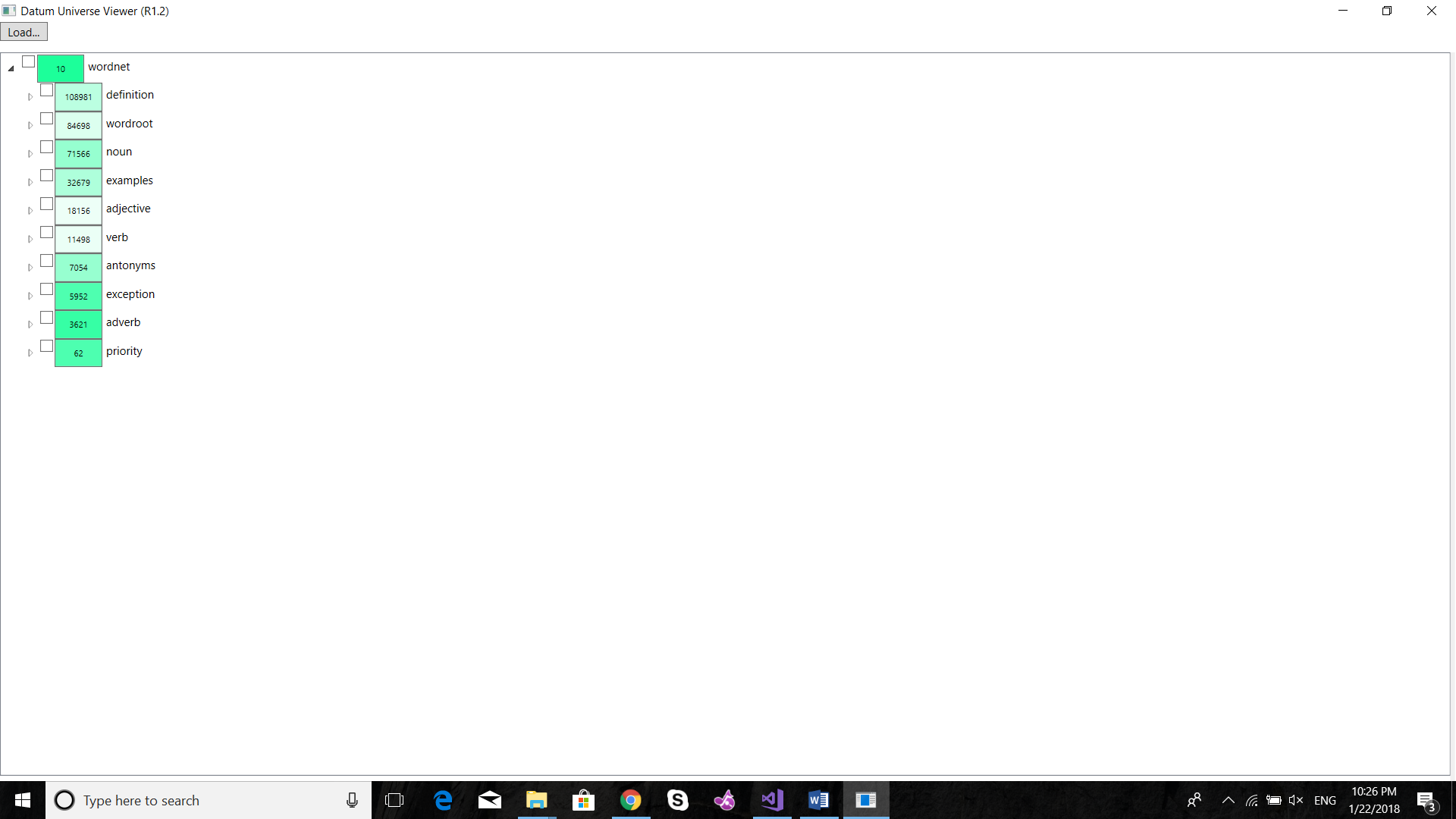
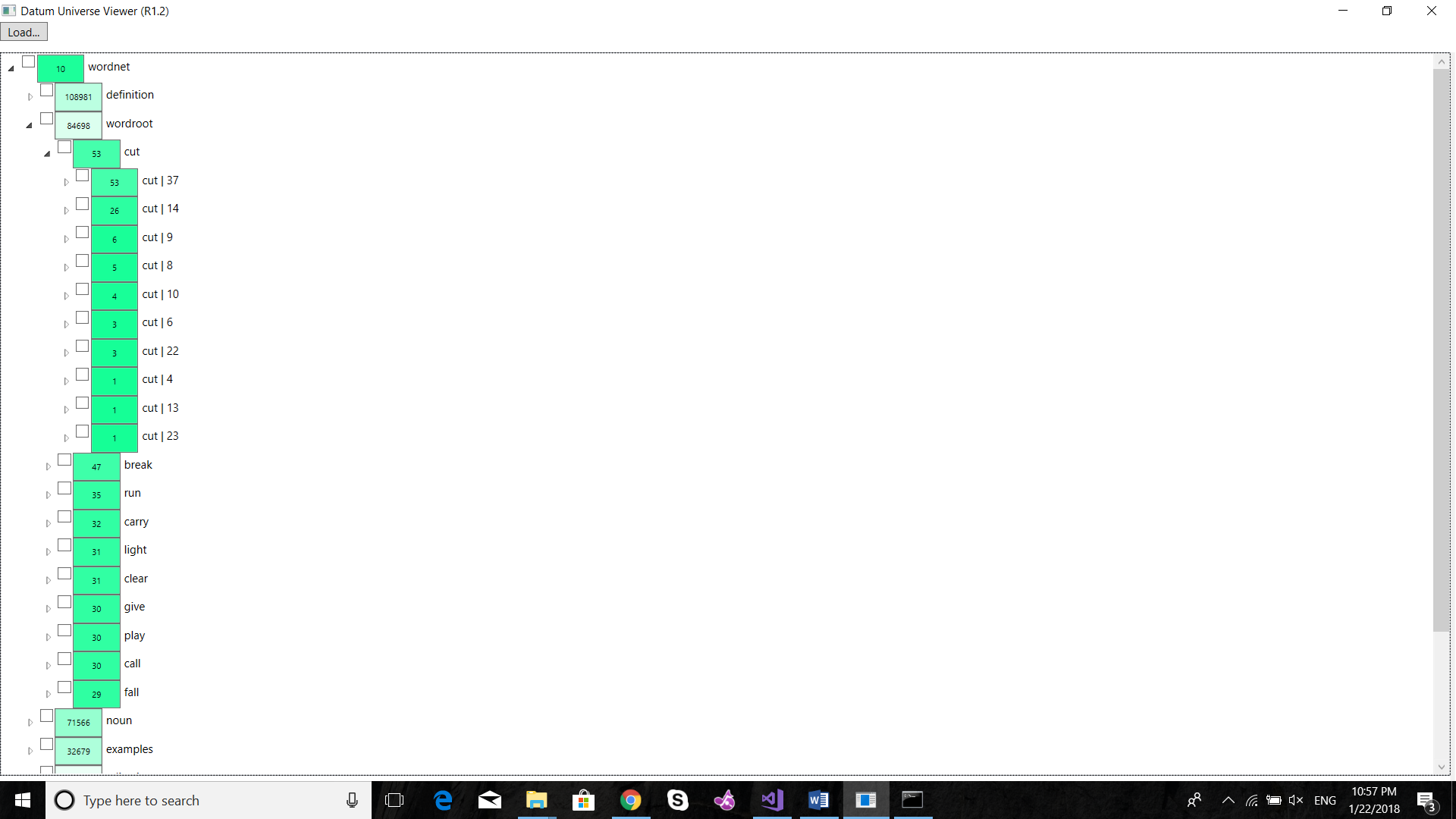
**Wordnet to Datum universe design description**

