepts and how these concepts are related.  In particular, the GO Consortium is a large bioinformatics effort to standardize the representation of gene and gene product attributes across all species.  It provides a controlled vocabulary of terms for describing gene product characteristics and gene product annotation data.  For this practice, you will be working with two files from the Consortium:

* [go-basic.obo](https://canvas.instructure.com/courses/1688943/files/83284664/download?wrap=1) (source [here (Links to an external site.)](http://geneontology.org/docs/download-ontology/))
* a subset of [goa\_human.gaf](https://canvas.instructure.com/courses/1688943/files/83284667/download?wrap=1" \o "goa_human_subset.gaf) (source [here (Links to an external site.)](http://current.geneontology.org/products/pages/downloads.html))

This is [GO term format (Links to an external site.)](http://geneontology.org/docs/GO-term-elements) and this is [GAF format (Links to an external site.)](http://geneontology.org/docs/go-annotation-file-gaf-format-2.1/), so that you know how to parse each file, as well as the [GO Graph (Links to an external site.)](http://geneontology.org/docs/ontology-relations/), so that you understand the tree-like structure of GO terms.

In general, a GO term will look something like this, though some terms may be missing some elements or have additional elements (i.e. subset, alt\_id, etc.) listed:

[Term]  
id: GO:0000001  
name: mitochondrion inheritance  
namespace: biological\_process  
def: "The distribution of mitochondria, including the mitochondrial genome, into daughter cells after mitosis or meiosis, mediated by interactions between mitochondria and the cytoskeleton." [GOC:mcc, PMID:10873824, PMID:11389764]  
synonym: "mitochondrial inheritance" EXACT []  
is\_a: GO:0048308 ! organelle inheritance  
is\_a: GO:0048311 ! mitochondrion distribution

In particular, you will want to capture from each GO term, the id and is\_a values for this assignment.

As for the GAF format, each annotation will look something like this (some columns omitted for easier readability):

UniProtKB A0A024R0T9 APOC4-APOC2 GO:0006629 GO\_REF:0000002 IEA ...

In particular, you will want to capture from each annotation the DB Object ID and the GO ID values for this assignment.

Your goal will be to create an annotation report of human genes and their list of associating GO terms (including the parent terms) so that it matches exactly with the ‘results.tsv’ (GO terms are sorted).

**Instructions:**

**For this pratice, your new Python 3 program must:**

* **have a function named split\_terms() that accepts one parameter:**
  + *filename*: a file path to a GO terms file

This function should open the file and split the file into individual GO terms.  It should return the separated terms as a list; if the file cannot be found, return an empty list.

* **have a function named map\_protein\_to\_go() that accepts one parameter:**
  + *filename*: a file path to a GO annotations file (the GAF)

This function should open the file in order to build the mapping relationship between the protein ID (DB Object ID) and its list of associating GO terms.  Note that some proteins are annotated with duplicate GO terms, e.g. O00232 is annotated with GO:0005576 twice, so think about which collection would be most appropriate to use to store this list.

Once built, return the dictionary.  If the file cannot be found, return an empty dictionary.

* **have a function named parse\_go\_term() that accepts one parameter:**
  + *term*: a single GO term

This function is responsible for parsing for the following elements:

* + - ID
    - is\_a

Grab only the GO IDs from each element, e.g. "GO:0000001", not "id: GO:0000001" or "is\_a: GO:0000001 ! mitochondrion inheritance"

If you inspect the file, you will notice that some terms may have one is\_a value, multiple is\_a values, or none at all.  You will want to capture all the is\_a values, if any, in a collection.  If there are none, then the collection should be empty.

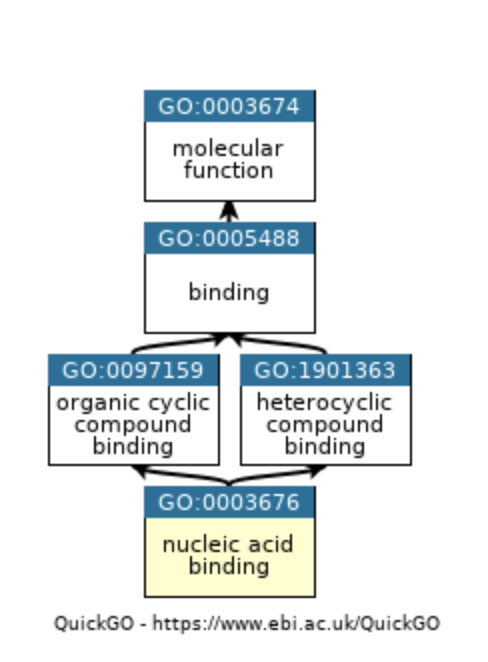
Once parsed, return the ID and is\_a values.

