ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ

Διπλωματική Εργασία

Εξόρυξη Δεδομένων της Νέας Ελληνικής Γλώσσας και οντολογική δόμησή τους στην πλατφόρμα του Protégé

Ιωάννης Κορωναίος

Αρ. Μητρώου: 71446794

Επιβλέπων Καθηγητής:
Ευάγγελος Παπακίτσος

ΑΙΓΑΛΕΩ 2024

UNIVERSITY OF WEST ATTICA SCHOOL OF ENGINEERING

DEPARTMENT OF INDUSTRIAL DESIGN AND PRODUCTION ENGINEERING

Diploma Thesis

Data retrieval for Modern Greek Language and their ontological structuring in PROTÉGÉ platform

Student name and surname:

Ioannis Koronaios

Registration Number:

71446794

Supervisor name and surname:

Evangelos Papakitsos

Athens, 2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ
ΑΤΤΙΚΗΣ ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ

Τίτλος εργασίας

Εξόρυξη Δεδομένων της Νέας Ελληνικής Γλώσσας και οντολογική δόμησή τους στην πλατφόρμα του Protégé

Μέλη Εξεταστικής Επιτροπής συμπεριλαμβανομένου και του Εισηγητή

Η διπλωματική εργασία εξετάστηκε επιτυχώς από την κάτωθι Εξεταστική Επιτροπή:

Α/α	ΟΝΟΜΑ ΕΠΩΝΥΜΟ	ΒΑΘΜΙΔΑ/ΙΔΙΟΤΗΤΑ	ΨΗΦΙΑΚΗ ΥΠΟΓΡΑΦΗ
1	Ε.Χ. ΠΑΠΑΚΙΤΣΟΣ	ΕΔΙΠ Α΄	Evangelos Papakitsos Digitally signed by Evangelos Papakitsos Date: 2024.10.24 20:59:09 +03'00'
2	Ν. ΛΑΣΚΑΡΗΣ	ΕΠΙΚΟΥΡΟΣ ΚΑΘΗΓΗΤΗΣ	Nikolaos Laskaris Digitally signed by Nikolaos Laskaris Date: 2024.10.25 12:31:38 +03'00'
3	Χ. ΔΡΟΣΟΣ	ΕΠΙΚΟΥΡΟΣ ΚΑΘΗΓΗΤΗΣ	Digitally signed by CHRISTOS DROSOS Date: 2024.10.25 15:05:20 +03'00'

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ/ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Κορωναίος Ιωάννης του Ιορδάνη, με αριθμό μητρώου 71446794 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Βιομηχανικής Σχεδίασης και Παραγωγής, δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της πτυχιακής/διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.
Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο Δηλών

Κορωναίος
Ιωάννης

ΕΥΧΑΡΙΣΤΙΕΣ

Ευχαριστώ πολύ τον επιβλέποντα καθηγητή μου για την εμπιστοσύνη και τη στήριξη που μου προσέφερε κατά την εκπόνηση της παρούσας διπλωματικής εργασίας. Επιπλέον, θεωρώ χρέος μου να ευχαριστήσω την υποψήφια διδάκτορα του Πανεπιστημίου Δυτικής Αττικής κα Νικολέττα Σαμαρείδη, η οποία μου εμπιστεύθηκε μέρος της υψηλού επιπέδου διδακτορικής της διατριβής. Τέλος, θα ήθελα να ευχαριστήσω την σύζυγό μου, τους γονείς μου και την υπόλοιπη οικογένειά μου χωρίς την βοήθεια και την υπομονή της οποίας δεν θα είχα βρει τον απαιτούμενο χρόνο για να φέρω εις πέρας την ανά χείρας εργασία.
Για την βοήθεια όλων των παραπάνω προσώπων είμαι βαθύτατα ευγνώμων.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΛΗΨΗ

Η παρούσα Διπλωματική Εργασία εξετάζει την δυνατότητα άντλησης γλωσσικών δεδομένων της Νέας Ελληνικής Γλώσσας από μια δομημένη πηγή στο διαδίκτυο και την φόρτωση αυτών σε μια οντολογία στην πλατφόρμα Protégé. Ως πηγή χρησιμοποιήθηκε το «Λεξικό της Κοινής Νέας Ελληνικής» του Μανώλη Τριανταφυλλίδη, το οποίο είναι διαθέσιμο σε ηλεκτρονική μορφή στο διαδίκτυο. Η εφαρμογή μας έχει σαν στόχο να συγκεντρώσει όλα τα λήμματα που περιέχονται στο παραπάνω λεξικό σε ένα αρχείο τύπου csv (comma – separated values) και στην συνέχεια, να τα μεταφορτώσει από το αρχείο αυτό σε μια Οντολογία που να είναι συμβατή με την πλατφόρμα Protégé. Εντός της οντολογίας, τα λήμματα θα πρέπει να οργανωθούν σε κλάσεις, ανάλογα με το Μέρος του Λόγου στο οποίο ανήκουν. Επιπλέον, για κάθε λήμμα πρέπει είναι διακριτά τα εκάστοτε επιμέρους χαρακτηριστικά του.

Η εφαρμογή μας αναπτύσσεται σε τρία scripts, για λόγους ευκολίας στην διαχείριση του όγκου των δεδομένων. Συνεπώς, έχουμε:

1. **Άντληση λημμάτων σε ένα αρχείο .csv**
   Η άντληση των λημμάτων πρέπει να γίνει με τέτοιο τρόπο, ώστε να είναι διακριτό το Μέρος του Λόγου στο οποίο ανήκουν, αλλά ταυτόχρονα να καταχωρούνται ξεχωριστά και διάφορα επιμέρους χαρακτηριστικά τους όπως συνώνυμες ή αντώνυμες λέξεις, ετυμολογία, προφορά κ.ά.

2. **Δημιουργία της Οντολογίας**
   Μέσω κώδικα θα πρέπει να δημιουργήσουμε μια Οντολογία σε πρότυπο owl, το οποίο μπορεί να αναγνωστεί από το Protégé. Η οντολογία μας, θα περιλαμβάνει μια κύρια κλάση, την «Μέρη_του_Λόγου/Parts_of_Speech» και στη συνέχεια υποκλάσεις σε διάφορα επίπεδα, αναπαριστώντας τα Μέρη του Λόγου όπως περιέχονται στην Γραμματική της Νέας Ελληνικής, καθώς και τις μεταξύ τους σχέσεις. Επίσης, θα δημιουργηθούν κλάσεις που θα απεικονίζουν τα επιμέρους χαρακτηριστικά κάθε λήμματος, όπως αυτά αναφέρονται παραπάνω. Οι κλάσεις αυτές θα είναι μέρος των Data Properties και των Annotation Properties της οντολογίας.

3. **Μεταφόρτωση από το αρχείο .csv στην Οντολογία**
   Το τρίτο script της εφαρμογής μας θα αντλεί από το csv τα λήμματα οργανωμένα και τα χαρακτηριστικά αυτών και θα τα φορτώνει στην Οντολογία του προηγούμενου βήματος, φροντίζοντας για την ορθή καταχώρησή τους στις υπάρχουσες κλάσεις. Ο χωρισμός των γλωσσικών όρων και η ταξινόμηση των επιμέρους ιδιοτήτων τους θα γίνει βάσει των καταχωρημένων στο csv στοιχείων.

Πρέπει να αναφερθεί πως η παρούσα εργασία αποτελεί ένα μέρος της υποστήριξης της διδακτορικής διατριβής της κας Νικολέττας Σαμαρείδη.

Τα scripts που αναπτύχθηκαν είναι τα *1.LexikoOrganosi.py*, *2.Dimioyrgia_Ontologias.py* και *3.Prosthiki_Individuals*, τα οποία και παρατίθενται στο παράρτημα της εργασίας.

Λέξεις-κλειδιά: εξόρυξη δεδομένων, Python, οντολογία, OWL, Protégé, Νέα Ελληνική Γλώσσα, γλωσσική τεχνολογία.

ABSTRACT

The present Diploma Thesis examines the possibility of extracting linguistic data of the Modern Greek Language from a structured online source and loading them into an ontology on the Protégé platform. The source used was the "Dictionary of Standard Modern Greek" by Manolis Triantafyllidis, which is available in electronic form on the internet. Our application aims to collect all the entries contained in the above dictionary into a CSV (comma-separated values) file, and then upload them from this file into an ontology that is compatible with the Protégé platform. Within the ontology, the entries must be organized into classes, depending on the Part of Speech to which they belong. Additionally, each entry's individual characteristics will also be clearly distinguished. For the sake of simplicity, we deemed it necessary for the application to consist of three (3) scripts, making the steps clearer both during the implementation and the presentation of the work. Therefore, the work is developed in the following three applications:

Our application is developed in three scripts for easier management of the large volume of data. Thus, we have:

1. **Extracting entries into a .csv file**
   The extraction of the entries must be done in such a way that the Part of Speech to which they belong is clear, while at the same time, various individual characteristics, such as synonyms or antonyms, etymology, pronunciation, etc., are recorded separately.

2. **Creation of the Ontology**
   Through code, we need to create an ontology in the OWL format, which can be recognized by Protégé. Our ontology will include a main class, PartsOfSpeech, followed by subclasses at various levels, representing the Parts of Speech as described in the Grammar of Modern Greek, as well as the relationships between them. In addition, classes will be created to represent the individual characteristics of each entry, as mentioned above. These classes will be part of the Data Properties and Annotation Properties of the ontology.

3. **Uploading from the .csv file into the Ontology**
   The third script of our application will extract the organized entries and their characteristics from the CSV and load them into the ontology created in the previous step, ensuring their proper categorization into the existing classes. The division of linguistic terms and the classification of their individual properties will be done based on the information stored in the CSV.

It should be noted that this work is part of the support for the doctoral dissertation of Mrs. Nikoletta Samaridi.

The scripts that were developed are: 1. LexikoOrganosi.py, 2. Dimioyrgia_Ontologias.py, and 3. Prosthiki_Individuals, which are provided in the appendix of this thesis.

Keywords: data mining, Python, ontology, OWL, Protégé, Modern Greek Language, language technology.

1. ΕΙΣΑΓΩΓΗ

1.1 Εξόρυξη Δεδομένων – Data Mining

Η παρούσα Διπλωματική Εργασία κινείται στο γνωστικό χώρο της Εξόρυξης Δεδομένων (Data Mining). Ως Data Mining ορίζεται «η διαδικασία εξόρυξης πληροφοριών μέσα από την ανάλυση μεγάλου όγκου δεδομένων, αποσκοπώντας στην αναγνώριση τάσεων και προτύπων» [1]. Πιο απλά, είναι άντληση ενός μικρού ή μεγάλου όγκου δεδομένων και η αξιοποίησή του μέσα από διαδικασίες και αλγορίθμους προκειμένου να εξάγουμε ένα συμπέρασμα για την συμπεριφορά ενός συστήματος ή μιας ομάδας όντων. Τις περισσότερες φορές η Εξόρυξη δεδομένων αποτελείται από 5 βήματα, την **Συλλογή Δεδομένων (Data Collection)**, τον **Καθαρισμό Δεδομένων (Data Cleaning)**, τον **Μετασχηματισμό Δεδομένων (Data Transformation)**, την **Εξόρυξη Δεδομένων** (**Data Mining**) και την **Ανάλυση και ερμηνεία Δεδομένων**, τα οποία και κρίνουμε απαραίτητο να αναλυθούν παρακάτω. Η εφαρμογή που αναπτύξαμε αφορά στα πρώτα τρία βήματα [2].

Η **Συλλογή Δεδομένων** είναι η εύρεση των διαθέσιμων και καταλληλότερων πηγών για τα δεδομένα που ψάχνουμε και η άντληση αυτών σε μία δομή (π.χ. ένα αρχείο, μια νέα βάση δεδομένων κ.ά.) [1,2]. Ορισμένες φορές, τα αρχικά στοιχεία τα οποία συλλέγουμε μπορεί να περιέχουν κάποια περιττή στο στόχο μας πληροφορία ή κάποιο μέρος τους να είναι λάθος, ή να περιέχεται ένα στοιχείο, ένας όρος διπλός. Τα παραπάνω έρχονται να εξαλειφθούν με το **Data Cleaning**, το δεύτερο στάδιο της διαδικασίας [1,3]. Επίσης, για να αξιοποιήσουμε για τον σκοπό μιας εφαρμογής τα δεδομένα που συλλέξαμε, συχνά απαιτείται ένας **Μετασχηματισμός (Data Transformation)**, δηλαδή να πάρουν τα στοιχεία μας μια άλλη μορφή που να επιτρέπει να αναλυθούν, αναλόγως με τον αλγόριθμο που ακολουθείται αμέσως μετά στην **Εξόρυξη δεδομένων**, η οποία σαν έξοδο έχει ένα συμπέρασμα («ανάγνωση προτύπων και τάσεων») [3]. Η διαδικασία θα μπορούσε να τελειώνει εδώ, αλλά σαν τελικό στάδιο αναφέρεται συχνά και η **Ανάλυση και Ερμηνεία** του συμπεράσματος της εξόρυξης, προκειμένου να ληφθούν αν είναι απαραίτητο οι κατάλληλες ενέργειες (π.χ. σε ένα παράδειγμα με μια ομάδα καταναλωτών, να αποφασίσει η επιχείρηση ποιο προϊόν πρέπει να προωθήσει σε αυτούς ή ποιο πρέπει να αποσύρει κλπ.) [1,2,4].

1.2 Σημασιολογικός Ιστός και Οντολογίες

Ο **Σημασιολογικός Ιστός** σαν όρος εισήλθε από τον sir Tim John Berners-Lee, τον μηχανικό που εφηύρε τον Παγκόσμιο Ιστό. Μετά από την δημιουργία του Παγκόσμιου Ιστού, ο Tim Berners-Lee οραματιζόταν την βελτίωση της επικοινωνίας μεταξύ ανθρώπου – μηχανής. Αργότερα, αυτό άρχιζε να υλοποιείται με την ανάπτυξη δομών με πληροφορία εύκολα κατανοητή από μηχανές, τις λεγόμενες «Οντολογίες».

Από το 1993, όπου έκαναν την πρώτη τους εμφάνιση, μέχρι και σήμερα έχουν δοθεί διάφοροι ορισμοί για το τι αποτελούν οι **Οντολογίες** στον Σημασιολογικό Ιστό. Παραθέτουμε τις τρεις που θεωρούμε πιο αντιπροσωπευτικές.

Κατά τον Tom Gruber «*Οντολογία είναι μια ρητή προδιαγραφή μιας εννοιολογικής υπόστασης*», δηλαδή «*πρόκειται για μια αναπαράσταση της γνώσης που περιλαμβάνει οντότητες, έννοιες και τις μεταξύ τους σχέσεις*» [8]. Επιπλέον, οι Michael Uschold και Martin King λίγο αργότερα όρισαν την οντολογία ως «*μια κατηγοριοποίηση, δηλαδή ένας ξεκάθαρος ορισμός των βασικών εννοιών και των σχέσεων μεταξύ τους σε έναν συγκεκριμένο τομέα*» [10]. Τέλος, οι Genevieve Bell και Paul Dourish το 2007 έδωσαν τον ορισμό ότι «*Μια οντολογία στον Σημασιολογικό Ιστό είναι μια κοινή κατανόηση ενός τομέα που μπορεί να χρησιμοποιηθεί για να περιγράψει δεδομένα και να διευκολύνει την διαλειτουργικότητα μεταξύ συστημάτων*» [9], ο οποίος αποτελεί και τον επικρατέστερο σήμερα [7]. Με πιο απλά λόγια, μια Οντολογία αποτελεί μια οργανωμένη δομή δεδομένων, κατηγοριοποιημένα σε σύνολα και υποσύνολα (κλάσεις και υποκλάσεις) με σαφή ορισμό των μεταξύ τους σχέσεων και ενίοτε και ορισμένες έννοιες που χαρακτηρίζουν τα δεδομένα αυτά.

Η δομή μιας Οντολογίας στον Σημασιολογικό ιστό είναι σχετικά απλή. Τα δεδομένα που περιέχονται στην οντολογία χωρίζονται σε 2 είδη, τα individuals και στα Properties. Τα individuals αποτελούν τα κύρια στοιχεία της Οντολογίας. Για παράδειγμα, έστω ότι δημιουργούμε μια οντολογία με σκοπό να αναπαραστήσουμε την παγκόσμια βιβλιογραφία. Σε αυτήν την περίπτωση, κάθε βιβλίο θα θεωρείται ως individual ενώ σαν properties θα μπορούν να ορίζονται ο τίτλος του, ο συγγραφέας του, η τιμή του στην αγορά, ο κωδικός ISBN του κ.ο.κ. Στους περισσότερους τύπους οντολογιών, τα properties χωρίζονται σε Data και Annotation properties [10, 12].