Wordnet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations.

The datum file “wordnet.datum” is a datum universe designed for wordnet using python.If you open the file using DUV, you should find the 10 instances under wordnet katum arranged as following:



In English, each word can have many meanings, that’s why in wordnet for each word there can be many different synsets, and the way these synsets are distinguished is using something called priority. For example : the synset ‘dog.n.01’ refers to the animal dog , but the synset ‘dog.n.03’ refers to an informal term for a man. In the datum universe we have all the words as instances for the katum wordroot and for each word , we have a numbered instance representing its priority which is the number sub-string part of the synset in wordnet, like in the synset ‘dog.n.01’ and ‘dog.n.03’ , have priorities 01 and 03 respectively , so in the datum universe we would have the numbers 1 and 3 as instances to the katum ‘dog’ and each have its own instances and attributes based on its representation in wordnet.

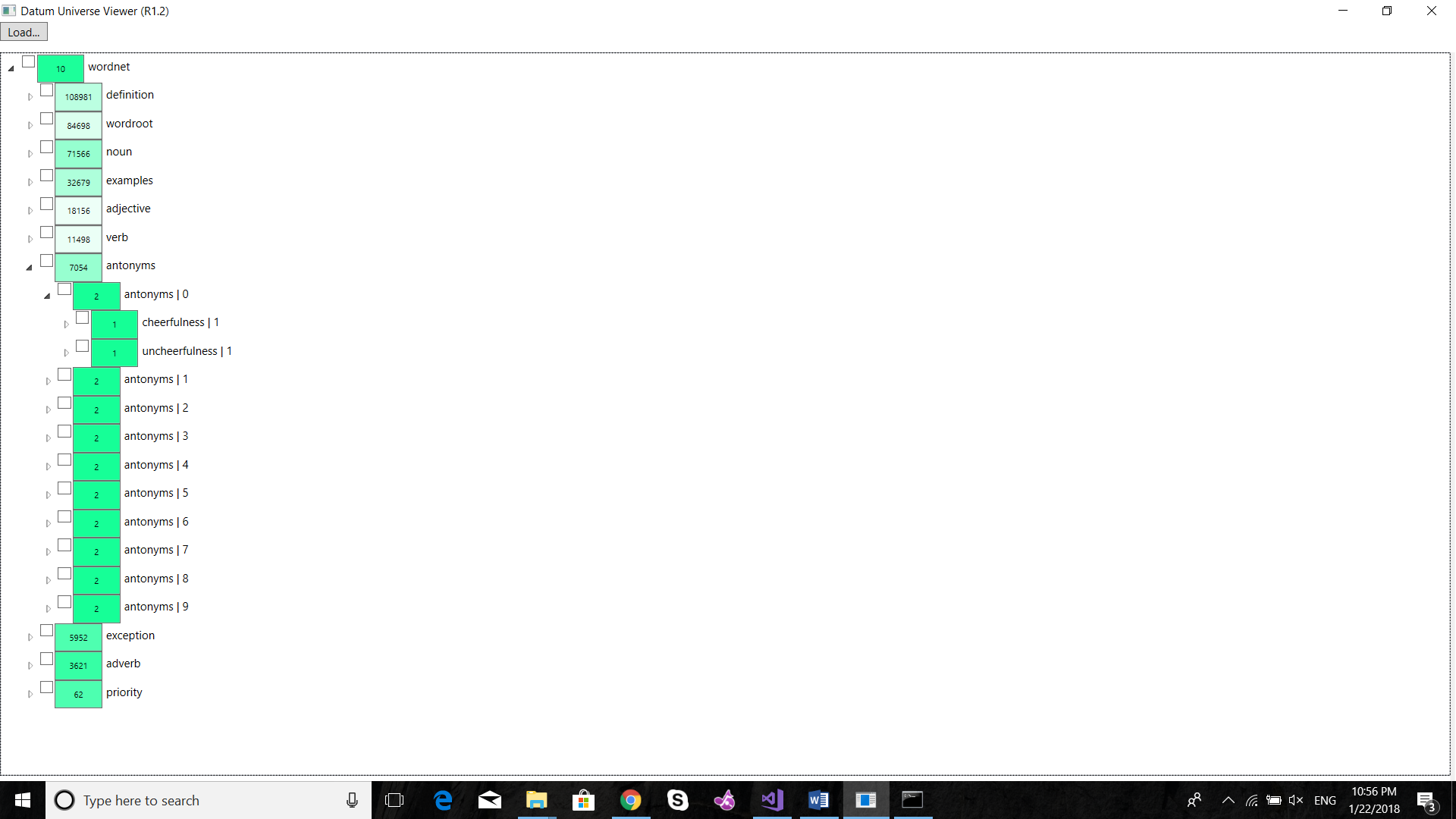


In wordnet synsets can have three relations: hypernyms (attributes), hypnonyms(instances), and antonyms.

