CSC 247/447 Programming Assignment 3 **Due Monday April 26th**

Write a program to perform reference resolution on simple stories, where the input will be a simplified semantic representation of sentences.

In particular, here is a simple story and the corresponding input you would get. The input consists of a sentence, a summary of the subject NP, the main verb, a summary of the direct object (if any), and then a list of any other NPs that occur in the sentence.

```
"Jack saw Sam at the party"
        (NAME "Jack" ONT::PERSON J1)
        (VERB "saw" ONT::ACTIVE-PERCEPTION V1)
        (NAME "Sam" ONT::MALE-PERSON S1)
        (THE "party" ONT::GATHERING-EVENT P1)

"Sam gave him a drink"
        (NAME "Sam" ONT::MALE-PERSON S2)
        (VERB "gave" ONT::GIVING V2)
        (PRO "him" ONT::MALE-PERSON H1)
        (A "drink" ONT::BEVERAGES D1)

"It tasted sour"
        (PRO "it" ONT::NATURAL-OBJECT I1)
        (VERB "tasted" ONT::ACTIVE-PERCEPTION V3)
```

You program should read such files and output equality relations between all the discourse entities.
Your output for the above input might look like this:

```
        (COREF S2 S1)      — i.e., the two instances of Sam are the same person
        (COREF H1 J1)      — i.e., the pronoun in the second sentence refers to Jack
        (COREF I1 D1)       — i.e., the pronoun in the third sentence refers to the drink
```

The chapter on discourse reference in the readings provides many examples of issues related to reference resolution, and you should try to capture as many cases as you can using the ontology but you are not expected to use any common sense reasoning.

Here is a formal specification of the inputs you may receive:

The format of a basic entry is
  (<spec> <word> <type> <DE>)
    where <spec> may be THE, THE-SET, PRO, PRO-SET, A, SOME, INDEF-SET, NAME, and VERB
  <word> is the head word of the phrase

<type> is the inferred type in the ontology — note the input will assume we have been able to use selectional constraints placed by the verb, so you see "it" above is ONT::NATURAL-OBJECT rather than the more abstract ONT::REFERENTIAL-SEM.
<DE> a unique identifier produced for each phrase

We will also provide a special syntax for conjoined noun phrases, allowing a fifth argument that is a list of the explicit components. So the phrase "John and Mary" would be represented as

```
(THE-SET "John and Mary" ONT::PERSON JM1
        (NAME "John" ONT::MALE-PERSON J2)
        (NAME "Mary" ONT::FEMALE-PERSON M2))
```

As always, your code should be well documented and you should provide a report outlining the strategy your program implements, and discuss its strengths and weaknesses.

Things to watch out for:

— there's no simple type restriction strategy — for instance in "A police officer pulled be over. She gave me a ticket", there was no indication initially that police officer was female.

— reflexive constraints — note in "John saw him", the pronoun can't refer to John

— inter and intra sentence reference — not all antecedents are in a previous sentence, as in "John laughed as he opened the present".

— a pronoun may refer to the verb — "John choked. It happened so quickly"

— when there is a chain of coreferring expressions, always refer to the last one mentioned. e.g., in a story with (THE "dog" X D1) …. (PRO "it" X I1) … (PRO "it" X I2), the output would be (COREF I1 D1) and then (COREF I2 I1) — not (COREF I2 D1).

You do not need to deal with possessive pronouns, as in "His car" as the current representation doesn't have a way to capture them. But for a bonus, you can develop an input for such constructions and implement a strategy to process them. Such examples, however, will not be included in the test data. If you work on the bonus part, please include your own example in your report.

Here are a couple of more examples, which will also be included in your development files. Each example is a story containing one or more sentences. You should output equality relations from each sentence to each line of your output file.

Example 1 – This shows your program processing sentence by sentence

"The large dog ate a bone"

```
(THE "large dog" ONT::NONHUMAN-ANIMAL D5)
(VERB "ate" ONT::EAT V5)
(A "bone" ONT::FOOD B5)
```

This is the first sentence from the story and this is the only sentence we have so far. There is nothing to resolve yet, so the output for this sentence is an empty pair of parentheses.

You read the next sentence:

“It choked on the bone”
(PRO “It” ONT::ANIMAL I5)
(VERB “choked” ONT::OBSTRUCTED-BREATHING C5)
(THE “bone” ONT::FOOD B6)

Output is (COREF I5 D5) (COREF B6 B5), based on the first and the second sentence.

“We pulled the bone out”
(PRO-SET “We” ONT::PERSON W5)
(VERB “pull” ONT::PULL P5)
(THE “bone” ONT::FOOD B7)

The output should be (COREF B7 B6).
(note, pronouns such as I, me and we are defined by context so you won’t find a coreference unless they are repeated!)

“It was so grateful”
(PRO “It” ONT::ANIMAL I6)
(VERB “was” ONT::HAVE-PROPERTY V7)

Output is (COREF I6 I5).

Example 2 This shows the actual input and output formats expected

The input file is:
“Some boys and girls met at the park”
(INDEF-SET “Some boys and girls” ONT::PERSON BG2 (SOME “boys” ONT::MALE-PERSON B11) (SOME “girls” ONT::FEMALE-PERSON G11))
(VERB “met” ONT::MEET V6)
(THE “park” ONT::REGION-FOR-ACTIVITY P11)

“They sang a song”
(PRO-SET “They” ONT::PERSON T8)
(VERB “sang” ONT::SING S11)
(A “song” ONT::MUSIC S12)

“The boys didn’t like it”
(THE “boys” ONT::MALE-PERSON B12)
(VERB “like” ONT::APPRECIATE V12)
(PRO “it” ONT::EVENT-OF-ACTION I12)

“They left”
(PRO-SET “they” ONT::PHYS-OBJECT T13)

```
        (VERB “left” ONT::DEPART V14)
```

The output file for this story is

```
()
(COREF T8 BG2)
(COREF B12 B11) (COREF I12 S11)
(COREF T13 B12)
```

Details for Submission of your Program

Submit your program and the documentation on Blackboard. Your program should be defined so that it can be run with the command line

```
        python3 [yourprogram].py [input-file] [output-file]
```

Each input file would contain a story and look like this:

```
“The large dog ate a bone”
        (THE “large dog” ONT::NONHUMAN-ANIMAL D5)
        (VERB “ate” ONT::EAT V5)
        (A “bone” ONT::FOOD B5)

“It choked on the bone”
        (PRO “It” ONT::ANIMAL I5)
        (VERB “choked” ONT::OBSTRUCTED-BREATHING C5)
        (THE “bone” ONT::FOOD B6)

“We pulled the bone out”
        (PRO-SET “We” ONT::PERSON W5)
        (VERB “pull” ONT::PULL P5)
        (THE “bone” ONT::FOOD B7)

“It was so grateful”
        (PRO “It” ONT::ANIMAL I6)
        (VERB “was” ONT::HAVE-PROPERTY V7)
```

Your output should be like this (the output should be empty parentheses if there is nothing to resolve):

```
()
(COREF I5 D5) (COREF B6 B5)
(COREF B7 B6)
(COREF I6 I5)
```