Τα **Data Properties** συνδέουν τα individuals με άλλα δεδομένα ή τιμές δεδομένων [7]. Στο παραπάνω παράδειγμα, η τιμή του κάθε βιβλίου ή ο συγγραφέας του θα εντάσσονταν στα Data Properties της οντολογίας. Αντίθετα, τα **Annotation Properties** χρησιμοποιούνται για να δώσουν έναν πιο εκτενή σχολιασμό για τα individuals ή τις κλάσεις μιας οντολογίας, αλλά δεν προσφέρουν στους λογικούς υπολογισμούς ή την συλλογιστική της. Περιέχουν περισσότερο εξηγήσεις για την καλύτερη τεκμηρίωση των στοιχείων της [7, 12]. Έτσι, στο παράδειγμα με τα βιβλία, στα Annotation properties θα ανήκουν ο κωδικός ISBN ή ο τίτλος κάθε βιβλίου που χρησιμεύουν μόνο ως αναγνωριστικά για το βιβλίο.

Τέλος, μια οντολογία περιέχει υποχρεωτικά κλάσεις και υποκλάσεις ώστε να γίνονται πιο σαφείς οι σχέσεις μεταξύ των Individuals. Σε κάθε οντολογία υπάρχει η Κύρια κλάση, από την οποία προκύπτουν άλλες κλάσεις και υποκλάσεις αυτών. Με τον τρόπο αυτόν χωρίζονται τα άτομα της οντολογίας σε επίπεδα και κατηγορίες. Αν παραμείνουμε στο παράδειγμα με τα βιβλία, οι κλάσεις της Οντολογίας θα μπορούσαν να είναι οι κατηγορίες βιβλίων που κυκλοφορούν, π.χ. κλάση: **Scientific**, **Literature**, **History**, κ.ά. Υποκλάσεις αυτών θα μπορούσαν να είναι οι **Chemistry(Scientific)**, **Physics(Scientific)**, **Linguistics(Scientific), Medieval-Age(History), Modern-Times(History)** κ.ο.κ. [12].

Από την πρώτη τους εμφάνιση στον Σημασιολογικό Ιστό, οι Οντολογίες έπρεπε κάπως να οριστούν ώστε να μπορούν να διαβαστούν από υπολογιστικές μηχανές. Έτσι, δημιουργήθηκαν διάφορα πρότυπα, δηλαδή τύποι αρχείων που τις αναπαριστούν με τρόπο που μπορεί να καταλάβει ένας ηλεκτρονικός

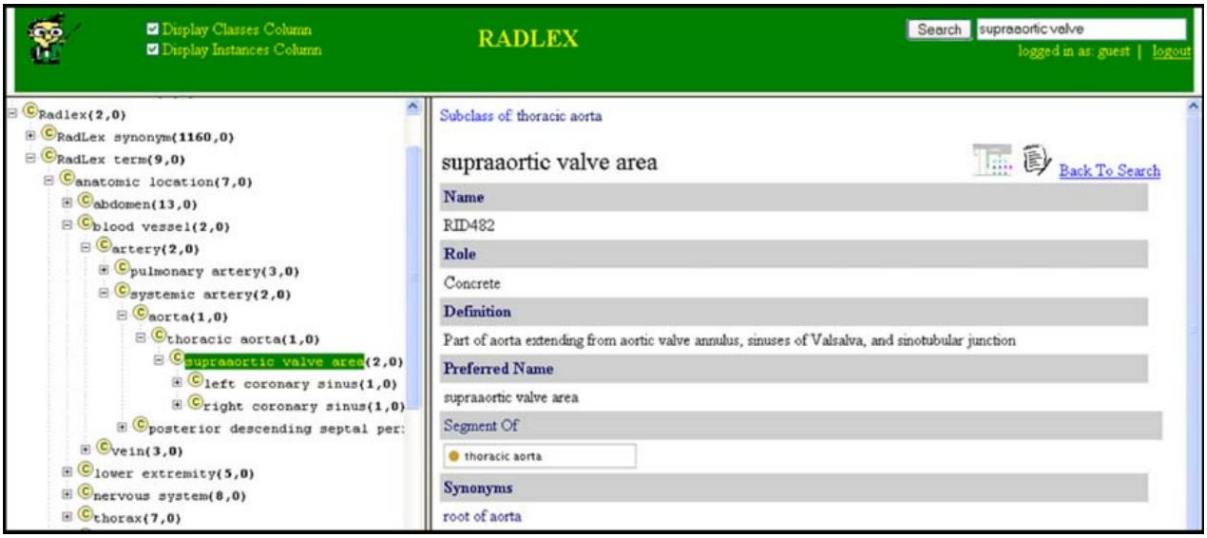
υπολογιστής. Τα πρότυπα που έχουν δημιουργηθεί, και κάποια εξ αυτών χρησιμοποιούνται μέχρι και σήμερα, είναι τα:

a. **RDF (Resource Description Framework) – 1999**
   Το πρώτο πρότυπο που αναπτύχθηκε για αναπαράσταση δεδομένων στον Σημασιολογικό Ιστό. Αρχικά αναπαριστούσε τριπλέτες όρων στη μορφή «υποκείμενο, ρήμα, αντικείμενο». Μέχρι και σήμερα αποτελεί την βάση για την αναπαράσταση των σχέσεων των όρων μιας οντολογίας σε όλα τα επόμενα πρότυπα [11].
b. **DCMI (Dublin Core Metadata Initiative) – 2000** [11].
c. **RDF Schema (RDFs) – 2000**
   Επέκταση του αρχικού προτύπου RDF. Αναπτύχθηκε μόλις ένα έτος μετά από αυτό και παρείχε για πρώτη φορά μια γλώσσα για την περιγραφή της σχέσης μεταξύ των κλάσεων μιας οντολογίας και της ταξινόμησης των όρων εντός αυτής. Αποτέλεσε το πρώτο βήμα προς τη δημιουργία ιεραρχικών δομών για δεδομένα [11].
d. **OBO (Open Biomedical Ontologies) – 2000** [11].
e. **OWL (Web Ontology Language) – 2004**
   Το πλέον χρησιμοποιούμενο σήμερα πρότυπο για την αναπαράσταση ποικίλων ειδών οντολογιών. Αναγνωρίζεται από την πλατφόρμα Protégé και επιτρέπει την αναπαράσταση γνώσης με δυνατότητα παραγωγής συμπερασμάτων σύμφωνα με τα δεδομένα και τις σχέσεις που αναπτύσσονται στην οντολογία [12,13].
f. **BFO (Basic Formal Language) – 2006** [11].
g. **SPARQL (SPARQL Protocol and RDF Query Language) – 2008** [11].
h. **SKOS (Simple Knowledge Organization System) -2009** [11].
i. **FIBO (Financial Industry Business Ontology) – 2014** [11].

Αναλύσαμε τα τρία πρότυπα τα οποία χρειάστηκε να μάθουμε για την ανάπτυξη της οντολογίας μας στην παρούσα διπλωματική εργασία. Το πρότυπο το οποίο υιοθετήσαμε είναι το πρότυπο OWL το οποίο είδαμε ως πρώτο προτεινόμενο στην πλατφόρμα Protégé.

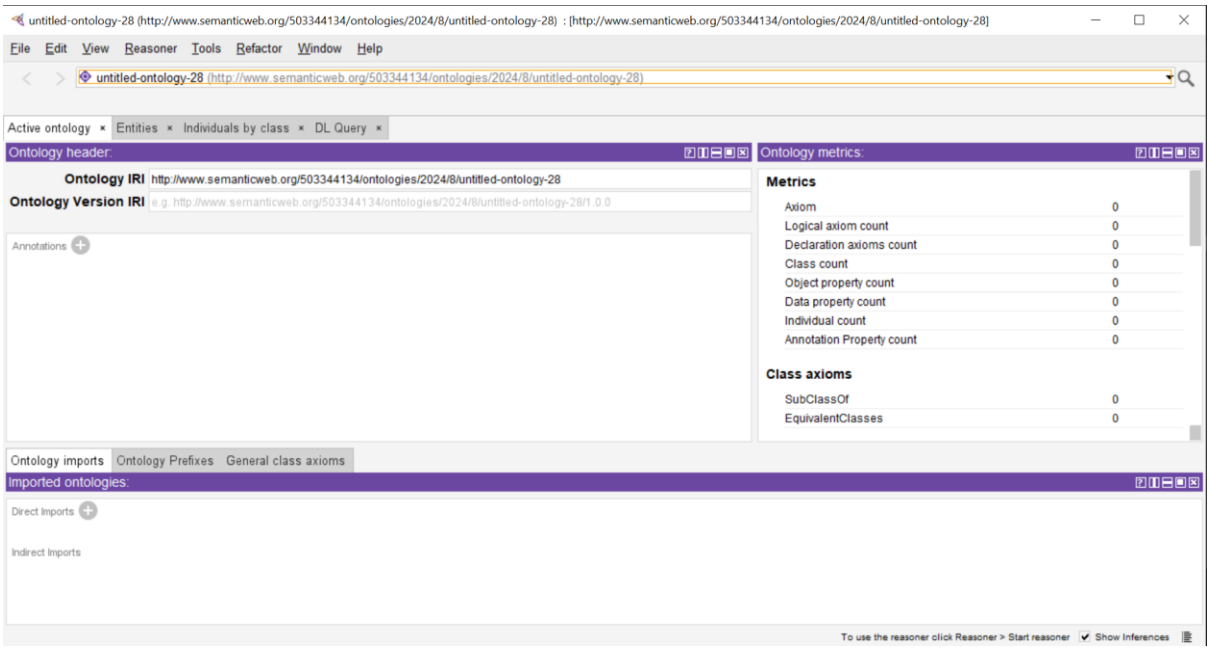
1.3 Πλατφόρμα Protégé

Στις αρχές της δεκαετίας του 1980 αναπτύχθηκε στο Πανεπιστήμιο του Stanford η πλατφόρμα Protégé. Αρχικά, η πλατφόρμα χρησιμοποιήθηκε στο πλαίσιο έρευνας στον τομέα της Βιοϊατρικής Πληροφορικής (Biomedical Informatics). Τα δεδομένα που διαχειρίζονταν ήταν καθαρά ιατρικά. Από την δεκαετία του 1990, το Protégé άρχισε να κερδίζει αναγνωρισιμότητα σε διάφορες εφαρμογές και εκτός της ιατρικής με αποκορύφωμα τη χρήση του σε συστήματα του Σημασιολογικού Ιστού και του διαδικτύου. Το 2004, η πλατφόρμα διαμορφώθηκε περαιτέρω ώστε να μπορεί να υποστηρίξει οντολογίες σε πρότυπο OWL (Web Ontology Language), κάτι που την κατέστησε το ιδανικό εργαλείο για κάθε εφαρμογή που αφορά τον χώρο της Τεχνητής Νοημοσύνης και της Μηχανικής Μάθησης.



Εικόνα 1: *Περιβάλλον Protégé την δεκαετία του 2000[14]*

Με την πάροδο των χρόνων το Protégé έχει εξελιχθεί στην πιο διαδεδομένη πλατφόρμα για ανάπτυξη οντολογιών, υποστηρίζοντας και άλλα μεταγενέστερα του OWL πρότυπα όπως το RDF, RDFs και το SPARQL.



Εικόνα 2*: Περιβάλλον του Protégé σήμερα*

Στην Ελλάδα, με την μελέτη και την ανάπτυξη Οντολογιών και με την χρήση του Protégé έχουν ασχοληθεί αρκετοί ακαδημαϊκοί από το 2005 και έπειτα. Ενδεικτικά αναφέρουμε στην βιβλιογραφία μας 3 δημοσιεύσεις [15, 16, 17].

2. ΛΕΙΤΟΥΡΓΙΚΕΣ ΠΡΟΔΙΑΓΡΑΦΕΣ

Στην παρούσα Διπλωματική εργασία σκοπός είναι να δημιουργηθεί μια οντολογία για την Ελληνική Γλώσσα, η οποία να περιέχει όλα τα λήμματά της, οργανωμένα βάσει του Μέρους του Λόγου στο οποίο ανήκουν, να περιέχει λεξιλογικές ιδιότητες και χαρακτηριστικά για τα λήμματα αυτά και να μπορεί να παρασταθεί στην οντολογική πλατφόρμα Protégé.

Για την επίτευξη αυτού του σκοπού, η εφαρμογή μας επιτελεί τρεις λειτουργίες, την **Άντληση Λημμάτων από μια διαδικτυακή πηγή σε ένα αρχείο τύπου .csv**, την **Δημιουργία μιας Οντολογίας σε πρότυπο owl**, το οποίο αναγνωρίζεται από το Protégé και την **Μεταφόρτωση των Λημμάτων από το .csv στην πλατφόρμα του Protégé**. Οι συγκεκριμένες τρεις φάσεις αναλύονται συνοπτικά παρακάτω.

2.1 Άντληση Λημμάτων από διαδικτυακή πηγή

Ως διαδικτυακή πηγή για τα λήμματα της Νέας Ελληνικής Γλώσσας επιλέχθηκε το «Λεξικό της Κοινής Νέας Ελληνικής Γλώσσας» του Μανώλη Τριανταφυλλίδη, το οποίο βρίσκεται διαθέσιμο σε ηλεκτρονική μορφή στο διαδίκτυο, στον ιστότοπο «Πύλη για την Ελληνική Γλώσσα» [23]. Χρησιμοποιώντας κώδικα σε γλώσσα προγραμματισμού Python, η εφαρμογή μας κάνει αιτήματα στον ιστότοπο και φορτώνει όλα τα λήμματα σε ένα αρχείο csv. Βάσει των κανόνων που περιέχονται στην διδακτορική διατριβή της κας Νικολέττας Σαμαρείδη [17] θα πρέπει για κάθε λήμμα να είναι σαφώς ορισμένες οι ιδιότητες *«Λέξη», «Λέξη με άρθρο», «Προφορά», «Ετυμολογία», «Μέρος του Λόγου», «Ορισμός – Επεξήγηση», «Αρσενικός τύπος», «Θηλυκός τύπος», «Αντώνυμο», «Συνώνυμο», «Συνώνυμη φράση», «Υποκοριστικό», «Μεγεθυντικό», «Φράση» (παράδειγμα χρήσης του λήμματος), «Παροιμία».* Συνεπώς, καθώς ο κώδικας αντλεί τα λήμματα, παράλληλα για κάθε ένα από αυτά οργανώνει τα επιμέρους χαρακτηριστικά του σε columns.

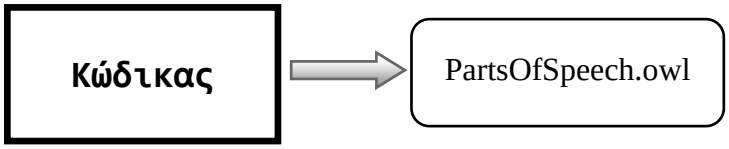
Στο τέλος αυτής της λειτουργίας προκύπτει ένα αρχείο τύπου .csv (LexikoOrganised.csv) το οποίο περιέχει 15 στήλες (columns), μία για κάθε χαρακτηριστικό που περιγράψαμε παραπάνω και 49.541 γραμμές (rows), όσες και τα λήμματα που περιέχονται στο ηλεκτρονικό Λεξικό.

Διάγραμμα 1: *Λειτουργία ανάκτησης λημμάτων από το διαδίκτυο*

2.2 Δημιουργία Οντολογίας σε πρότυπο owl

Η δεύτερη λειτουργία που επιτελείται στην εφαρμογή μας είναι εκείνη της δημιουργίας της Οντολογίας. Όπως αναφέρεται και στο κεφάλαιο της Εισαγωγής, στην Οντολογία μας πρέπει να ορίζονται μια κύρια κλάση με τις υποκλάσεις της και οι ιδιότητες που θα χαρακτηρίζουν κάθε individual.

Στο παράδειγμά μας, τα individuals της οντολογικής δομής που θα αναπτύξουμε θα αποτελούν τα λήμματα του «Λεξικού Τριανταφυλλίδη». Συμπερασματικά, η Κύρια Κλάση θα είναι η «Μέρη του Λόγου», η οποία θα αναλύεται σε ακόμη 3 επίπεδα υποκλάσεων. Ακόμη, ως ιδιότητες των individuals θα νοούνται τα χαρακτηριστικά εκείνα που θα υπάρχουν στις στήλες του αρχείου .csv του προηγούμενου βήματος, δηλαδή η «Λέξη», το «Συνώνυμο», το «Αντώνυμο» κ.ο.κ. Για την διάκριση των παραπάνω ιδιοτήτων σε Data Properties και Annotation Properties χρησιμοποιήθηκε ένα μέρος των κανόνων της διατριβής της κας Σαμαρείδη [17]. Με την ολοκλήρωση της λειτουργίας προκύπτει το αρχείο «PartsOfSpeech.owl» που περιέχει αποθηκευμένη την οντολογία.



Διάγραμμα 2: *Λειτουργία δημιουργίας της Οντολογίας owl*

2.3 Μεταφόρτωση των Λημμάτων στην πλατφόρμα Protégé

Η τρίτη και τελευταία λειτουργία της εργασίας μας είναι η φόρτωση των λημμάτων από το αρχείο csv του βήματος 2.1 στην δημιουργηθείσα οντολογία του βήματος 2.2. Ο κώδικας διαβάζει για κάθε λήμμα στο αρχείο csv, την στήλη με το Μέρος του Λόγου στο οποίο ανήκει για να το εντάξει στην κατάλληλη κλάση της οντολογίας. Επιπλέον, διαβάζει τις τιμές στις υπόλοιπες στήλες και τις προσθέτει στα κατάλληλα Data ή Annotation Properties. Στο τέλος, προκύπτει μια οντολογία με τα λήμματα του «LexikoOrganised.csv» αποθηκευμένη ως «PartsOfSpeech.owl».

