Name of the Student: _____

Below are the solutions for the last question of the previous assignment; coupled with the discussion in class, re-reading of some parts of the last weeks material and this weeks assignment will solidify the basics of feature structures and unification. The latter would need more training and discussion to be fully grasped; especially the use of variables in feature structures and how feature structures are integrated with phrase structure rules. We will deal with them next week in class.

$$\begin{bmatrix} \text{F} & \text{x} \end{bmatrix} \sqcup \begin{bmatrix} \text{G} & \text{y} \end{bmatrix} = \begin{bmatrix} \text{F} & \text{x} \\ \text{G} & \text{y} \end{bmatrix}$$

$$\begin{bmatrix} \text{F} & \begin{bmatrix} \text{J} & \text{k} \end{bmatrix} \end{bmatrix} \sqcup \begin{bmatrix} \text{G} & \begin{bmatrix} \text{J} & \text{s} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \text{F} & \begin{bmatrix} \text{J} & \text{k} \end{bmatrix} \\ \text{G} & \begin{bmatrix} \text{J} & \text{s} \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{F} & \begin{bmatrix} \text{J} & \text{k} \end{bmatrix} \end{bmatrix} \sqcup \begin{bmatrix} \text{J} & \text{k} \end{bmatrix} = \begin{bmatrix} \text{F} & \begin{bmatrix} \text{J} & \text{k} \end{bmatrix} \\ \text{J} & \text{k} \end{bmatrix}$$

Question 1

Please read Tallerman (2015), chapter 4, closely.[1] We will continue to have such readings to get well-versed in linguistic terminology and concepts.

Question 2

I will give you a tiny grammar and ask you to slightly extend it. But before that we need to fix some notational conventions.

Remember that feature structures are functions: at a given depth, you cannot have two feature specifications with the same name. Also remember that we exploited this property to access information embedded in a feature structure via path specifications. Given a feature structure named F in (1), you can reach its various components in the way exemplified in (2):

(1)
$$\begin{bmatrix} \text{TYPE} & \text{phrase} \\ \text{HEAD} & \begin{bmatrix} \text{TYPE} & \text{verb} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{itr} \\ \text{SPR} & + \end{bmatrix} \end{bmatrix}$$

(2)
$$\begin{aligned} F(\langle\ \text{TYPE}\ \rangle) &= \text{phrase} \\ F(\langle\ \text{VAL COMPS}\ \rangle) &= \text{itr} \end{aligned}$$

Now we simplify this notation as follows:

(3)
$$\begin{aligned} F(\langle \text{TYPE} \rangle) &\equiv \langle F\ \text{TYPE}\ \rangle = \text{phrase} \\ F(\langle \text{VAL COMPS} \rangle) &\equiv \langle F\ \text{VAL COMPS}\ \rangle = \text{itr} \end{aligned}$$

[1] By the way, the Turkish example (28) is such a curious pick.

Now comes the grammar; first the rules, then I will explain how to read the rules, then will come the lexicon:

(4) Phrase structure rules with feature constraints:
Rule 1:

$$X_0 \rightarrow X_1 \tag{1}$$
$$\langle X_0 \text{ PHON} \rangle = \langle X_1 \text{ PHON} \rangle \tag{2}$$
$$\langle X_0 \text{ HEAD} \rangle = \langle X_1 \text{ HEAD} \rangle \tag{3}$$
$$\langle X_1 \text{ VAL COMPS} \rangle = \text{nil} \tag{4}$$
$$\langle X_1 \text{ VAL SPEC} \rangle = - \tag{5}$$
$$\langle X_0 \text{ VAL SPEC} \rangle = + \tag{6}$$