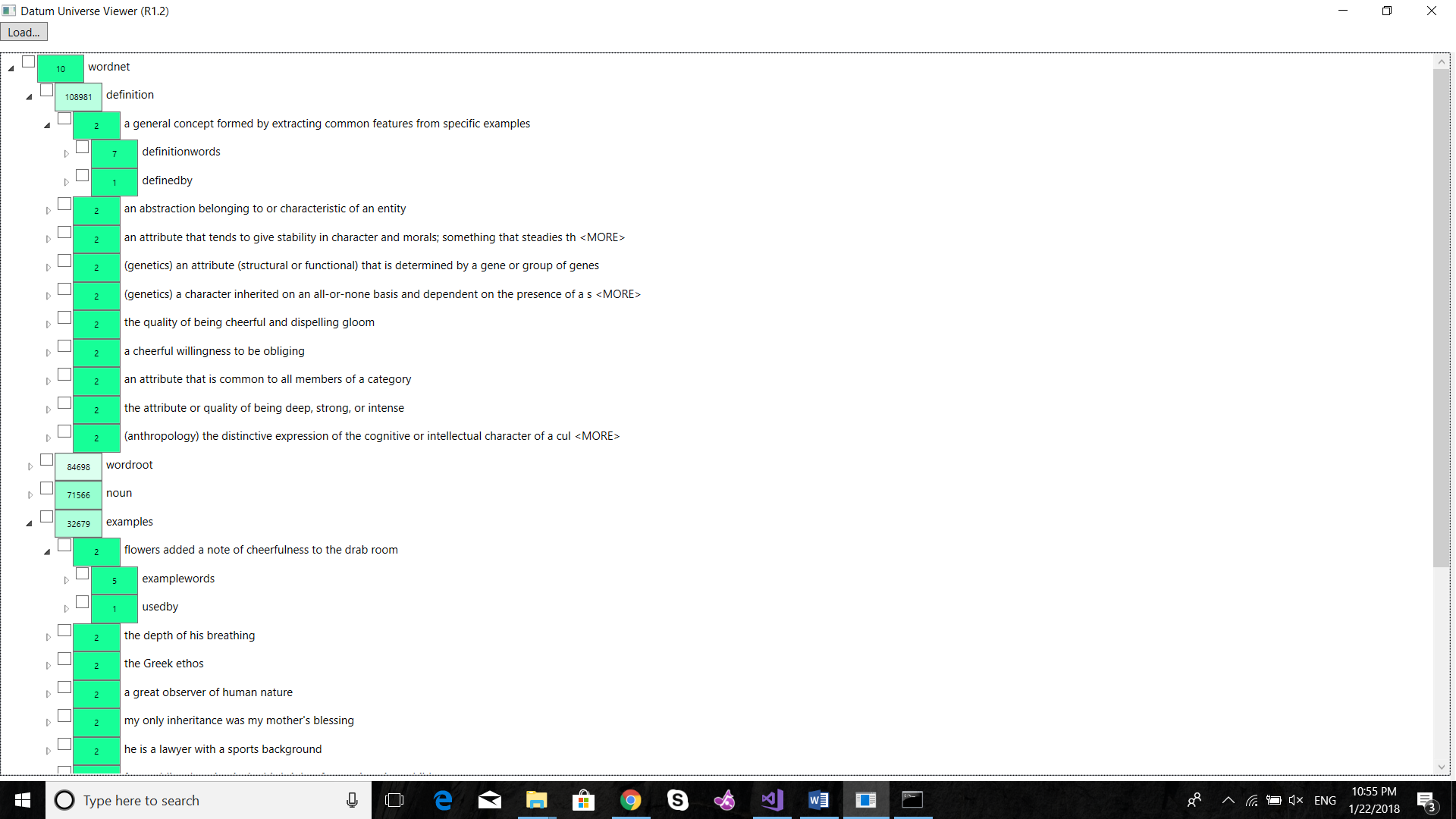
**Hypernyms**: are more general meanings of the word, like: domestic\_animal and canine are the hypernyms of dog. This relationship is represented in the form of attributes. Instance 1 of the katum dog has the katums domestic\_animal and canine as attributes.