Διάγραμμα 3: *Λειτουργία δημιουργίας της Οντολογίας owl*

3. ΣΧΕΔΙΑΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Στο προηγούμενο κεφάλαιο παρουσιάσαμε εν συντομία το τι επιτελεί καθεμία από τις τρεις λειτουργίες της εφαρμογής μας. Στο παρόν κεφάλαιο αναπτύσσουμε εκτενέστερα το κομμάτι αυτό, αναλύοντας ακόμη περισσότερο τα δεδομένα, τις διαδικασίες και τα αποτελέσματα κάθε λειτουργίας.

3.1 Μονάδα 1: Εξόρυξη Λημμάτων από το Διαδικτυακό Λεξικό

Σκοπός
Η πρώτη μονάδα της εφαρμογής έχει σαν στόχο την δημιουργία ενός αρχείου csv στο οποίο θα συμπεριλαμβάνονται όλα τα λήμματα που περιέχονται στο ηλεκτρονικό «Λεξικό της Κοινής Νέας Ελληνικής» του Μανώλη Τριανταφυλλίδη. Επίσης, στο αρχείο αυτό για κάθε λήμμα θα πρέπει να είναι διακριτό το Μέρος του Λόγου στο οποίο ανήκει και οι λοιπές ιδιότητές του, όπως αυτές αναφέρονται και στα προηγούμενα κεφάλαια.

Διαδικασία
Ένα λήμμα στο διαδικτυακό Λεξικό που χρησιμοποιούμε ως πηγή βρίσκεται στην παρακάτω μορφή:

> **ψεύτης** ο [pséftis] Ο10 θηλ. **ψεύτρα** [pséftra] Ο25 : ως χαρακτηρισμός προσώπου που λέει ψέματα· (πρβ. *ψευδολόγος, τερατολόγος, παραμυθάς*). ΑΝΤ ειλικρινής: *Μεγάλος / τρομερός / φοβερός / αδιάντροπος ~. Βγαίνω ~*, η εξέλιξη των γεγονότων διαψεύδει προβλέψεις ή εκτιμήσεις μου· διαψεύδομαι: *Μακάρι να βγω ~, αλλά δε νομίζω ότι έχεις δίκιο*. ΦΡ *ο Θεός να με βγάλει ψεύτη*, μακάρι να διαψευστώ. || *ρώτα τον μπάρμπα μου τον ψεύτη*, ως ειρωνική απάντηση σε κπ. που επικαλείται τη μαρτυρία προσώπου αναξιόπιστου. ΠΑΡ *Ο κλέφτης και ο ~ τον πρώτο χρόνο χαίρονται*, γρήγορα αποκαλύπτονται και υφίστανται τις συνέπειες. || (ως επίθ.) για ό,τι μας δίνει μια ψευδή αντίληψη ως προς το τι είναι πραγματικά: *Ψεύτη ντουνιά. Ψεύτρα κοινωνία. Ψεύτρα ελευθερία*, που δεν είναι πραγματική ελευθερία. **ψευτάκος** ο ΥΠΟΚΟΡ (οικ.). **ψευταράκος** ο ΥΠΟΚΟΡ (οικ.). **ψευτράκος** ο ΥΠΟΚΟΡ (οικ.). **ψευταράς** ο θηλ. **ψευταρού** & **ψευτρού** ΜΕΓΕΘ (οικ.).
>
> [αρχ. *ψεύστης* με αποβ. του [s] για απλοπ. του συμφ. συμπλ.· *ψεύ(της) -τρα* (πρβ. ελνστ. *ψεύστρια*)· *ψεύτ(ης) -άκος· ψευτα ρ(άς) -άκος· ψεύτρ(α) -άκος· ψεύτ(ης) -αράς· ψευταρ(άς) -ού· ψεύτρ(α) -ού*]

Από την παραπάνω μορφή, πρέπει να αναλυθεί στην κύρια λέξη και στα επιμέρους χαρακτηριστικά της: *«Λέξη», «Λέξη με άρθρο», «Προφορά», «Ετυμολογία», «Μέρος του Λόγου», «Ορισμός – Επεξήγηση», «Αρσενικός τύπος», «Θηλυκός τύπος», «Αντώνυμο», «Συνώνυμο», «Συνώνυμη φράση»,*

«Υποκοριστικό», «Μεγεθυντικό», «Φράση» (παράδειγμα χρήσης του λήμματος), «Παροιμία».
Επομένως, στο συγκεκριμένο παράδειγμα ο κώδικάς μας πρέπει να χωρίσει το λήμμα ως εξής:

- **Λέξη:** ψεύτης
- **Λέξη με άρθρο:** ψεύτης ο
- **Προφορά:** pséftis
- **Ετυμολογία:** αρχ. ψεύστης με αποβ. του [s] για απλοπ. του συμφ. συμπλ.· ψεύ(της) -τρα (πρβ. ελνστ. ψεύστρια)· ψεύτ(ης) -άκος· ψευτα ρ(άς) -άκος· ψεύτρ(α) -άκος· ψεύτ(ης) -αράς· ψευταρ(άς) -ού· ψεύτρ(α) -ού
- **Μέρος του Λόγου:** Ουσιαστικό
- **Ορισμός:** ως χαρακτηρισμός προσώπου που λέει ψέματα· (πρβ. ψευδολόγος, τερατολόγος, παραμυθάς). ΑΝΤ ειλικρινής: Μεγάλος / τρομερός / φοβερός / αδιάντροπος ~. Βγαίνω ~, η εξέλιξη των γεγονότων διαψεύδει προβλέψεις ή εκτιμήσεις μου· διαψεύδομαι: Μακάρι να βγω ~, αλλά δε νομίζω ότι έχεις δίκιο. ΦΡ ο Θεός να με βγάλει ψεύτη, μακάρι να διαψευστώ. || ρώτα τον μπάρμπα μου τον ψεύτη, ως ειρωνική απάντηση σε κπ. που επικαλείται τη μαρτυρία προσώπου αναξιόπιστου. ΠΑΡ Ο κλέφτης και ο ~ τον πρώτο χρόνο χαίρονται, γρήγορα αποκαλύπτονται και υφίστανται τις συνέπειες. || (ως επίθ.) για ό,τι μας δίνει μια ψευδή αντίληψη ως προς το τι είναι πραγματικά: Ψεύτη ντουνιά. Ψεύτρα κοινωνία. Ψεύτρα ελευθερία, που δεν είναι πραγματική ελευθερία. ψευτάκος ο ΥΠΟΚΟΡ (οικ.). ψευταράκος ο ΥΠΟΚΟΡ (οικ.). ψευτράκος ο ΥΠΟΚΟΡ (οικ.). ψευταράς ο θηλ. ψευταρού & ψευτρού ΜΕΓΕΘ (οικ.).
- **Αρσενικός τύπος:** -
- **Θηλυκός τύπος:** ψεύτρα
- **Αντώνυμο:** ειλικρινής
- **Συνώνυμο:** -
- **Συνώνυμη φράση:** -
- **Υποκοριστικό:** ο ψευτάκος (για λόγους αποφυγής πολυπλοκότητας επιλέγουμε ένα τύπο)
- **Μεγεθυντικό:** ο ψευταράς
- **Φράση:** ο Θεός να με βγάλει ψεύτη
- **Παροιμία:** Ο κλέφτης και ο ~ τον πρώτο χρόνο χαίρονται.

Για να επιτευχθεί το παραπάνω, ο κώδικάς μας αρχικά δημιουργεί στο αρχείο μας τις στήλες *«word», «words», «spelling», «etymology», «PartsOfSpeech», «definition», «annotation», «male», «female», «antonym», «synonym», «synonymPhrase», «diminutive», «magnifying», «phrase»* και «*saying»*.
Στη συνέχεια, χωρίζει το λήμμα σε δύο μέρη, εκείνο που βρίσκεται προ του **πρώτου δίστιγμου «:»** και σε εκείνο που έπεται αυτού. Τα δύο μέρη καταχωρούνται ως τιμές στις μεταβλητές *«front»* και *«rear»* αντίστοιχα. Αυτό συμβαίνει, γιατί υπάρχουν χαρακτηριστικά που πρέπει να ληφθούν υπόψιν μόνο αν περιέχονται πριν το πρώτο δίστιγμο και άλλα μόνο αν βρίσκονται μετά από αυτό. Για παράδειγμα, ένα μέσο ώστε να αναγνωριστεί ο θηλυκός τύπος του λήμματος «ψεύτης» είναι το «θηλ.» που προηγείται του «ψεύτρα». Το «θηλ.» αυτό λαμβάνεται υπόψιν μόνο αν βρίσκεται πριν το πρώτο «:». Αν βρίσκεται μετά από αυτό, τότε δεν δηλώνει τον θηλυκό τύπο του αρχικού λήμματος, αλλά

κάποιας άλλης λέξης, π.χ. του μεγεθυντικού του (βλ. «θηλ. ψευταρού & ψευτρού»).

word	words	spelling	etymology	PartsOfSpeech	definition	male	female	antonym	synonym	synonymPhrase	diminutive	magnifying	phrase	saying
α	α το	á	από το φθόγ	NonInflectedNoun	(προφ.) το γράμμα άλφα.									
α (1)	α 1	á	ηχομιμ., (πρβ	Interjection	1.δηλώνει ποικίλα συναισθήματα ανάλογα με το νόημα του λόγου, τον τόνο και το χρωματισμό της φωνής ή τις κινήσεις του σώματος και τ									~! γεια σου / μπράβο,
α (2)	α 2 & άι	άι & άι	α: σύντμ. τοι	Interjection	πριν από επιφωνηματική πρόταση· (πρβ. άντε)· ανάλογα με τα συμφραζόμενα και το χρωματισμό της φωνής εκφράζει: α. αγανάκτηση, δυσφορία, αποπομπή· πήγαινε, τράβα: (με προσ									
α (3)	α καπέλα	akapéla	λόγ. < ιταλ. a	NonInflectedAdject	(μουσ.) για χορωδιακό έργο που εκτελείται χωρίς συνοδεία οργάνων: Μουσική ~. || (ως επίρρ.): Αυτή η σύνθεση τραγουδιέται ~.									
α-	α- 1	a	αρχ. στερ. πρ	Blank	στερητικό πρόθημα. 1. δηλώνει: α. (σε επίθετα) το αντίθετο από αυτό που εκφράζει η πρωτότυπη λέξη: αβέβαιος, άκακος, αναρμόδιος, ανάφαγος, ανέμελος, ανεπίσημος, ανέξοδος, αν									
α- (1)	α- 2		αρχικό α- σε	Blank	το αρχικό α- προπαροξύτονων ρηματικών επιθέτων σε -τος ως στερητικό για να δηλώσει ότι το προσδιοριζόμενο δεν έχει υποστεί την ενέργεια που εκφράζει το ρήμα από το οποίο πα									
α- (2)	α- 3		μον. προτακτ	Blank	(λαϊκότρ., λογοτ.) προτακτικό· μπαίνει από αναλογία ή συνεκφορά στην αρχή ουσιαστικών, επιθέτων και ρημάτων που αρχίζουν από σύμφωνο: (βδέλλα) αβδέλλα, (λυγαριά) αλυγαριά									
α- (3)	α- 4		όπως στο α-	Blank	(λαϊκότρ.) αρκτικό· σε ουσιαστικά, επίθετα, ρήματα και επιρρήματα από αλλαγή του αρχικού τους φωνήεντος: (εγγόνι) αγγόνι, (έντερο) άντερο, (ομφαλός) αφαλός· (ελαφρός) αλαφρό									
Α,	Α, α το	álfa	αρχ. Α (σημιτ	NonInflectedParts	1.το πρώτο γράμμα του ελληνικού αλφαβήτου, το ψηφίο άλφα*. 2. στο σύστημα απόδοσης των αριθμητικών με τα γράμματα του αλφαβήτου: α. (με διακριτικό τονικό σημάδι) Α' ή α' =									
αβαείο	αβαείο το	avaío	λόγ. αβά(ς) -(	Ousiastiko	1.μοναστήρι ρωμαιοκαθολικών που διοικείται από αβά. || εκκλησία που παλαιότερα ήταν αβαείο: Στο ιστορικό ~ του Γουεστμίνστερ βρίσκονται οι τάφοι των βασιλέων της Μεγάλης									
αβαθής	αβαθής -ής -(	avaθís	λόγ. < ελνστ.	Epitheto	που δεν έχει βάθος, άβαθος, ρηχός. ΑΝΤ βαθ βαθύς:									
αβαθμολόγητος	αβαθμολόγη	avaθmolóji	λόγ. α- 1 βαθ	Epitheto	που δε βαθμολογήθηκε ακόμα. ΑΝΤ βαθμολο βαθμολογημένος:									
άβαθος	άβαθος -η -ο	ávaθos	α- 1 βάθ(ος)	Epitheto	1α.που δεν έχει μεγάλο ή αρκετό βάθος· ρηχός, ανάβαθος, αβαθής: Άβαθο πηγάδι. Άβαθη κοίτη / όχθη / σπηλιά. Άβαθα νερά. β. (μτφ.) που δεν προχωρεί σε βάθος, επιπόλαιος, ρηχός:									
άβαθα	άβαθα			Adverb										
αβαθούλωτος	αβαθούλωτο	avaθúlotos	α- 1 βαθουλ	Epitheto	που δεν είναι ή που δεν έγινε βαθουλός. ΑΝΤ βαθουλωμένος.									
άβακας	άβακας ο	ávakas	λόγ.: Ι1, Ιl: αρ	Ousiastiko	Ι1.κατάλληλα διαμορφωμένη επίπεδη επιφάνεια που χρησιμοποιείται σε επιτραπέζια παιχνίδια, π.χ. σκάκι, τάβλι. 2. (αρχιτ., αρχαιολ.) η πλάκα που αποτελεί το επάνω μέρος του κιονο									
αβάκιο	αβάκιο το	avákio	λόγ. < αρχ. ἄ	Ousiastiko	(λόγ.) μικρός άβακας, η πλάκα στην οποία έγραφαν οι μικροί μαθητές.									
αβαλσάμωτος	αβαλσάμωτο	avalsámoto	α- 1 βαλσαμ	Epitheto	που δε βαλσαμώθηκε· αταρίχευτος. ΑΝΤ βαλ βαλσαμωμένος.									
άβαλτος	άβαλτος -η -(	ávaltos	α- 1 βαλ- (βά	Epitheto	που δεν τον έχουν βάλει, δεν τον έχουν ακόμη βαλμένος:									
αβαγκάρντ	αβαγκάρντ	avangárd	λόγ. < γαλλ.	NonInflectedNoun	αυτός ή αυτοί που πρωτοπορούν ή που επιδιώκουν να πρωτοπορούν προβάλλοντας τις πιο προωθημένες ή ακραίες θέσεις· πρωτοπορία: Η ~ του συνδικαλιστικού κινήματος / της τέχνη									
αβαγκαρντισμός	αβαγκαρντι	avangardiz	λόγ. αβαγκ	Ousiastiko	η τάση για πρωτοπορία: Ό,τι χαρακτηρίζει μερικούς νεότερους καλλιτέχνες είναι ένας άκρατος, υστερικός ~.									
αβανιά	αβανιά η	avaná	μον. αβάν(ης	Ousiastiko	(λαϊκότρ., λογοτ.) 1. άδικη κατηγορία· συκοφαντία, κακολογία: Του κόλλησαν την ~ πως τάχα αυτός ήταν ο κλέφτης. Πιο πολύ τον έπνιγαν οι αβανιές του κόσμου. 2. ζημιά, κακοτυχία,									
αβανιάρης	αβανιάρης -(	avanáris	αβαν(ιά) -ιάρ	Epitheto	(λαϊκότρ., λογοτ.) που συνηθίζει να λέει, να διαδίδει αβανιές· συκοφάντης.									
αβάνς	αβάνς το	avánς	λόγ. < γαλλ.	NonInflectedNoun	η προαναφλέξη σε μηχανές εσωτερικής καύσεως.									
αβάντα	αβάντα η	avánda	παλ. ιταλ. αν	Ousiastiko	1α.(προφ.) πλεονέκτημα: Αυτή η δουλειά έχει πολλές αβάντες. β. κέρδος, όφελος που συνήθ. προέρχεται από επιλήψιμη διαδικασία· μίζα: Άμα σου τελειώσω τη δουλειά, τι ~ θα πάρω									
αβανταδόρικος	αβανταδόρι	avandaδór	αβανταδόρ(	Epitheto	που αφορά τον αβανταδόρο ή που έχει σχέση με αυτόν: Αβανταδόρικο κόλπο / τέχνασμα. Αβανταδόρικη σκηνοθεσία. αβανταδόρικα ΕΠΙΡΡ.									
αβανταδόρικα	αβανταδόρικα			Adverb										
αβανταδόρος	αβανταδόρο	avandaδór	αβάντ(α) -αδ	Ousiastiko	1α.παίχτης τυχερού παιχνιδιού αβανταδόρισσα									
αβαντάζ	αβαντάζ το	avantáz	λόγ. < γαλλ.	NonInflectedNoun	πλεονέκτημα: Το διαμέρισμα έχει πολλά ~. Δέξου την πρότασή του, έχει πολλά ~.									
αβάντζα	αβάντζα η	avándza	αβαντ(άρω	Ousiastiko	(προφ.) 1. προκαταβολή μισθού, οφειλής ή χρέους· μπροστάντζα: Πήρα ~ τρία χιλιάρικα. 2. αβάντα2. 3. συγκαταβατικό φέρσιμο: Δε θα δεχτώ να μου κάνει τέτοιες αβάντσες αυτός ο τ									
αβάντζο	αβάντζο το	avándzo	ιταλ. avanzo	Ousiastiko	στις ΦΡ πάμε ~?, για τυχερά παιχνίδια, όταν οι παίχτες συμφωνούν να αυξήσουν τον αριθμό των πόντων του παιχνιδιού. δίνω ~, παραχωρώ	πάμε ~?,								
αβάντι	αβάντι	avánti	ιταλ. και βεν	Adverb	(προφ.) εμπρός, άρχισε ή ας αρχίσουμε. ΦΡ ~ μαέστρο! (συχνά και πειραχτικά).								~ μαέστρο! (συχνά και πειραχτικά).	
αβάρα	αβάρα	avára	αβαρ(άρω) -(	NonInflectedParts	(γαυτ.) παράγγελμα το οποίο αντιστοιχεί στην προστακτική απομάκρυνε: ~, να μη χτυπήσουμε στο βράχο. ~ από δω, φύγε γρήγορα, στρίβε.	κάνω / βάζω ~,								
αβαράρω	αβαράρω	avarár	ιταλ. varare(	Verb	(γαυτ.) απομακρύνω βάρκα ή μικρό πλοίο από κάποιο σημείο, απομακρύνομαι με τα χέρια ή με το κουπί: Μπαίνουμε στις βάρκες και αβαράρουν									