Rule 2:

$$X_0 \rightarrow X_1 \quad X_2 \tag{7}$$
$$\langle X_0 \text{ HEAD CAT} \rangle = \langle X_2 \text{ HEAD CAT} \rangle \tag{8}$$
$$\langle X_1 \text{ HEAD CAT} \rangle = \text{n} \tag{9}$$
$$\langle X_2 \text{ HEAD CAT} \rangle = \text{v} \tag{10}$$
$$\langle X_1 \text{ VAL SPEC} \rangle = + \tag{11}$$
$$\langle X_2 \text{ VAL SPEC} \rangle = + \tag{12}$$
$$\langle X_0 \text{ PHON} \rangle = \langle X_1 \text{ PHON} \rangle \# \langle X_2 \text{ PHON} \rangle \tag{13}$$
$$\langle X_1 \text{ HEAD AGR} \rangle = \langle X_2 \text{ HEAD AGR} \rangle \tag{14}$$

Here is how to read and what to do with these rules: The symbols X_i in (4) are variables that will be instantiated by feature structures. The equalities following the phrase structure rule are constraints that need to be satisfied by a full instantiation – it might be helpful to think of them as declarative well-formedness constraints for phrase structure trees, rather than instructions to perform certain operations. In instantiating the variables in rules you have two options: (i) either pick something from the lexicon, or (ii) use some feature structure you already produced. The '#' sign in constraint (13) is the concatenation operation; it tells to concatenate the two values. Here is the lexicon:

(5) Lexicon:

$$\begin{bmatrix} \text{PHON} & \text{/jane/} \\ \text{HEAD} & \begin{bmatrix} \text{CAT} & \text{n} \\ \text{AGR} & \begin{bmatrix} \text{PER} & \text{3rd} \\ \text{NUM} & \text{sg} \end{bmatrix} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{nil} \\ \text{SPEC} & + \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{PHON} & \text{/sleep/} \\ \text{HEAD} & \begin{bmatrix} \text{CAT} & \text{v} \\ \text{AGR} & \begin{bmatrix} \text{NUM} & \text{pl} \end{bmatrix} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{nil} \\ \text{SPEC} & - \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{PHON} & \text{/sleeps/} \\ \text{HEAD} & \begin{bmatrix} \text{CAT} & \text{v} \\ \text{AGR} & \begin{bmatrix} \text{PER} & \text{3rd} \\ \text{NUM} & \text{sg} \end{bmatrix} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{nil} \\ \text{SPEC} & - \end{bmatrix} \end{bmatrix}$$

The grammar is intended to generate *Jane sleeps* while not allowing **Jane sleep*. Your task is to,

(a) understand how the grammar works;

(b) check whether the grammar does what it aims to do;

(c) extend it, so that you generate *I/we/you/they sleep*, without generating their ungrammatical counterparts.

Now comes some extra material that is not a part of the assignment. The same grammar can be represented using variables embedded in feature structures, instead of separately stated constraints. This design needs a better grasp of unification; therefore we started with explicit constraint specification. We will discuss the format below in the class. The lexicon is the same as above, phrase structure rules become:

(6) Phrase structure rules with re-entrant feature specifications:
Rule 1:

$$\begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & \boxed{2} \\ \text{VAL} & \begin{bmatrix} \text{SPEC} & + \end{bmatrix} \end{bmatrix} \rightarrow \begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & \boxed{2} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \text{nil} \\ \text{SPEC} & - \end{bmatrix} \end{bmatrix}$$

Rule 2:

$$\begin{bmatrix} \text{PHON} & \boxed{1}\#\boxed{2} \\ \text{HEAD} & \begin{bmatrix} \text{CAT} & \boxed{3} \end{bmatrix} \end{bmatrix} \rightarrow \begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & \begin{bmatrix} \text{CAT} & \text{n} \\ \text{AGR} & \boxed{4} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{SPEC} & + \end{bmatrix} \end{bmatrix} \quad \begin{bmatrix} \text{PHON} & \boxed{2} \\ \text{HEAD} & \begin{bmatrix} \text{CAT} & \boxed{3}\,\text{v} \\ \text{AGR} & \boxed{4} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{SPEC} & + \end{bmatrix} \end{bmatrix}$$