**Hyponyms**: are more specific meanings of the word, like: corgi and dalmatian are hyponyms of dog. This relationship is represented if the form of instances. Instance 1 of the katum dog has the katums corgi and dalmatian as instances.

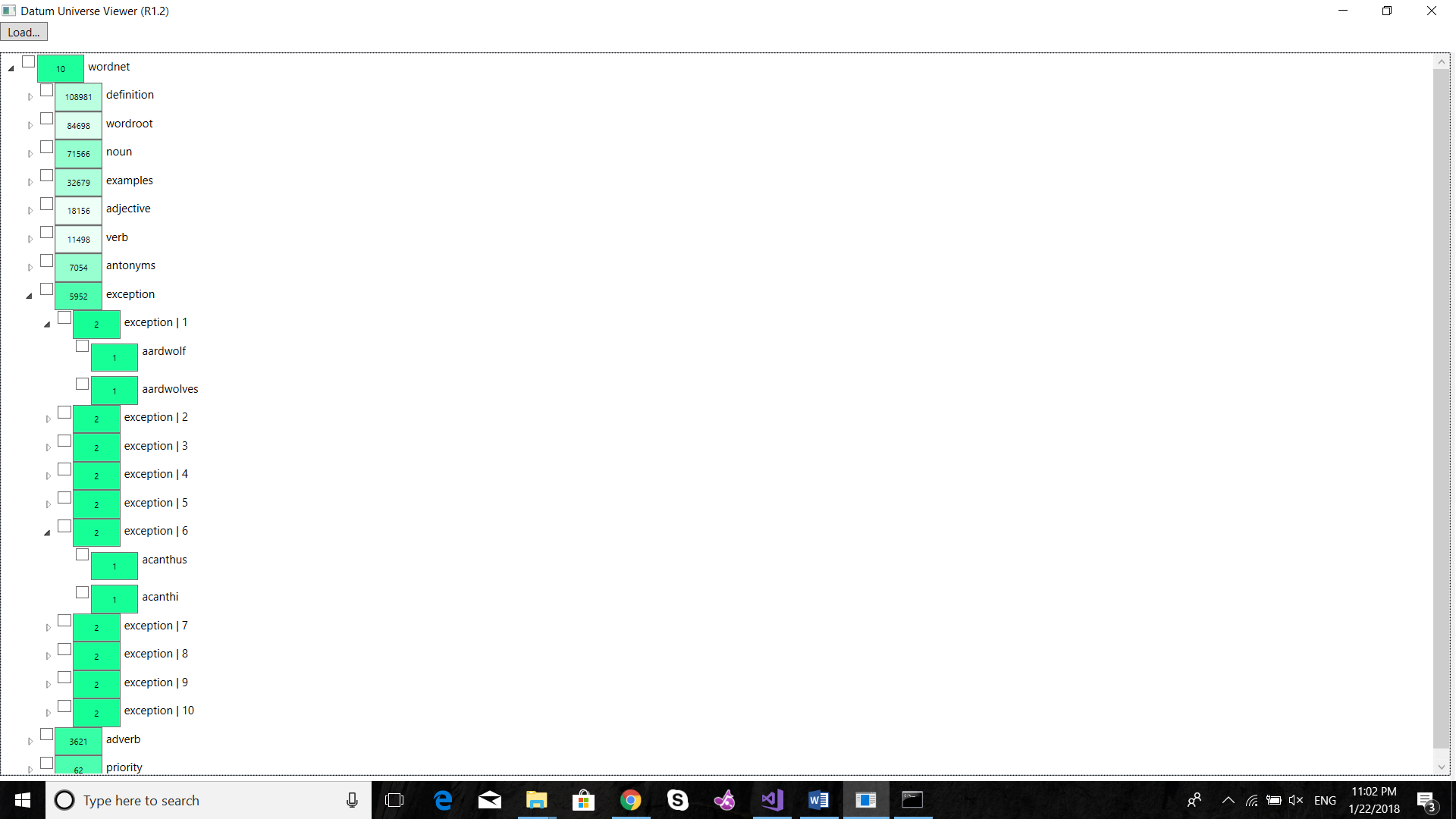
**Antonyms**: is the opposite of the meaning of the word, like: cheerfulness and uncheerfulness are antonyms to each other. In wordnet we have the katum antonyms and it has numbered katums as instances for each antonyms (2 words), the instances for each numbered katum are the numbered instance(sense/priority) of the two opposite words. Like: instance 1 of the katum cheerfulness and instance 1 of the katum uncheerfulness are both instances of a numbered katum which is an instance of antonyms