Εικόνα 3*: Οι στήλες στο αρχείο csv*

Έπειτα, ο κώδικας διαιρεί το λήμμα στα επιμέρους τμήματά του χρησιμοποιώντας ορισμένα αναγνωριστικά ή λέξεις-κλειδιά που βρίσκει σε αυτό. Τα αναγνωριστικά αυτά είναι συγκεντρωμένα σε ένα τμήμα της έρευνας της κας Σαμαρείδη και μας τα παρέθεσε εκείνη [17]. Ενδεικτικά αναφέρουμε τον τύπο «ΦΡ» μετά από τον οποίο βρίσκεται φράση, το «ΠΑΡ» που προηγείται της παροιμίας, το «ΑΝΤ» που προηγείται του αντώνυμου, «ΣΥΝ» του συνώνυμου, το «θηλ.» που είδαμε παραπάνω κ.ά.

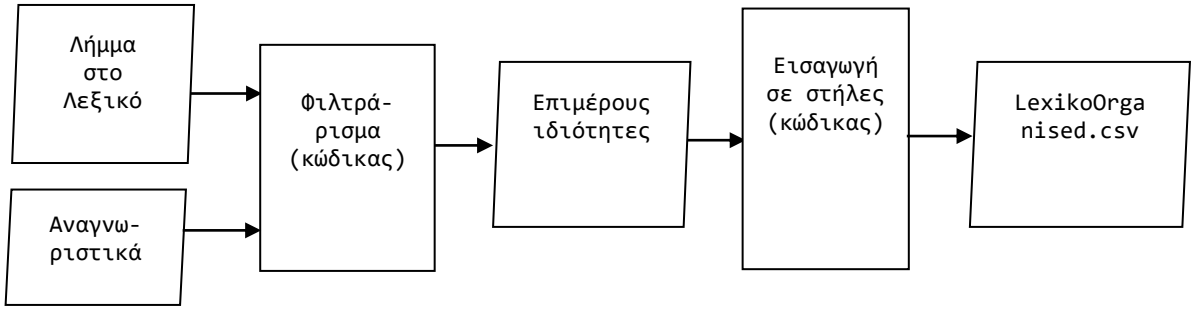
```
268
269         if ' αρσ. ' in front:
270             male = words[words.index('αρσ.')+1]
271             if male=='(στη':
272                 male = words[words.index('αρσ.')+4]
273             if male=='και':
274                 male = ''
```

Εικόνα 4: *Παράδειγμα στον κώδικα για τον ορισμό του αρσενικού τύπου του λήμματος*

Για την τιμή της στήλης «PartsOfSpeech», ο κώδικας ψάχνει στην μεταβλητή «front», η οποία περιέχει το μέρος από την αρχή μέχρι το πρώτο δίστιγμο, τον τύπο που βρίσκεται μετά τις πρώτες αγκύλες ([ ]). Ο τύπος αυτός δηλώνει το Μέρος του Λόγου στο οποίο ανήκει το λήμμα. Στο παράδειγμά «ψεύτης», ο τύπος είναι «Ο10» και δηλώνει ότι το λήμμα ανήκει στα Κλιτά Ουσιαστικά. Οι υπόλοιποι τύποι που υπάρχουν ως αναγνωριστικά του μέρους του λόγου στο λεξικό του Μανώλη Τριανταφυλλίδη συγκεντρώθηκαν από την κα Νικολέττα Σαμαρείδη και είναι οι [17]:

- *Ο (άκλ.),* για άκλιτα Ουσιαστικά
- *Ε (άκλ.)* ή *(άκλ., ως επίθ.),* για άκλιτα επίθετα

- *επιφ. (άκλ.)* ή *επιφ.*, για επιφώνημα
- *αντων. αόρ. (άκλ.)*, για Άκλιτη Αόριστη Αντωνυμία
- *(άκλ.) αντων. ερωτ.*, για Άκλιτη Ερωτηματική Αντωνυμία
- *αντων. αναφ. (άκλ.)*, για Άκλιτη Αναφορική Αντωνυμία
- *(άκλ.) αριθμτ. επίθ. απόλ.*, για Άκλιτο Αριθμητικό Επίθετο Απόλυτο
- *(άκλ., ως επίρρ.)* ή *(άκλ.) (ως επίρρ.)* ή *(ως επίρρ.)* ή *επίρρ.* για Επίρρημα
- *επίρρ. χρον.* για επίρρημα χρονικό
- *επίρρ. τοπ.*, για επίρρημα τοπικό
- *επίρρ. τροπ.*, για επίρρημα τροπικό
- *επίρρ. ποσ.*, για επίρρημα ποσοτικό
- *επίρρ. βεβ.*, για επίρρημα βεβαιωτικό
- *Ρ*, για Ρήματα
- *αντων. δεικτ.*, για Δεικτική Αντωνυμία
- *αντων. προσ.*, για Προσωπική Αντωνυμία
- *αντων. αόρ.*, για Αόριστη Αντωνυμία
- *αντων. αναφ.*, για Αναφορική Αντωνυμία
- *αντων. αυτοπ.*, για Αυτοπαθή Αντωνυμία
- *αντων. κτητ.*, για Κτητική Αντωνμία
- *αντων. ερωτ.*, για Ερωτηματική Αντωνυμία
- *πρόθ.*, για Πρόθεση
- *σύνδ. υποθ.*, για Υποθετικό Σύνδεσμο
- *σύνδ. τελ.*, για Τελικό Σύνδεσμο
- *σύνδ. διστ.*, για Διστακτικό Σύνδεσμο
- *σύνδ. ειδ.*, για Ειδικό Σύνδεσμο
- *σύνδ. αντιθ.*, για Αντιθετικό Σύνδεσμο
- *σύνδ. ερωτ.*, για Ερωτηματικό Σύνδεσμο
- *σύνδ. αιτιολ.*, για Αιτιολογικό Σύνδεσμο
- *σύνδ. διαχ.*, για Διαχωριστικό Σύνδεσμο
- *σύνδ. συμπλεκτ.*, για Συμπλεκτικό Σύνδεσμο
- *σύνδ. συμπερ.*, για Συμπεριληπτικό Σύνδεσμο
- *σύνδ. αποφατικός.*, για Αποφατικό Σύνδεσμο
- *αριθμτ. επίθ.*, για Αριθμητικό Επίθετο
- *αριθμτ. επίθ. τακτ.*, για Αριθμητικό Επίθετο Τακτικό
- *αριθμτ. επίθ. απόλ.*, για Αριθμητικό Επίθετο Απόλυτο
- *αριθμτ. αναλ.*, για Αριθμητικό Αναλογικό
- *αριθμτ. τακτ.*, για Αριθμητικό Τακτικό
- *αριθμτ. πολλαπλ.*, για Αριθμητικό Πολλαπλασιαστικό
- *αριθμτ. απολ.*, για Αριθμητικό απόλυτο
- *αριθμ. περιλ.*, για Αριθμητικό Επίθετο Περιληπτικό
- *Ε*, για Επίθετο
- *αόριστο άρθρο,*
- *άρθρο οριστικό.*



Διάγραμμα 4: *Λειτουργία δημιουργίας της Οντολογίας owl*

3.2 Μονάδα 2: Δημιουργία Οντολογίας στο πρότυπο owl

Σκοπός

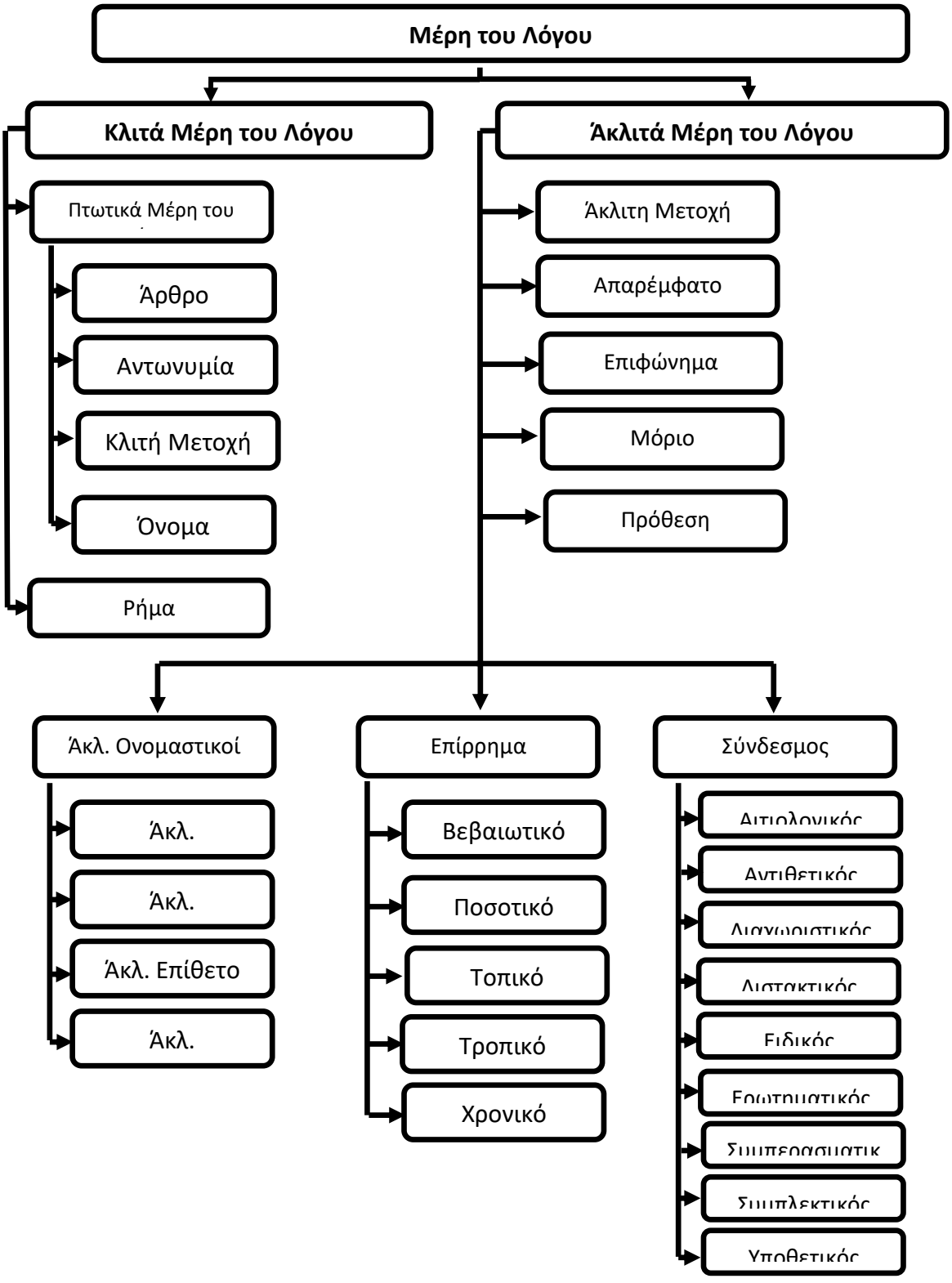
Το αποτέλεσμα του δεύτερου τμήματος της εργασίας μας είναι μια Οντολογία σε πρότυπο owl η οποία να απεικονίζει την δομή των μερών του Λόγου όπως αυτή ορίζεται στην Γραμματική της Νέας Ελληνικής Γλώσσας (βλέπε διάγραμμα 5). Ταυτόχρονα, στην οντολογία πρέπει να είναι ορισμένες και οι ιδιότητες που θα έχει κάθε λήμμα, όπως αυτές βρίσκονται στο αρχείο csv και να είναι σωστά ταξινομημένες σε Data Properties και Annotation Properties.

Διαδικασίες

Στον κώδικά μας ορίζουμε με μια εντολή της βιβλιοθήκης «owlready» της Python την νέα μας οντολογία και στην συνέχεια εγγράφουμε σε αυτήν τις κατάλληλες κλάσεις. Πρώτα ορίζουμε την «Μέρη _Του_Λόγου/PartsOfSpeech» και κάτω από αυτήν τις υποκλάσεις «Κλιτά_Μέρη_Του_Λόγου/Inflected_Parts_Of_Speech», την «Άκλιτα Μέρη του Λόγου/Non inflected Parts Of Speech» και την «Blank». Η κλάση «Blank» δημιουργήθηκε γιατί κατά την εξαγωγή των λημμάτων παρατηρήσαμε ότι υπάρχουν λήμματα στο Λεξικό, τα οποία δεν έχουν κάποιο αναγνωριστικό όσον αφορά το Μέρος του Λόγου στο οποίο ανήκουν. Επομένως, στο αρχείο csv δεν είχαν κάποια τιμή στο χαρακτηριστικό «PartsOfSpeech» και ο κώδικάς μας δεν θα μπορούσε να τα εντάξει κάπου. Συνεπώς, αποφασίσαμε να βάλουμε μία κλάση Blank για να συμπεριλάβει και αυτά τα λήμματα.

Η διαδικασία ορισμού της κάθε υποκλάσης γίνεται με αναφορά στην κλάση στην οποία ανήκει. Για παράδειγμα, όταν ορίζουμε την «Κλιτά_Μέρη_του_Λόγου/Inflected_Parts_Of_Speech», γράφουμε:

```
class
Κλιτά_Μέρη_του_Λόγου/Inflected_Parts_Of_Speech(Μέρη_Του_Λογού/Parts_of_S
peech):
      pass
```



Διάγραμμα 5: *Λειτουργία δημιουργίας της Οντολογίας owl*

Τα Data Properties και τα Annotation Properties υπάρχουν σαν ειδικός τομέας σε κάθε Οντολογία του Protégé και αφορούν έννοιες που χαρακτηρίζουν το λήμμα. Για τον λόγο αυτόν εμείς θα δημιουργήσουμε ορισμένες κλάσεις που θα εντάσσονται στα Data και Annotation Properties.

Στα Data Properties εντάσσουμε τις παρακάτω κλάσεις

- **differentFormOfFemale**
- **differentFormOfMale**
- **differentFormOfLemma**
- **hasAntonyms** για το αντώνυμο του λήμματος
- **hasAsFemale** για το θηλυκό γένος κάποιου λήμματος αρσενικού γένους
- **hasAsMale** για το αρσενικό γένος κάπου λήμματος θηλυκού γένους
- **hasDiminutive** για το υποκοριστικό ενός λήμματος
- **hasEtymology** για την ετυμολογία του λήμματος
- **hasGender** για να δηλώσει το αν έχει γένος η λέξη ή όχι
- **hasInflection**
- **hasMagnifying** για το μεγεθυντικό του λήμματος
- **hasSynonyms** για το συνώνυμο του λήμματος
- **hasSynonymousPhrase** για τη συνώνυμη φράση του λήμματος
- **hasValue**
- **hasVocalTranscription** για την προφορά του λήμματος
- **inflectionalCategoryTriantafyllides**
- **isFemale** για δήλωση του αν είναι το λήμμα θηλυκού γένους ή όχι
- **isMale** για δήλωση του αν είναι το λήμμα αρσενικού γένους ή όχι
- **isNeutral** για δήλωση του αν είναι το λήμμα ουδέτερου γένους ή όχι
- **isPlural** για δήλωση του αν είναι το λήμμα σε πληθυντικό ή όχι
- **isSingular** για δήλωση του αν είναι το λήμμα σε ενικό ή όχι.