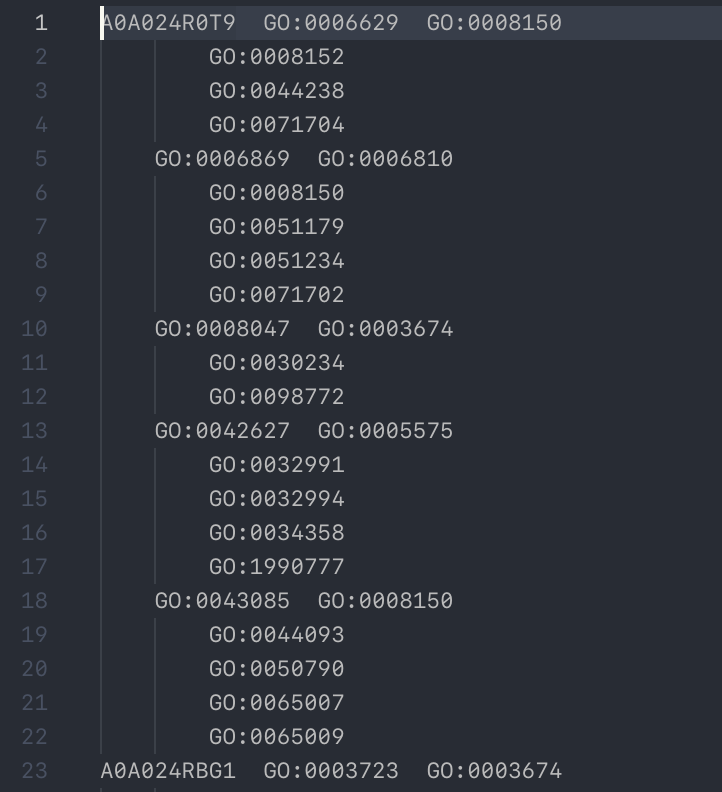
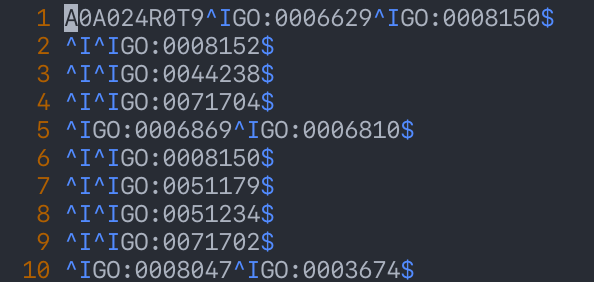
* **have a function named find\_parent\_terms() that accepts two parameters:**
  + *go\_id*: a single GO ID
  + *go\_dict*: a dictionary of GO terms

This function should recursively look for all of the parent terms of the starting GO term and return them as a collection (you will need to determine which collection to use).

For example, GO:0000166 should have 6 terms in the collection:

* + - GO:1901265
    - GO:0036094
    - GO:1901363
    - GO:0097159
    - GO:0005488
    - GO:0003674

even though it only has two direct parent (is\_a) terms.  
  
Note that some GO terms may share the same parent term as a sibling term. For example, GO:0097159 and GO:1901363, which are the parent terms of GO:0003676, both share GO:0005488 as a parent term:  
  
Therefore, you will need to think about how to best store this list so that the result does not contain duplicates.

* **map a GO term to its direct parent GO terms.  Use one of the functions defined above to help build this mapping relationship.**
* **write an annotation report that looks something like this:**  
    
  Essentially, the report should contain all the protein IDs in the first column, its list of associating GO terms in the second column, and each GO term's parent terms in the third column.  You will want to format the report where the protein name is printed only for the first GO term and the GO term is only printed for the first parent GO term.  Every value should be aligned to their respective columns (protein IDs in column 1, GO terms in column 2, parent terms in column 3).  
    
  **Note**: it may be hard to visually judge if the output file is formatted correctly in the text editor, but if you open it in Excel or use vim then enter :set list, you should find that all the terms are in their respective columns, e.g.  
    
  
* accept three command-line arguments, with the last one being optional:
  + *input\_terms* - input GO terms file, must be .obo
  + *input\_annotations* - input gene associations file, must be .gaf
  + *output\_filename* - output filename **without** the file extension, use .tsv as the extension (default output file: results.tsv)

Implement the same logic and provide helpful error messages as your previous assignment.

* **not have any code (besides function definitions) in the main namespace.  Utilize main() and if \_\_name\_\_ == "\_\_main\_\_".**
* **use the modules: sys and re. only**