**Definition and example**: Each sense (priority) of each word in wordnet can have a definition and an example. However a definition can be used to define one word only, but an example can be used by many words. A definition/example is a string that consists of more than one word to help to understand the meaning and how to use a sense (priority) of a word. In the datum universe there are katums definition and examples , and for each definition/example there are two katums definition words/ example words and definedby/usedby, the first katum definition words/example words breaks down the definition/example into a set of words and connects these words to the words instances of the katum wordroot (if exists) to connect the definitions/examples to the words that are being used to describe a specific word. Definedby/usedby is used to refer to which word(s) has this definition/example.



**Exception**: in wordnet, words exist in the most absolute form, however some words can have an exceptional plural which is not just the word + s ,like: acanthi , its plural is acanthus. for this we have a numbered instance for the katum exception which has two instances: the word and its special form



**Noun/verb/adverb/adjective**: In wordnet, each words exist in these 4 forms: noun, verb,adverb,adjective.

Synsets exist in the form ‘just.a.01’ the ’a’ here means its an adjective, and in ‘just.r.03’ the ‘r’ means its an adverb , in ‘change.n.01’ the ‘n’ means it’s a noun and in ‘change.v.02’ the ‘v’ means it’s a verb. In the datum universe each numbered instance of a word in wordroot has one of these 4 katums(noun,verb,adverd,adjective) as an attribute based on its type. And this katum is used to define this sense of the word. So each the adjective and the adverb forms of ‘just’ has their own numbered instances of ‘just’ and each of them represent these two different senses of the word(they have different attributes, instances, definitions, examples …etc)

**Lemma**: In wordnet, some words doesn’t really exist in the tree hierarchy as there are some other words which have exactly the same meaning. Like: pupil and student, where there is a synset called “student.n.01” but pupil has exactly the same meaning so to find it you have to search for lemmas. Lemmas of student has pupil.

In the datum universe to represent this relationship, we have a katum called lemmas which has all such words as pupil as instances and these words has its lemmas as instances. So if you want to get the list of lemmas of student in C# you can easily write lemma.sof(student) which will return a list of two words : pupil – educate.

**Has:** In wordnet, There are two relations between words that explains components, called meronym and holonym. Meronym means parts , like school’s meronyms is classroom, holonym means part of , like classroom’s holonym is school, which means that holonym is the opposite if meronym.

In the datum universe to represent these relationships, we have a katum called has. The instances of has are those words that are not part of anything, their instances are their parts, like body is an instance of has, leg, arm, head, neck…etc are instances of body, and ankle, foot, crus ..etc are instances of leg.

To use the advantages of Ix and Ax. We decided to make all the instances under has native instances. Like to add the instance body under has . we created a native instance to has which has the name body + a number and then we added a connection between this new instance and the instance body using is(non native instance). In the same way, to add leg+ a number is a native instance to body + a number and a non native instance to leg.