Στα **Annotation Properties** εντάσσουμε τις:

- **definition**
- **definitionFromDictionaryOfModernStandardGreek** για τον ορισμό του λήμματος, όπως αυτός υπάρχει στο Λεξικό Τριανταφυλλίδη
- **example**
- **expression**
- **expressionFromDictionaryOfModernStandardGreek**
- **inflection**
- **inflectionParadigmsTriantafyllides**
- **isDefinedBy**
- **label** για την καταχώρηση του λήμματος ως λέξη
- **morphologicalVariety**
- **orthography**
- **phrase**
- **phraseFromDictionaryofModernStandardGreek** για την εισαγωγή της Φράσης (column «phrase» του αρχείου csv) από το Λεξικό του Τριανταφυλλίδη
- **pronunciation**
- **saying**
- **sayingFromDictionaryofModernStandardGreek** για την εισαγωγή της παροιμίας (column «saying» του αρχείου csv) από το Λεξικό του Τριανταφυλλίδη

- **seeAlso**
- **semantics**
- **style**
- **syntax**
- **typeFormation**
- **image**.

Κάποιες κλάσεις των Data και των Annotation Properties για τις οποίες δεν δίνουμε ορισμό, είναι γιατί τις προσθέσαμε εκεί κατόπιν υπόδειξης της κας Σαμαρείδη για μελλοντική χρήση από την ίδια. Όλη η δομή μάς δόθηκε ως οδηγία από εκείνην, απλώς για τις κλάσεις που χρησιμοποιήσαμε στον κώδικά μας έχουμε την δυνατότητα να δώσουμε και τον λόγο ύπαρξής τους.

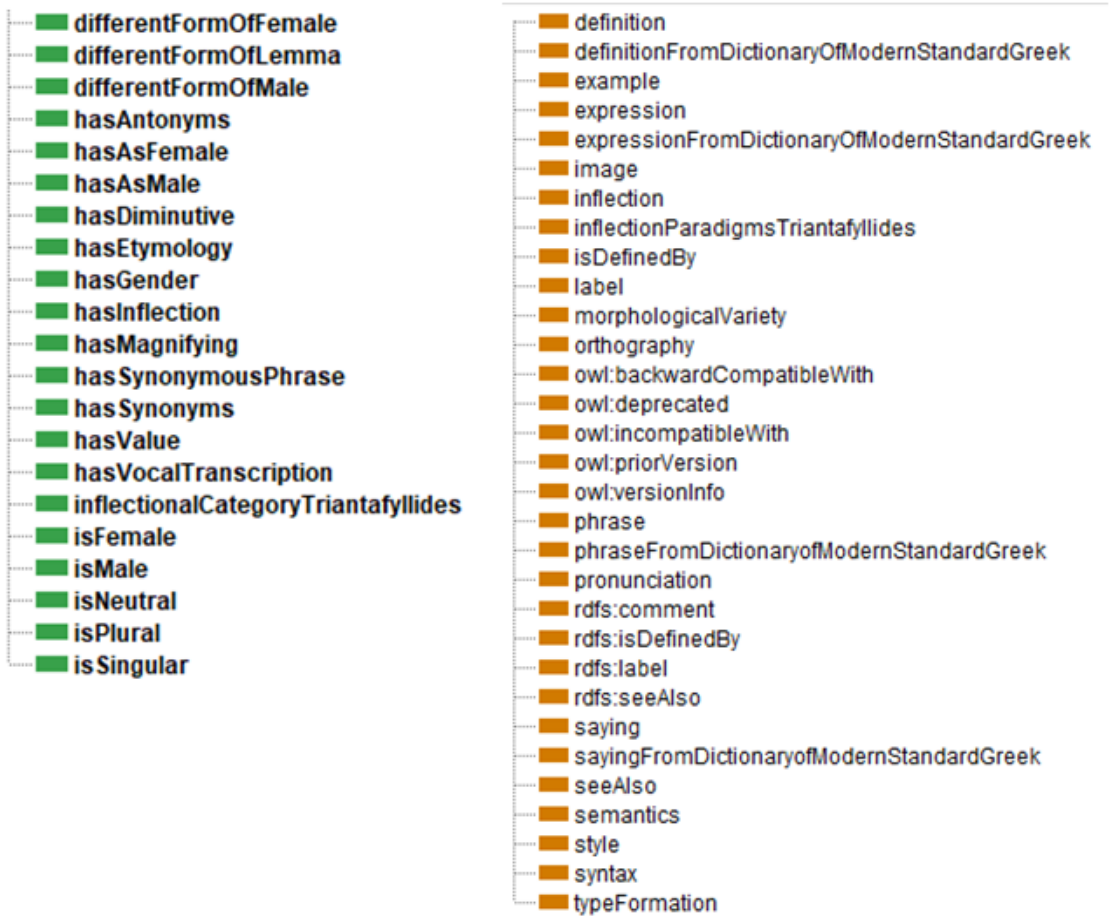
Κατόπιν όλων των παραπάνω, με μια εντολή αποθηκεύουμε την οντολογία σε ένα νέο αρχείο «PartsOfSpeech.owl».

Για να δούμε την οντολογία μας, θα πρέπει να έχουμε εγκαταστήσει το Protégé στον υπολογιστή μας. Ανοίγουμε το Protégé και επιλέγουμε «File < Open» και στη συνέχεια την τοποθεσία του αρχείου «PartsOfSpeech.owl».

Παρακάτω φαίνονται ορισμένες εικόνες με τις κλάσεις και τις υποκλάσεις της Οντολογίας μας, καθώς και των Data και Annotation Properties, όπως φαίνονται στην πλατφόρμα Protégé μετά την εκτέλεση του κώδικά μας.

Εικόνα 5: *Άκλιτα Μέρη του Λόγου και υποκλάσεις*

Εικόνα 6: *Κλιτά Μέρη του Λόγου και υποκλάσεις*



Εικόνα 7: *Data Properties και Annotation Properties*

3.3 Μονάδα 3: Φόρτωση Λημμάτων και χαρακτηριστικών στην Οντολογία

Σκοπός
Από το τελευταίο μέρος του κώδικα προκύπτει και ο τελικός στόχος της εργασίας, δηλαδή η οντολογία «PartsOfSpeech» η οποία θα περιέχει όλα τα λήμματα του Λεξικού του Τριανταφυλλίδη.

Δεδομένα που διαχειρίζεται
Τα δεδομένα που διαχειρίζεται είναι τα οργανωμένα λήμματα που βρίσκονται στο αρχείο «LexikoOrganised.csv» και η υπάρχουσα οντολογία «PartsOfSpeech.owl».

Διαδικασίες
Στο στάδιο αυτό περιέχονται τρεις διαδικασίες. Η πρώτη είναι η συνάρτηση η οποία διαβάζοντας το csv δημιουργεί ένα individual της Οντολογίας για κάθε νέα γραμμή του αρχείου – πρακτικά για κάθε λήμμα. Αφού δημιουργήσει το individual, το εντάσσει στην κλάση του Μέρους του Λόγου στο οποίο ανήκει.

```
def add_individual(row):
    word = row['word']
    part_of_speech = row['PartsOfSpeech']

    # Βρίσκουμε την κατάλληλη κλάση
    cls = onto.search_one(iri="*"+part_of_speech)
    if cls is None:
        print(f"Η κλάση '{part_of_speech}' δεν βρέθηκε στην οντολογία.")
        return
```

Εικόνα 8: *Συνάρτηση Δημιουργίας Individuals*

Στη συνέχεια έχουμε μια συνάρτηση για την ένταξη των εκάστοτε χαρακτηριστικών στα κατάλληλα Data Properties του κάθε individual.

```
# Συνάρτηση για να ορίσουμε data properties
    def set_data_property(individual, prop_name, value):
        prop = getattr(individual, prop_name, None)
        if prop is None:
            print(f"Η ιδιότητα '{prop_name}' δεν υπάρχει ή δεν είναι σωστά
ορισμένη.")
            return
        if isinstance(prop, list):  # Αν είναι λίστα, προσθέτουμε την τιμή
            prop.append(value)
        else:  # Αλλιώς, αναθέτουμε απευθείας την τιμή
            setattr(individual, prop_name, value)
```

Εικόνα 9: *Συνάρτηση για τον ορισμό των Data Properties*

Τέλος, αναπτύσσουμε μια τρίτη συνάρτηση για την ανάθεση των αντίστοιχων χαρακτηριστικών στα σωστά Annotation Properties, όποτε αυτό πρέπει να γίνει.

```python
# Συνάρτηση για να ορίσουμε annotation properties
def set_annotation_property(individual, prop_name, value):
    prop = getattr(individual, prop_name, None)
    if prop is None:
        print(f"Η ιδιότητα '{prop_name}' δεν υπάρχει ή δεν είναι σωστά ορισμένη.")
        return
    setattr(individual, prop_name, value)
```

Εικόνα 10: *Συνάρτηση για την κατανομή στα Annotation Properties*

Αφού ορίσουμε τις παραπάνω συναρτήσεις για τα Data και τα Annotation Properties μετά χρησιμοποιούμε πολλαπλές εντολές if για να ορίσουμε ποια column του αρχείου csv που ο κώδικάς μας επεξεργάζεται ανήκουν στα Data και ποια στα Annotation Properties. Πιο συγκεκριμένα, για τα columns *spelling, etymology, male, female, antonym, synonym, synonymPhrase, diminutive, magnifying* και *phrase* χρησιμοποιούμε την «set_data_property()», ενώ για τα *definition* και *saying* τη συνάρτηση «set_annotation_property()».

Τέλος, ο κώδικάς μας αποθηκεύει τις αλλαγές που έκανε στο ίδιο αρχείο της οντολογίας.

```python
# Αποθηκεύουμε την ενημερωμένη οντολογία σε αρχείο OWL
onto.save(file="C:/Users/503344134.HCAD/.vscode/Protege/Teliko/3.PartsOfSpeech7.owl", format="rdfxml")
```

Εικόνα 11: *Εντολή αποθήκευσης της Οντολογίας μετά την φόρτωση των Λημμάτων*

4. ΥΛΟΠΟΙΗΣΗ

Η υλοποίηση του κώδικα παρατίθεται στο Παράρτημα της παρούσας εργασίας.

5. ΑΞΙΟΛΟΓΗΣΗ ΛΕΙΤΟΥΡΓΙΑΣ

Κύριος στόχος της εφαρμογής ήταν μέσω κώδικα Python να καταφέρει να εξάγει σωστά τα λήμματα από μία διαδικτυακή πηγή και ύστερα, να τα εισάγει ταξινομημένα με βάση το Μέρος του Λόγου τους σε μια οντολογία στην πλατφόρμα του Protégé. Θεωρούμε πως αυτός ο στόχος επιτεύχθηκε σε έναν ικανοποιητικό βαθμό, με ορισμένες δυσκολίες που αναφέρουμε παρακάτω.

1. <u>Μοναδικότητα Λημμάτων</u>
   Ορισμένα λήμματα είναι παρόμοια μεταξύ τους, στον βαθμό που δεν μπορούσαν να γίνουν διακριτά από τον κώδικά μας. Παράδειγμα, αποτελούν τα λήμματα

> **α 1** [á] επιφ. : 1.δηλώνει ποικίλα συναισθήματα ανάλογα με το νόημα του λόγου, τον τόνο και το χρωματισμό της φωνής ή τις κινήσεις του σώματος και τους μορφασμούς του προσώπου (συνήθ. συνοδεύεται από επιφωνηματική ή ερωτηματική φράση ή πρόταση): α. χαρά, ευχαρίστηση, θαυμασμό: ~! τι όμορφα που είναι εδώ! ~! πολύ χάρηκα που σε είδα! || με επανάληψη ααα! [aaa◎], για κτ. πολύ εντυπωσιακό, θεαματικό ή νόστιμο. ΦΡ ~! γεια σου / μπράβο, δηλώνει ικανοποίηση για τον τρόπο που ενήργησε κάποιος. β. πόθο ή ευχή ανεκπλήρωτη· ε1β: ~! και να ˈμουν στη θέση σου! ~! και να μου ˈπεφτε το λαχείο! ~! και να μου ˈδιναν όσα ζητούσα! γ. έκπληξη, απορία: ~! πότε ήρθες; ~! πού το βρήκες; || αντίδραση στα λεγόμενα κάποιου: ~! αλήθεια; κι ύστερα;, τι έγινε μετά; ~ στο καλό / στο διάολο!, σώπα, δεν μπορώ να το πιστέψω· (πρβ. άντεII). || τρόμο: ~! με τρόμαξες! || πόνο, λύπη· (πρβ. αχ, βρε, ποπό): ~! κακό που έπαθα! ~! κακό που τους βρήκε! ~! τι κρίμα! ~! πώς στενοχωρέθηκα! δ. κοροϊδία· (πρβ. ε): ~! τον κουτό / το βλάκα! ~! ρε◎ ε. αποστροφή, αγανάκτηση· (πρβ. ε): ~! τον ελεεινό / τον απαίσιο! ~! μα δεν υποφέρεσαι πια! ~! να σου πω, το ˈχεις παρακάνει πια! || επιτείνει τη σημασία του που 3: ~! που να (σε) πάρει η ευχή / ο διάολος. || απογοήτευση: ~! ρε, πώς καταντήσαμε! ~! ρε, τι γίνεται στον κόσμο! στ. απειλή: ~! και να σε πιάσω, αλίμονό σου! ~! έτσι είσαι; να δεις κι εγώ! ζ. ανυπομονησία: ~! αργείς πολύ! ~! δεν μπορώ να περιμένω! η. βεβαίωση: ~! ναι! αυτό ακριβώς θέλω! ~! μάλιστα, αυτό είναι, το βρήκες! || (για κτ. που το ξέχασε κάποιος και ξαφνικά το θυμάται): ~! ξέχασα να σου πω τι έμαθα. Λοιπόν, τι λέγαμε; ~! ναι◎ θ. έντονη επιθυμία να μη γίνει κτ.: ~! όχι, μη!, δε θέλω. ~! όχι, μην του δώσεις λεφτά! ~! όχι, μην έρχεσαι εδώ! || έντονη αντίρρηση: ~! όχι έτσι! ~! δε συμφωνώ! ι. (προφ.) προσπάθεια, έγνοια: Από το πρωί, ~ να κάνω τις δουλειές του σπιτιού, ~ να ψωνίσω, κατακουράστηκα! ~ το ένα, ~ το άλλο, δεν κατάλαβα πώς πέρασε η ώρα! 2. στη θέση μονολεκτικής απάντησης: α. αρνητικής, με το μπα*: Κρυώνεις; -~ μπα, όχι καθόλου. β. θετικής: Θα έρθεις μαζί μου; -~ θά ˈρθω, ναι, θά ˈρθω, καλά λες. γ. αόριστης μετριαστικής: Έγινες καλά; -~, έτσι κι έτσι. Πώς τα βλέπεις τα πράγματα; -~, ούτε καλά ούτε άσκημα. 3α. με επανάληψη ααα◎ [aaa◎], χωρίς συγκεκριμένη σημασία, τη στιγμή που ο ομιλητής προσπαθεί να βρει καλύτερη διατύπωση της σκέψης του· (πρβ. ε). β. ρυθμικά επαναλαμβανόμενο χρησιμοποιείται ως νανούρισμα: Ααα◎ νάνι. [ηχομιμ., (πρβ. αρχ. p, επιφ. συμπόνιας, ζηλοφθονίας ή περι φρόνησης)]

> α 2 & άι [ái & ái] επιφ. : πριν από επιφωνηματική πρόταση· (πρβ. άντε)· ανάλογα με τα συμφραζόμενα και το χρωματισμό της φωνής εκφράζει: α. αγανάκτηση, δυσφορία, αποπομπή· πήγαινε, τράβα: (με προστ. ή να και υποτ. ρ. με ανάλογη σημ.) ~ ή άι πνίξου / παράτα μας / χάσου / να χαθείς. || (με την πρόθ. σε και έναρθρο ουσ.) ~ ή άι στην ευχή / στο διάβολο / στο καλό. ~ ή άι στη δουλειά σου, πήγαινε, κοίτα, συνέχισε τη δουλειά σου. || φιλική αποδοκιμασία: ~ ή άι να χαθείς! β. παρακίνηση: Άι στο καλό, παιδί μου, και πρόσεχε, άντε, πήγαινε στο καλό. || απορία για το τι θα γίνει: ~ ή άι να δούμε πώς θα τα βγάλουμε πέρα. ~ ή άι να δούμε τι θα γίνει. [α: σύντμ. του άι· άι: άε < αρχ. ἄγε `εμπρός!' (προστ. του ἄγω `πηγαίνω') με αποβ. του μεσοφ. [γ] και διφθογγοπ.]

Για τα παραπάνω λήμματα ο κώδικας του πρώτου μέρους θα εισάγει σε δύο γραμμές της στήλης «word» την τιμή «α». Στο κομμάτι της Άντλησης των Δεδομένων από το διαδικτυακό Λεξικό δεν υπάρχει πρόβλημα, διότι εκεί ο

κώδικάς μας δεν αγνοεί ένα λήμμα αν αυτό είναι όμοιο με κάποιο προηγούμενό του. Το πρόβλημα εμφανίστηκε στο τρίτο μέρος της εργασίας, κατά την εισαγωγή δηλαδή των λημμάτων από το αρχείο csv στην οντολογία «PartsOfSpeech». Εκεί, προσπαθήσαμε με διάφορα αναγνωριστικά να ξεχωρίσουμε τα όμοια λήμματα μεταξύ τους, αλλά δεν υπήρξε αποτέλεσμα. Έτσι, η λύση που σκεφτήκαμε ήταν να επεξεργαστούμε χειροκίνητα τα όμοια λήμματα μέσα από το πρόγραμμα του Excel.

Αρχικά, επιλέξαμε να μας επισημαίνει το excel ως «Bad» (χρωματισμός με κόκκινο χρώμα) τα κελιά εκείνα τα οποία περιείχαν διπλότυπες λέξεις. Στην συνέχεια, προσθέσαμε μια νέα στήλη μετά την στήλη «word» που περιέχει τις λέξεις και με συνάρτηση ορίσαμε σε κάθε διπλότυπη λέξη να εμφανίζεται ο αριθμός εμφανίσεών της. Δηλαδή, αν η λέξη «α» εμφανιζόταν στην στήλη «words» δύο φορές, η συνάρτηση, την πρώτη φορά θα έβαζε δίπλα της τον αριθμό (1) και την δεύτερη τον αριθμό (2). Έπειτα, προσθέσαμε άλλη μία στήλη μετά την «word», η οποία χρησιμοποιούσε την συνάρτηση «CONCATENATE» για να συνενώσει την τιμή της στήλης «word» με την τιμή της στήλης που περιείχε τον αριθμό. Αυτή η συνένωση δεν εφαρμόστηκε στα κελιά που περιείχαν μη διπλότυπα λήμματα. Έτσι, αντί για «α» και «α» πλέον είχαμε δύο τιμές «α-1» και «α-2». Μόνο για τα διπλότυπα λήμματα, αντικαταστήσαμε την τιμή της στήλης «word» με την τιμή της στήλης όπου έγινε η συνένωση και στη συνέχεια ξανατρέξαμε τον κώδικα μεταφόρτωσης των λημμάτων από το αρχείο csv στην Οντολογία και τα λήμματα ήταν πλέον ξεχωριστά.

2. <u>Χρήση Ελληνικών Χαρακτήρων στα Ονόματα των Κλάσεων</u>
   Αρχικά αυτό που υπήρχε σαν στόχος ήταν τα ονόματα των κλάσεων να δημιουργηθούν με κώδικα σε μορφή «Ελληνικά/Αγγλικά», π.χ. «Μέρη_του_Λόγου/Parts_of_Speech», λόγω μιας σημαντικής λεπτομέρειας που υπήρχε στην διατριβή της κας Σαμαρείδη. Σε αυτό το κομμάτι αντιμετωπίσαμε πρόβλημα στο τρίτο μέρος του κώδικα. Συγκεκριμένα, όταν ορίζαμε τις κλάσεις στις οποίες θα έπρεπε να εισαχθούν τα μοντέλα, οι ελληνικοί χαρακτήρες δεν αναγνωρίζονταν από τον κώδικα, αλλά διαβάζονταν όπως παρακάτω:

#Î†Îº»Î¯Ï„Î¿_Î‘ÎÎ¸Î¡Î¼Î·Ï„Î¹ÎŒ/Non_Inflected_Numeral

Εικόνα 12: *Παράδειγμα για κλάση Άκλιτο_Αριθμητικό/Non_Inflected_Numeral*

   Έτσι, δώσαμε ονομασία μόνο σε Αγγλική Ορολογία και ο κώδικας έφερε τα σωστά αποτελέσματα. Στην συνέχεια, όμως, μέσω της πλατφόρμας Protégé, αλλάξαμε τις ονομασίες των κλάσεων και προσθέσαμε και το μέρος που υπολείπονταν στα ελληνικά.

3. <u>Σύγχυση μεταξύ ονομάτων των κλάσεων</u>

   Κατά την εκτέλεση του τρίτου μέρους του κώδικα, παρατηρήθηκε ότι τα λήμματα που θα έπρεπε να ανήκουν στις κλάσεις «Adjective», «Noun» και «Pronoun», εισάγονταν λανθασμένα στις κλάσεις «Non_Inflected_Adjective», «Non_Inflected_Noun» και «Non_Inflected_Relative_Pronoun» αντίστοιχα. Αυτό οφείλονταν στο ότι η εισαγωγή ενός λήμματος του αρχείο csv σε μια κλάση της Οντολογίας γίνονταν ως εξής:

Ο κώδικας διάβαζε στο LexikoOrganised.csv την τιμή της στήλης «PartsOfSpeech» και έψαχνε στην Οντολογία να βρει μια κλάση ίδια ή σχεδόν ίδια με την τιμή που είχε διαβάσει.

Για να διορθώσουμε το συγκεκριμένο λάθος, μετονομάσαμε τις κλάσεις «Adjective», «Noun» και «Pronoun» σε «Epitheto», «Ousiastiko» και «Antonumia» αντίστοιχα. Αφού δοκιμάσαμε και είδαμε ότι τα λήμματα είχαν πλέον εισαχθεί σωστά, από την πλατφόρμα Protégé επαναφέραμε τις ονομασίες όπως αναφέρουμε στο (2) του τρέχοντος κεφαλαίου.

6. ΣΥΜΠΕΡΑΣΜΑΤΑ

Συνοψίζοντας, η παρούσα εφαρμογή επιτυγχάνει σε έναν βαθμό τον σκοπό της μέσω κώδικα, έχοντας παράλληλα και ορισμένα μέρη τα οποία χρειάστηκαν την δική μας «χειροκίνητη» επέμβαση στο τελικό αποτέλεσμα.

Για μελλοντικές εφαρμογές κι ερευνητικές κατευθύνσεις, η παρούσα διπλωματική εργασία ανοίγει αρκετές δυνατότητες. Καταρχάς, η ίδια η διαδικασία εξόρυξης δεδομένων και η οντολογική δόμησή τους για τη Νέα Ελληνική Γλώσσα μπορεί να επεκταθεί με περαιτέρω ανάλυση κι ενσωμάτωση περισσότερων γλωσσικών χαρακτηριστικών, όπως φωνητικές ιδιότητες ή σύνθετες γραμματικές δομές. Επιπλέον, θα μπορούσε να ενσωματωθεί η δυναμική ενημέρωση της οντολογίας με νεότερα δεδομένα από πιο σύγχρονες πηγές ή ακόμα και από διαδικτυακές πλατφόρμες όπως τα social media, για την παρακολούθηση της εξέλιξης της γλώσσας σε πραγματικό χρόνο.

Μια άλλη ερευνητική κατεύθυνση θα ήταν η επέκταση της οντολογίας σε πολυγλωσσικά περιβάλλοντα, όπου τα δεδομένα από άλλες γλώσσες θα μπορούσαν να ενσωματωθούν και να συγκριθούν με τη Νέα Ελληνική Γλώσσα. Αυτό θα βοηθούσε στη μελέτη της διαγλωσσικής επιρροής και των πολιτισμικών διαφορών στη χρήση της γλώσσας. Τέλος, η ενσωμάτωση αυτών των δεδομένων σε συστήματα τεχνητής νοημοσύνης και μηχανικής μάθησης θα επέτρεπε τη δημιουργία πιο έξυπνων εργαλείων για γλωσσική ανάλυση και αυτόματη κατανόηση κειμένου.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. *Data Mining: Έννοια, Οφέλη και Τεχνικές* (2023), Big Blue Data Academy (https://bigblue.academy/gr/data-mining)
2. Han, J., Pei, J., & Kamber, M. (2011). *Data Mining: Concepts and Techniques* (3η έκδοση). Εκδ. οίκος Morgan Kaufmann.
3. Aggarwal, C. C. (2015). *Data Mining: The Textbook*. Εκδ. οίκος Springer.
4. Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar, επιστημονική επιμέλεια στα ελληνικά Βασίλειος Βερύκιος (2018) *Εισαγωγή στην Εξόρυξη Δεδομένων* (2η έκδοση). Εκδ. οίκος ΤΖΙΟΛΑ.
5. Διαδικτυακή πλατφόρμα του Πανεπιστημίου του Stanford για το Protégé (https://protege.stanford.edu/about.php)
6. Musen, M. A. (2015). *The Protégé Project: A Look Back and a Look Forward*. Περ. AI Matters, 1(4), 4-12.
7. Μανώλης Γεργατσούλης & Χρήστος Παπαθεοδώρου, *Εισαγωγή στις Οντολογίες και το Σημασιολογικό Ιστό*, Ομάδα Βάσεων Δεδομένων και Πληροφοριακών Συστημάτων, Τμήμα Αρχειονομίας – Βιβλιοθηκονομίας, Ιόνιο Πανεπιστήμιο
8. Gruber, T. R. (1993). *Toward principles for the design of ontologies used for knowledge sharing*. International Journal of Human-Computer Studies
9. Bell, G. & Dourish, P. (2007). *Yesterday's tomorrows: Notes on ubiquitous computing's dominant vision*. Personal and Ubiquitous Computing
10. Uschold, M., & King, M. (1995). *Towards a methodology for building ontologies*. Proceedings of IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing
11. Antoniou G., & van Harmelen F. (2008). *A Semantic Web Primer* (2η έκδοση). MIT Press.
12. Hitzler, P., Krötzsch, M., & Rudolph, S. (2010). *Foundations of Semantic Web Technologies*. Εκδ. οίκος Chapman and Hall/CRC.
13. Horridge, M., Bechhofer, S., & Noy, N. F. (2011). *The OWL API: A Java API for OWL Ontologies*. Περ. Semantic Web Journal, 2(1), 11-21.
14. Daniel L. Rubin, Natalya F. Noy, Mark A. Musen (2007), *Protégé: A Tool for Managing and Using Terminology in Radiology Applications, Journal Of Digital Imaging*, Πανεπιστήμιο Stanford (Πηγή: https://web.stanford.edu/group/rubinlab/pubs/Protege-RadiologyTerminology.pdf)
15. Μαρία Παπαφώτη (2009), *Ανάπτυξη Οντολογίας στο Protégé για την Αναπαράσταση Προϊόντων και Λειτουργιών Τραπεζικών Οργανισμών* (επιβλέπων καθηγητής: Νικόλαος Βασιλειάδης), Διατμηματικό Πρόγραμμα Μεταπτυχιακών Σπουδών «Πληροφορική & Διοίκηση», Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης.
16. Χρήστος Κοντζίνος, Δημήτριος Ασκούνης (2017), *Ανάλυση, Σύγκριση Και Χρήση Οντολογιών Ανθρώπινης Συμπεριφοράς Σε Πραγματικά Δεδομένα*, Άρτεμις +, Εθνικό Μετσόβιο Πολυτεχνείο.

17. Nikoletta E. Samaridi, Nikitas N. Karanikolas, Evangelos C. Papakitsos, and Michail Papoutsidakis. 2020. «*Designing a Greek Electronic Dictionary based on Ontology*». 24th Pan-Hellenic Conference on Informatics (PCI 2020), November 20–22, 2020, Athens, Greece. ACM, New York, NY, USA, pp 223–225, ACM ISBN: 978-1-4503-8897-9/20/11, https://doi.org/10.1145/3437120.3437311

18. Gómez-Pérez, A., Fernández-López, M., & Corcho, O. (2004). *Ontological Engineering: With Examples from the Areas of Knowledge Management, E-Commerce and the Semantic Web*. Εκδ. οίκος Springer.

19. Jurafsky, D., & Martin, J. H. (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (2η έκδοση). Εκδ. οίκος Prentice Hall.

20. Navigli, R. (2012). *A Quick Tour of Word Sense Disambiguation, Induction and Related Approaches. In Proceedings of the 2012 International Conference on Computational Linguistics* (pp. 13-25)

21. Fellbaum, C. (Ed.). (1998). *WordNet: An Electronic Lexical Database*. MIT Press

22. Bosque-Gil, J., Gracia, J., & Montiel-Ponsoda, E. (2016). *Ontology-based Multilingual Lexical Resources: The Lemon-Bib Model.* In Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016) (pp. 3846-3851).

23. Πύλη για την ελληνική γλώσσα, *Λεξικό Μανώλη Τριανταφυλλίδη σε ηλεκτρονική μορφή,* (https://www.greek-language.gr/greekLang/modern_greek/tools/lexica/triantafyllides/)

24. Sweigart, A. (2015). *Automate the Boring Stuff with Python*. No Starch Press.

25. Matthes, E. (2019). *Python Crash Course: A Hands-On, Project-Based Introduction to Programming (2η έκδοση).* No Starch Press.

26. Ramalho, L. (2015). *Fluent Python: Clear, Concise, and Effective Programming*. O'Reilly Media.

27. Lutz, M. (2013). *Learning Python (5η έκδοση).* O'Reilly Media.

28. Slatkin, B. (2015). *Effective Python: 90 Specific Ways to Write Better Python*. Addison-Wesley.

29. Gaddis, T. (2021). *Ξεκινώντας με την Python* (4η ελληνική έκδοση). Εκδόσεις Γκιούρδα

30. Python Software Foundation. (2023). *Python 3 Documentation*. (πηγή: https://docs.python.org/3/)

31. Real Python. (2023). *Real Python Tutorials*. (πηγή: https://realpython.com/)

32. Learn Python. (2023). *Learn Python Interactive Tutorials*. (πηγή: https://www.learnpython.org/)

33. Stack Overflow. (2023). *Python Tag Q&A.* (πηγή: https://stackoverflow.com/questions/tagged/python)

34. Geeks for Geeks. (2023). *Python Programming Language*. (πηγή: https://www.geeksforgeeks.org/python-programming-language/)

ΠΑΡΑΡΤΗΜΑΤΑ

Παράρτημα Κώδικα

Άντληση Λημμάτων και οργάνωση σε .csv

```
import requests
from bs4 import BeautifulSoup
import time

def web(url):
    n = 0
    page = requests.get(url, timeout=80)
    soup = BeautifulSoup(page.text, "html.parser")
    #for link in soup.findAll('dl'):
    for li in soup.findAll('dt'):
        #word =
li.find('b').get_text().replace(';','?')
        dt = li.get_text().replace(';','?')
        t1 = dt.find('[')
        t2 = dt.find(']')
        akt = dt.find(':')
        front = dt[0:akt-1]
        words = front.split(' ')
        rear = dt[akt+2:].replace('\n','')
        rear_words = rear.split(' ')

        if t1 < akt:
            word = dt[0:t1-1]
        else:
            word = dt[0:akt-1]
        #if word.startswith(('-')):
        #    word = "='"+word

        if t2 < akt:
            spelling = dt[t1+1:t2]
        else:
            spelling = ' '

        etymology = li.find('p')
        if etymology != None:
            etymology = etymology.get_text().replace(';','?').strip()
        else:
            etymology = ' '
        etym = dt.find(etymology,akt)

        if akt < etym:
            definition = dt[akt+1:etym].strip()
```

```
        else:
            definition = ' '

        if len(words)>= 2:
            if '(άκλ.' in front:
                if 'Ο (άκλ.)' in front:
                    myclass = 'Άκλιτο_ουσιαστικό'
                    link = 'Ο (άκλ.)'
                elif 'Ε (άκλ.)' in front or '(άκλ., ως επίθ.)' in
front:
                    myclass = 'Άκλιτο_επίθετο'
                    link = 'Ε (άκλ.)'
                elif 'επιφ. (άκλ.)' in front:
                    myclass = 'Επιφώνημα'
                    link = 'επιφ. (άκλ.)'
                elif 'αντων. αόρ. (άκλ.)' in front:
                    myclass = 'Άκλιτη_αόριστη_αντωνυμία'
                    link = 'αντων. αόρ. (άκλ.)'
                elif '(άκλ.) αντων. ερωτ.' in front:
                    myclass = 'Άκλιτη_ερωτηματική_αντωνυμία'
                    link = '(άκλ.) αντων. ερωτ.'
                elif 'αντων. αναφ. (άκλ.)' in front:
                    myclass = 'Άκλιτη_αναφορική_αντωνυμία'
                    link = 'αντων. αναφ. (άκλ.)'
                elif '(άκλ.) αριθμτ. επίθ. απόλ.' in front:
                    myclass = 'Άκλιτο_αριθμητικό'
                    link = '(άκλ.) αριθμτ. επίθ. απόλ.'
                elif '(άκλ., ως επίρρ.)' in front:
                    myclass = 'Επίρρημα'
                    link = '(άκλ., ως επίρρ.)'
                elif '(άκλ.) (ως επίρρ.)' in front:
                    myclass = 'Επίρρημα'
                    link = '(άκλ.) (ως επίρρ.)'
                else:
                    myclass = 'Άκλιτα_μέρη_του_λόγου'
                    link = '(άκλ.)'

            elif 'επίρρ.' in front:
                if 'επίρρ. χρον.' in front:
                    myclass = 'Χρονικό_επίρρημα'
                    link = 'επίρρ. χρον.'
                elif 'επίρρ. τοπ.' in front:
                    myclass = 'Τοπικό_επίρρημα'
                    link = 'επίρρ. τοπ.'
                elif 'επίρρ. τροπ.' in front:
                    myclass = 'Τροπικό_επίρρημα'
                    link = 'επίρρ. τροπ.'
                elif 'επίρρ. ποσ.' in front:
```

```
                    myclass = 'Ποσοτικό_επίρρημα'
                    link = 'επίρρ. ποσ.'
                elif 'επίρρ. βεβ.' in front:
                    myclass = 'Βεβαιωτικό_επίρρημα'
                    link = 'επίρρ.
βεβ.'
                elif '(ως επίρρ.)' in front:
                    myclass = 'Επίρρημα'
                    link = 'ως επίρρ.'
                else:
                    myclass = 'Επίρρημα'
                    link = 'επίρρ.'

            elif words[-1].startswith(('Ο','Ο')):
                myclass = 'Ουσιαστικό'
                link = words[-1]
            elif ' Ο' in front:
                myclass = 'Ουσιαστικό'
                for w in words:
                    if w.startswith('Ο') and w != words[0]:
                        link = w
                        break

            elif words[-1].startswith(('Ρ','Ρ')):
                myclass = 'Ρήμα'
                link = words[-1]
            elif ' Ρ' in front:
                myclass = 'Ρήμα'
                for w in words:
                    if w.startswith('Ρ') and w != words[0]:
                        link = w
                        break

            elif 'αντων.' in front:
                if 'αντων. δεικτ.' in front:
                    myclass = 'Δεικτική_αντωνυμία'
                    link = 'αντων. δεικτ.'
                elif 'αντων. προσ.' in front:
                    myclass = 'Προσωπική_αντωνυμία'
                    link = 'αντων. προσ.'
                elif 'αντων. αόρ.' in front:
                    myclass = 'Αόριστη_αντωνυμία'
                    link = 'αντων. αόρ.'
                elif 'αντων. αναφ.' in front:
                    myclass = 'Αναφορική_αντωνυμία'
                    link = 'αντων. αναφ.'
                elif 'αντων. αυτοπ.' in front:
                    myclass = 'Αυτοπαθητική_αντωνυμία'
```

```
                link = 'αντων. αυτοπ.'
            elif 'αντων. κτητ.' in front:
                myclass = 'Κτητική_αντωνυμία'
                link = 'αντων. κτητ.'
            elif 'αντων. ερωτ.' in front:
                myclass = 'Ερωτηματική_αντωνυμία'
                link = 'αντων. ερωτ.'
            else:
                myclass = 'Αντωνυμία'
                link = 'αντων.'

        elif 'πρόθ.' in front:
            myclass = 'Πρόθεση'
            link = 'πρόθ.'

        elif 'σύνδ.' in front:
            if 'σύνδ. υποθ.' in front:
                myclass = 'Υποθετικός_σύνδεσμος'
                link = 'σύνδ. υποθ.'
            elif 'σύνδ. τελ.' in front:
                myclass = 'Τελικός_σύνδεσμος'
                link = 'σύνδ. τελ.'
            elif 'σύνδ. διστ.' in front:
                myclass = 'Διστακτικός_σύνδεσμος'
                link = 'σύνδ. διστ.'
            elif 'σύνδ. ειδ.' in front:
                myclass = 'Ειδικός_σύνδεσμος'
                link = 'σύνδ. ειδ.'
            elif 'σύνδ. αντιθ.' in front:
                myclass = 'Αντιθετικός_σύνδεσμος'
                link = 'σύνδ. αντιθ.'
            elif 'σύνδ. ερωτ.' in front:
                myclass = 'Ερωτηματικός_σύνδεσμος'
                link = 'σύνδ. ερωτ.'
            elif 'σύνδ. αιτιολ.' in front:
                myclass = 'Αιτιολογικός_σύνδεσμος'
                link = 'σύνδ. αιτιολ.'
            elif 'σύνδ. διαχ.' in front:
                myclass = 'Διαχωριστικός_σύνδεσμος'
                link = 'σύνδ. διαχ.'
            elif 'σύνδ. συμπλεκτ.' in front:
                myclass = 'Συμπλεκτικός_σύνδεσμος'
                link = 'σύνδ. συμπλεκτ.'
            elif 'σύνδ. συμπερ.' in front:
                myclass = 'Συμπερασματικός_σύνδεσμος'
                link = 'σύνδ. συμπερ.'
            elif 'σύνδ. αποφατικός.' in front:
                myclass = 'Αποφατικός_σύνδεσμος'
```

```
                link = 'σύνδ. αποφατικός.'
            else:
                myclass = 'Σύνδεσμος'
                link = 'σύνδ.'

        elif 'επιφ.' in front:
            myclass = 'Επιφώνημα'
            link = 'επιφ.'

        elif 'επίθ.' in front:
            if 'αριθμτ. επίθ.' in front:
                myclass = 'Αριθμητικό_Επίθετο'
                link = 'αριθμτ. επίθ.'
            elif 'αριθμτ. επίθ. τακτ.' in front:
                myclass = 'Αριθμητικό_επίθετο_τακτικό'
                link = 'αριθμτ. επίθ. τακτ.'
            elif 'αριθμτ. επίθ. απόλ.' in front:
                myclass = 'Αριθμητικό_επίθετο_απόλυτο'
                link = 'αριθμτ. επίθ. απόλ.'
        elif 'αριθμτ. αναλ.' in front:
            myclass = 'Αριθμητικό_επίθετο_αναλογικό'
            link = 'αριθμτ. αναλ.'
        elif 'αριθμτ. τακτ.' in front:
            myclass = 'Αριθμητικό_επίθετο_τακτικό '
            link = 'αριθμτ. τακτ.'
        elif 'αριθμτ. πολλαπλ.' in front:
            myclass = 'Αριθμητικό_επίθετο_πολλαπλασιαστικό'
            link = 'αριθμτ. πολλαπλ.'
        elif 'αριθμτ. απολ.' in front:
            myclass = 'Αριθμητικό_επίθετο_απόλυτο'
            link = 'αριθμτ. απολ.'
        elif 'αριθμ. περιλ.' in front:
            myclass = 'Αριθμητικό_επίθετο_περιληπτικό'
            link = 'αριθμ. περιλ.'
        elif words[-1].startswith(('E','E')):
            myclass = 'Επίθετο'
            link = words[-1]
        elif ' E' in front:
            myclass = 'Επίθετο'
            for w in words:
                if w.startswith('E') and w != words[0]:
                    link = w
                    break
        elif ' -ή -ό' in front or ' -η -ο' in front:
            myclass = 'Επίθετο'
            link = 'E'

        elif 'αόριστο άρθρο' in front:
```

```
                myclass = 'Αόριστο_Άρθρο'
                link = 'αόριστο άρθρο'
            elif 'άρθρο οριστικό' in front:
                myclass = 'Οριστικό_Άρθρο'
                link = 'άρθρο οριστικό'

            else:
                myclass = ' '
                link =' '

        else:
            myclass = ' '
            link =' '
        n += 1

        male = ''               #αρσ. ή αρσ. &
        female = ''             #θηλ. ή θηλ. &
        antonym = ''            #ANT
        synonym = ''            #ΣΥΝ
        synonymPhrase = ''      #ΣΥΝ ΦΡ.
        diminutive = ''         #ΥΠΟΚΟΡ ή υποκορ.
        magnifying = ''         #ΜΕΓΕΘ ή μεγέθ.
        phrase = ''             #ΦΡ
        saying = ''             #ΠΑΡ ή ΠΑΡ ΦΡ ή ΠΑΡ έκφρ. ή γνωμ.

        #try:
        if ' θηλ. ' in front:
            female = words[words.index('θηλ.')+1]
            if female in ['&', 'και', '(προφ.)', '(λόγ.)',
'(λαϊκότρ.)'] :
                female = words[words.index('θηλ.')+2]
            if female == 'και':
                female = words[words.index('θηλ.')+3]
            if female == '(στη':
                female = words[words.index('θηλ.')+4]
            if female == '(συνήθ.':
                female = words[words.index('θηλ.')+5]

        if ' αρσ. ' in front:
            male = words[words.index('αρσ.')+1]
            if male=='(στη':
                male = words[words.index('αρσ.')+4]
            if male=='και':
                male = ''

        if ' ANT ' in rear:
            antonym = rear_words[rear_words.index('ANT')+1]
```

```
            if antonym  in [ '(του)', '~', 'Το', 'Μεγάλος', 'Ζωντανή',
'αβασίλευτη:', 'αγάνωτο.', 'αδειάζω*', 'ΦΡ', 'Σε' , 'σε' , 'στα',
'Απόκρυφα*'] :
                antonym = ''
            if antonym == 'αθωωτική.':
                antonym = 'αθωωτικός'
            if antonym == 'αποστρατιωτικοποίη':
                antonym = 'αποστρατιωτικοποίηση'
            if antonym == 'ωφέ':
                antonym = 'ωφέλιμος'
            if antonym == 'χειμωνιά':
                antonym = 'χειμωνιάτικος'
            if antonym == 'σχετικός1γ:' :
                antonym = 'σχετικός:'
            if antonym == 'ταμπλ':
                antonym = 'ταμπλ ντοτ'
            if antonym == 'τα':
                antonym = 'στεγνά'
            if antonym == 'με':
                antonym = 'με δόσεις'

        if ' ΣΥΝ ' in rear:
            synonym = rear_words[rear_words.index('ΣΥΝ')+1]
            if synonym in ['ΦΡ', 'τι']:
                synonym = ''
                s1 = rear_words.index('ΣΥΝ') #Το χρησιμοποιούμε σαν
πρώτη λέξη της φράσης που θα μπει σαν synonymPhrase
                synphrase = [] #Φτιάχνουμε μια λίστα για να πάρει την
συνώνυμη φράση και μετά θα καταχωρήσουμε την τιμή της στην string
'synonymPhrase'
                for var in rear_words[s1:] :
                    synphrase.append(var)
                    if '.' in var or ':' in var:
                        break

                synonymPhrase = " ".join(synphrase)
                synonymPhrase = synonymPhrase.replace('ΣΥΝ',
'').strip()

            #στις παρακάτω περιπτώσεις, ο κώδικας θα βρίσκει την πρώτη
τελεία στους τύπους που συμπεριλαμβάνονται στην if, άρα δεν θα φέρνει
την φράση που θέλουμε.Π.χ. θα φέρνει synonymPhrase = ΣΥΝ έκφρ.
            if synonym in ['έκφρ.', '(έκφρ.)', '(λόγ.)']:
                synonym = ''
                s1 = rear_words.index('ΣΥΝ') #Το χρησιμοποιούμε σαν
πρώτη λέξη της φράσης που θα μπει σαν synonymPhrase
```

```
                synphrase = [] #Φτιάχνουμε μια λίστα για να πάρει την
συνώνυμη φράση και μετά θα καταχωρήσουμε την τιμή της στην string
'synonymPhrase'
                period_count = 0 #Μεταβλητή για να μετράει τις τελείες.
Θέλουμε η φράση που θα επιλεχτεί να είναι αυτή μέχρι και την δεύτερη
τελεία ή την πρώτη άνω-κάτω τελεία.

                for var in rear_words[s1:] :
                    synphrase.append(var)
                    #Αν βρεις άνω-κάτω τελεία, σταμάτα
                    if ':' in var:
                        break
                    #Αν όχι, σταμάτα στην δεύτερη τελεία που θα βρεις
                    if '.' in var:
                        period_count +=1
                    if period_count == 2:
                        break

                synonymPhrase = " ".join(synphrase)
                synonymPhrase = synonymPhrase.replace('ΣΥΝ',
'').strip()
                if synonymPhrase == 'έκφρ. κτ.':
                    synonymPhrase = 'έκφρ. κτ. φτάνει στ΄ αυτιά μου'
                if synonymPhrase == 'ΦΡ κτ.':
                    synonymPhrase = 'ΦΡ κτ. τρέχει στα γύφτικα'
                if synonymPhrase == 'έκφρ. παίρνω κτ.':
                    synonymPhrase = 'έκφρ. παίρνω κτ. βαριά'
                if synonymPhrase == 'έκφρ. παίζω κτ.':
                    synonymPhrase = 'έκφρ. παίζω κτ. στα ζάρια'
                if synonymPhrase == 'ΦΡ δένω κτ.':
                    synonymPhrase = 'ΦΡ δένω κτ. κόμπο'
                if synonymPhrase == 'ΦΡ κάνω κτ.':
                    synonymPhrase = 'ΦΡ κάνω κτ. βούκινο'

            if synonym == 'ΕΠΙΡΡ':
                synonym = ''
                synonymPhrase = 'καλού κακού' #ειδική περίπτωση


        if ' ΥΠΟΚΟΡ ' in rear:
            diminutive = rear_words[rear_words.index('ΥΠΟΚΟΡ')-1]
            if len(diminutive) <= 2:
                diminutive = diminutive + ' ' +
rear_words[rear_words.index('ΥΠΟΚΟΡ')-2]
            if "-" in diminutive:
                d1 = rear_words[rear_words.index('ΥΠΟΚΟΡ')-3]
                d2 = rear_words[rear_words.index('ΥΠΟΚΟΡ')-2]
                d3 = rear_words[rear_words.index('ΥΠΟΚΟΡ')-1]
```

```
                diminutive = d1 + ' ' + d2 + ' ' + d3
            if diminutive == 'το κι':
                d1 = rear_words[rear_words.index('ΥΠΟΚΟΡ')-1]
                d2 = rear_words[rear_words.index('ΥΠΟΚΟΡ')-3]
                d3 = rear_words[rear_words.index('ΥΠΟΚΟΡ')-2]
                diminutive = d1 + ' ' + d2 + d3
            if diminutive == '(μειωτ.)':
                d1 = rear_words[rear_words.index('ΥΠΟΚΟΡ')-2]
                d2 = rear_words[rear_words.index('ΥΠΟΚΟΡ')-3]
                diminutive = d1 + ' ' + d2
            if diminutive == 'ΕΠΙΡΡ': #ΕΠ+αγγλικό Ι+ΡΡ.
                diminutive = 'αργούτσικος -η -ο'
                #συμβαίνει μόνο με το λήμμα 'αργός -η -ο' γιατί το
ΥΠΟΚΟΡ υπάρχει ως 'ΥΠΟΚΟΡ.' και δεν αναγνωρίζεται από τον κώδικα.
Αντίθετα αναγνωρίζεται το ΕΠΙΡΡ που υπάρχει πιο κάτω πριν από 'ΥΠΟΚΟΡ'.
            if diminutive == 'λικος -η -ο':
                diminutive = 'νοστιμούλικος -η -ο'

        #if ' υποκορ. ' in rear:
        #    diminutive = rear_words[rear_words.index('υποκορ.')-1]
        #    if len(diminutive) <= 2:
        #        diminutive = diminutive + ' ' +
rear_words[rear_words.index('υποκορ.')-2]

        #if ' ΜΕ ΓΕΘ 'in rear:
        #    magnifying = rear_words[rear_words.index('ΜΕ ΓΕΘ')-1]
        #    if len(magnifying) <= 2:
        #        magnifying = magnifying + ' ' +
rear_words[rear_words.index('ΜΕ ΓΕΘ')-2]

        if ' ΜΕΓΕΘ ' in rear:
            magnifying = rear_words[rear_words.index('ΜΕΓΕΘ')-1]
            if len(magnifying) <= 2:
                magnifying = magnifying + ' ' +
rear_words[rear_words.index('ΜΕΓΕΘ')-2]

        if ' ΜΕΓΕΘ. ' in rear:
            magnifying = rear_words[rear_words.index('ΜΕΓΕΘ.')-1]
            if len(magnifying) <= 2:
                magnifying = magnifying + ' ' +
rear_words[rear_words.index('ΜΕΓΕΘ.')-2]

        #Ειδικές περιπτώσεις που ξεφεύγουν του γενικού κανόνα
        if magnifying == 'η νάρα':
            magnifying = 'η μπαρμπουνάρα'
        if magnifying == 'κλεφταρού':
            magnifying = 'ο κλεφταράς'
        if magnifying == 'εγωισταρού':
```

```
            magnifying = 'ο εγωισταράς'
        if magnifying == 'ψευτρού':
            magnifying = 'ο ψευταράς'
        if magnifying == 'προφ.)':
            magnifying = 'περιπτωσάρα η (οικ., προφ.)'
        if magnifying == '1. ο':
            magnifying = 'ο πόρδος'


        #if ' μεγεθ. ' in rear:
        #    magnifying = rear_words[rear_words.index('μεγεθ.')-1]
        #    if len(magnifying) <= 2:
        #        magnifying = magnifying + ' ' +
rear_words[rear_words.index('μεγεθ.')-2]

        #PHRASE

        if ' ΦΡ ' in rear:
            if (rear_words[rear_words.index('ΦΡ')-1] != ' ΠΑΡ ') and
(rear_words[rear_words.index('ΦΡ')-1] != ' ΣΥΝ ') and
(rear_words[rear_words.index('ΦΡ')-1] != ' ΣΥΝ '):
                ph1 = rear_words.index('ΦΡ')
                ph2 = []
                for var2 in rear_words[ph1:] :
                    ph2.append(var2)
                    if '.' in var2 or '·' in var2 or 'ΠΑΡ' in var2:
                        break
                    phrase = " ".join(ph2)
                    #phrase = phrase.replace('ΦΡ', '').strip()

        if ' έκφρ. ' in rear:
            if rear_words[rear_words.index('έκφρ.')-1] != ' ΠΑΡ ':
                if (rear_words[rear_words.index('έκφρ.')-1] != ' ΣΥΝ ')
and (rear_words[rear_words.index('έκφρ.')-1] != ' ΣΥΝ '):
                    ph1 = rear_words.index('έκφρ.')
                    ph2 = []
                    for var2 in rear_words[ph1:] :
                        ph2.append(var2)
                        if '.' in var2 or '·' in var2 or 'ΠΑΡ' in var2:
                            break
                        phrase = " ".join(ph2)
                        #phrase = phrase.replace('έκφρ.', '').strip()

        if ' (έκφρ.) ' in rear:
            if rear_words[rear_words.index('(έκφρ.)')-1] != ' ΠΑΡ ':
                if (rear_words[rear_words.index('(έκφρ.)')-1] != ' ΣΥΝ
') and (rear_words[rear_words.index('(έκφρ.)')-1] != ' ΣΥΝ '):
                    ph1 = rear_words.index('(έκφρ.)')
```

```
                    ph2 = []
                    for var2 in rear_words[ph1:] :
                        ph2.append(var2)
                        if '.' in var2 or '·' in var2 or 'ΠΑΡ' in var2:
                            break
                        phrase = " ".join(ph2)
                        #phrase = phrase.replace('(έκφρ.)', '').strip()

        #SAYING

        if ' ΠΑΡ ' in rear:
            sa1 = rear_words.index('ΠΑΡ')
            sa2 = []
            for var3 in rear_words[sa1:] :
                sa2.append(var3)
                if '.' in var3 or '·' in var3:
                    break
                saying = " ".join(sa2)
                #saying = saying.replace('ΠΑΡ', '').strip()


        #except ValueError as ve:
        #    print(f'{words[0]}:{ve}')

        dt = dt.replace('\n','')
        file1.write(f'{words[0]};{word};{spelling};{etymology[1:-
1]};{link};{myclass};{definition};{dt};{male};{female};{antonym};{synon
ym};{synonymPhrase};{diminutive};{magnifying};{phrase};{saying}\n')

        if 'μπε. του' in dt:
            myclass = 'Μετοχή_παθητικού_ενεστώτα'
            link ='μπε. του'
            n += 1
            file1.write(f'{words[0]};{word};{spelling};{etymology[1:-
1]};{link};{myclass};{definition};{dt}\n')
        if 'μππ. του' in dt:
            myclass = 'Μετοχή_παθητικού_παρακειμένου'
            link ='μππ. του'
            n += 1
            file1.write(f'{words[0]};{word};{spelling};{etymology[1:-
1]};{link};{myclass};{definition};{dt}\n')

        if 'ΕΠΙΡΡ' in dt:
            myclass = 'Επίρρημα'
            link ='επίρρ.'
            n += 1
            if 'ΕΠΙΡΡ:' in dt:
```

```
                file1.write(f'{rear_words[rear_words.index("ΕΠΙΡΡ:")-
1]};{rear_words[rear_words.index("ΕΠΙΡΡ:")-
1]};;;{link};{myclass};;{dt}\n')
            elif 'ΕΠΙΡΡ.' in dt:
                file1.write(f'{rear_words[rear_words.index("ΕΠΙΡΡ.")-
1]};{rear_words[rear_words.index("ΕΠΙΡΡ.")-
1]};;;{link};{myclass};;{dt}\n')
            else:
                file1.write(f'{rear_words[rear_words.index("ΕΠΙΡΡ")-
1]};{rear_words[rear_words.index("ΕΠΙΡΡ")-
1]};;;{link};{myclass};;{dt}\n')

    return n

start = time.time()

file1 = open('LexikoOrganised.csv', 'w', encoding='utf-8-sig')
file1.write('word;words;spelling;etymology;category;PartsOfSpeech;defin
ition;annotation;male;female;antonym;synonym;synonymPhrase;diminutive;m
agnifying;phrase;saying\n')

N = 0
letters =
['Α','Β','Γ','Δ','Ε','Ζ','Η','Θ','Ι','Κ','Λ','Μ','Ν','Ξ','Ο','Π','Ρ','Σ
','Τ','Υ','Φ','Χ','Ψ','Ω']
#letters = ['Ω']
for letter in letters:
    print(letter)
    # Fetch all the HTML source from the url
    response = requests.get(f'https://www.greek-
language.gr/greekLang/modern_greek/tools/lexica/triantafyllides/search.
html?start=0&lq={letter}')

    soup = BeautifulSoup(response.text, 'html.parser')
    items = soup.find('td',id='found').get_text().split(' ')
    items = int(items[0].replace('.',''))
    decades = items // 10 + 1
    for i in range(decades):
        N = N + web(f'https://www.greek-
language.gr/greekLang/modern_greek/tools/lexica/triantafyllides/search.
html?start={i*10}&lq={letter}')

print(f'\nΣύνολο {N} λέξεις')

file1.close()
end = time.time()
print("Time taken in seconds : ", (end-start))
```

Δημιουργία της Οντολογίας PartsOfSpeech στο Protégé

```
from owlready2 import *

#Δημιουργία νέου OWL αρχείου
onto = get_ontology("http://example.org/MeriTouLogou.owl")

with onto:
    # Δημιουργία της κύριας κλάσης

    class MeriTouLogou(Thing): pass

    #Δημιουργία Υποκλάσεων της MeriTouLogou

    class Aklita_Meri_tou_Logou(MeriTouLogou): pass
    class Klita_Meri_tou_Logou(MeriTouLogou): pass
    class Blank(MeriTouLogou): pass

    #Δημιουργία Υποκλάσεων της Aklita_Meri_tou_Logou

    class Akliti_Metochi(Aklita_Meri_tou_Logou): pass
    class Aklitoi_Onomastikoi_Typoi(Aklita_Meri_tou_Logou): pass
    class Aparemfato(Aklita_Meri_tou_Logou): pass
    class Epirima(Aklita_Meri_tou_Logou): pass
    class Morio(Aklita_Meri_tou_Logou): pass
    class Prothesi(Aklita_Meri_tou_Logou): pass
    class Syndesmos(Aklita_Meri_tou_Logou): pass
    class Epiphonima(Aklita_Meri_tou_Logou): pass

    #Δημιουργία Υποκλάσεων των Υποκλάσεων της Aklita_Meri_tou_Logou

    #Aklitoi_Onomastikoi_Typoi(Aklita_Meri_tou_Logou)

    class Akliti_anaforiki_antonymia(Aklitoi_Onomastikoi_Typoi): pass
    class Akliti_aoristi_antonymia(Aklitoi_Onomastikoi_Typoi): pass
    class Akliti_erotimatiki_antonymia(Aklitoi_Onomastikoi_Typoi): pass
    class Aklito_arithmitiko(Aklitoi_Onomastikoi_Typoi): pass
    class Aklito_epitheto(Aklitoi_Onomastikoi_Typoi): pass
    class Aklito_Ousiastiko(Aklitoi_Onomastikoi_Typoi): pass

    #Epirima(Aklita_Meri_tou_Logou)

    class Chroniko_Epirima(Epirima): pass
    class Posotiko_Epirima(Epirima): pass
    class Topiko_Epirima(Epirima): pass
    class Tropiko_Epirima(Epirima): pass
    class Veveotiko_Epirima(Epirima): pass

    #Syndesmos(Aklita_Meri_tou_Logou)
```

```
    class Aitiologikos_Syndesmos(Syndesmos): pass
    class Antithetikos_Syndesmos(Syndesmos): pass
    class Diachoristikos_Syndesmos(Syndesmos): pass
    class Distaktikos_Syndesmos(Syndesmos): pass
    class Eidikos_Syndesmos(Syndesmos): pass
    class Erotimatikos_Syndesmos(Syndesmos): pass
    class Symperasmatikos_Syndesmos(Syndesmos): pass
    class Symplektikos_Syndesmos(Syndesmos): pass
    class Ypothetikos_Syndesmos(Syndesmos): pass

    #Δημιουργία Υποκλάσεων της Klita_Meri_tou_Logou

    class Ptotika_Meri_tou_Logou(Klita_Meri_tou_Logou): pass
    class Rima(Klita_Meri_tou_Logou): pass

    #Δημιουργία Υποκλάσεων της
Ptotika_Meri_tou_Logou(Klita_Meri_tou_Logou)
    class Arthro(Ptotika_Meri_tou_Logou): pass
    class Antonymia(Ptotika_Meri_tou_Logou): pass
    class Kliti_Metochi(Ptotika_Meri_tou_Logou): pass
    class Onoma(Ptotika_Meri_tou_Logou): pass
    class Parepomena_twn_Ptotikwn(Ptotika_Meri_tou_Logou): pass

    #Δημιουργία Υποκλάσεων της Arthro(Ptotika_Meri_tou_Logou)
    class Aoristo_Arthro(Arthro): pass
    class Oristiko_Arthro(Arthro): pass

    #Δημιουργία Υποκλάσεων της Antonymia(Ptotika_Meri_tou_Logou)
    class Anaforiki_Antonymia(Antonymia): pass
    class Aoristi_Antonymia(Antonymia): pass
    class Deiktiki_Antonymia(Antonymia): pass
    class Erotimatiki_Antonymia(Antonymia): pass
    class Ktitiki_Antonymia(Antonymia): pass
    class Prosopiki_Antonymia(Antonymia): pass

    #Δημιουργία Υποκλάσεων της Kliti_Metochi(Ptotika_Meri_tou_Logou)
    class Metochi_Pathitikou_Enestota(Kliti_Metochi): pass
    class Metochi_Pathitikou_Parakeimenou(Kliti_Metochi): pass

    #Δημιουργία Υποκλάσεων της Onoma(Ptotika_Meri_tou_Logou)
    class Arithmitiko(Onoma): pass
    class Epitheto(Onoma): pass
    class Ousiastiko(Onoma): pass

    #Δημιουργία Υποκλάσεων της Epitheto(Onoma)
    class Arithmitiko_Epitheto(Epitheto): pass
```

```
    #Δημιουργία Υποκλάσεων της Arithmitiko Epitheto(Epitheto)
    class Arithmitiko_epitheto_analogiko(Arithmitiko_Epitheto): pass
    class Arithmitiko_epitheto_pollaplasiastiko(Arithmitiko_Epitheto):
pass
    class Arithmitiko_epitheto_taktiko(Arithmitiko_Epitheto): pass

    #Δημιουργία DataProperties
    class differentFormOfFemale(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class differentFormOfMale(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class differentFormOfLemma(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasAntonyms(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasAsFemale(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasAsMale(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasDiminutive(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasEtymology(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasGender(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasInflection(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasMagnyfying(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasSynonyms(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasSynonymousPhrase(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class hasValue(DataProperty):
        domain = [MeriTouLogou]
```

```
        range = [str]
    class hasVocalTranscription(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class inflectionalCategoryTriantafyllides(DataProperty):
        domain = [MeriTouLogou]
        range = [str]
    class isFemale(DataProperty):
        domain = [MeriTouLogou]
        range = [bool]
    class isMale(DataProperty):
        domain = [MeriTouLogou]
        range = [bool]
    class isNeutral(DataProperty):
        domain = [MeriTouLogou]
        range = [bool]
    class isPlural(DataProperty):
        domain = [MeriTouLogou]
        range = [bool]
    class isSingular(DataProperty):
        domain = [MeriTouLogou]
        range = [bool]

    #Δημιουργία AnnotationProperties

    class definition(AnnotationProperty): pass
    class
definitionFromDictionaryOfModernStandardGreek(AnnotationProperty): pass
    class example(AnnotationProperty): pass
    class expression(AnnotationProperty): pass
    class
expressionFromDictionaryOfModernStandardGreek(AnnotationProperty): pass
    class inflection(AnnotationProperty): pass
    class inflectionParadigmsTriantafyllides(AnnotationProperty): pass
    class isDefinedBy(AnnotationProperty): pass
    class label(AnnotationProperty): pass
    class morphologicalVariety(AnnotationProperty): pass
    class orthography(AnnotationProperty): pass
    class phrase(AnnotationProperty): pass
    class
phraseFromDictionaryofModernStandardGreek(AnnotationProperty): pass
    class pronunciation(AnnotationProperty): pass
    class saying(AnnotationProperty): pass
    class
sayingFromDictionaryofModernStandardGreek(AnnotationProperty): pass
    class seeAlso(AnnotationProperty): pass
    class semantics(AnnotationProperty): pass
    class style(AnnotationProperty): pass
```

```
    class syntax(AnnotationProperty): pass
    class typeFormation(AnnotationProperty): pass
    class image(AnnotationProperty): pass


# Αποθήκευση της οντολογίας σε αρχείο OWL
onto.save(file="MeriTouLogou.owl", format="rdfxml")
```

Φόρτωση των Λημμάτων του .csv στην PartsOfSpeech (Protégé)

```
import csv
from owlready2 import *

# Φόρτωση της οντολογίας
onto =
get_ontology("C:/Users/503344134.HCAD/.vscode/Protege/NewWay/MeriTouLog
ou2.owl").load()

# Συνάρτηση για να προσθέσουμε individual στην κατάλληλη κλάση και να
καταχωρήσουμε τις ιδιότητες
def add_individual(row):
    word = row['words']
    part_of_speech = row['PartsOfSpeech']

    # Βρίσκουμε την κατάλληλη κλάση
    cls = onto.search_one(iri="*"+part_of_speech)
    if cls is None:
        print(f"Η κλάση '{part_of_speech}' δεν βρέθηκε στην
οντολογία.")
        return

    # Δημιουργούμε ένα νέο individual
    individual = cls(word.replace(" ", "_"))
    print(f"Δημιουργήθηκε το άτομο: {individual}")

    # Συνάρτηση για να ορίσουμε data properties
    def set_data_property(individual, prop_name, value):
        prop = getattr(individual, prop_name, None)
        if prop is None:
            print(f"Η ιδιότητα '{prop_name}' δεν υπάρχει ή δεν είναι
σωστά ορισμένη.")
            return
        if isinstance(prop, list):  # Αν είναι λίστα, προσθέτουμε την
τιμή
            prop.append(value)
        else:  # Αλλιώς, αναθέτουμε απευθείας την τιμή
            setattr(individual, prop_name, value)
        print(f"Ορίστηκε η τιμή '{value}' στην ιδιότητα
'{prop_name}'.")
```

```
    # Συνάρτηση για να ορίσουμε annotation properties
    def set_annotation_property(individual, prop_name, value):
        prop = getattr(individual, prop_name, None)
        if prop is None:
            print(f"Η ιδιότητα '{prop_name}' δεν υπάρχει ή δεν είναι
σωστά ορισμένη.")
            return
        setattr(individual, prop_name, value)
        print(f"Ορίστηκε η τιμή '{value}' στην ιδιότητα
'{prop_name}'.")

    # Ασφαλής προσθήκη ιδιοτήτων
    try:
        # Χειρισμός Data Properties
        if row['spelling']:
            set_data_property(individual, 'hasVocalTranscription',
row['spelling'])
        if row['etymology']:
            set_data_property(individual, 'hasEtymology',
row['etymology'])
        if row['category']:
            set_data_property(individual,
'inflectionalCategoryTriantafyllides', row['category'])
        if row['definition']:
            set_annotation_property(individual,
'definitionFromDictionaryOfModernStandardGreek', row['definition'])
        if row['annotation']:
            set_annotation_property(individual, 'hasAnnotation',
row['annotation'])
        if row['male']:
            set_data_property(individual, 'hasAsMale', row['male'])
        if row['female']:
            set_data_property(individual, 'hasAsFemale', row['female'])
        if row['antonym']:
            set_data_property(individual, 'hasAntonyms',
row['antonym'])
        if row['synonym']:
            set_data_property(individual, 'hasSynonyms',
row['synonym'])
        if row['diminutive']:
            set_data_property(individual, 'hasDiminutive',
row['diminutive'])
        if row['magnifying']:
            set_data_property(individual, 'hasMagnifying',
row['magnifying'])
        if row['phrase']:
            set_data_property(individual, 'hasPhrase', row['phrase'])
```

```
    except Exception as e:
        print(f"Παρουσιάστηκε σφάλμα: {e}")

# Διαβάζουμε το αρχείο CSV και προσθέτουμε τα individuals
with
open('C:/Users/503344134.HCAD/.vscode/Protege/NewWay/LexikoOrganised1.c
sv', newline='', encoding='utf-8-sig') as csvfile:
    reader = csv.DictReader(csvfile, delimiter=';')
    for row in reader:
        add_individual(row)

# Αποθηκεύουμε την ενημερωμένη οντολογία σε αρχείο OWL
onto.save(file="C:/Users/503344134.HCAD/.vscode/Protege/NewWay/MeriTouL
ogou2_updated.owl", format="rdfxml")
```