ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
Τμήμα Πληροφορικής

Εργασία Μαθήματος ***Επεξεργασία Φυσικής Γλώσσας***

Αριθμός εργασίας – Τίτλος εργασίας	***Απαλλακτική εργασία Ιουνίου-Σεπτεμβρίου 2022***
Όνομα φοιτητή	Καλλίγερος Αναστάσης
Αρ. Μητρώου	Π19253
Ημερομηνία παράδοσης	18/7/2022

Εκφώνηση εργασίας

Β.2 Λύση σε άλλη γλώσσα προγραμματισμού

Εναλλακτικά μπορείτε να χρησιμοποιήσετε άλλες γλώσσες προγραμματισμού ή libraries τον κώδικα και την λειτουργικότητα των οποίων πρέπει να τεκμηριώσετε πειστικά και κατανοητά σύμφωνα με τα παρακάτω :

Θέμα 1ο (20 μονάδες)

Ανάπτυξη Λεκτικού Αναλυτή σε άλλη γλώσσα Προγραμματισμού. Αναζητείστε στο διαδίκτυο ή αναπτύξτε εσείς Λεκτικό Αναλυτή που διαβάζει μια μικρή ιστορία και μπορεί να παράγει μια λίστα από προτάσεις, κάθε μια από τις οποίες περιέχει μια λίστα από λέξεις. Τεκμηριώστε πειστικά τον κώδικά σας.

Θέμα 2ο (20 μονάδες)

Ανάπτυξη Συντακτικού Αναλυτή σε άλλη γλώσσα Προγραμματισμού. Αναζητείστε στο διαδίκτυο ή αναπτύξτε εσείς Συντακτικό Αναλυτή που με βάση τους κανόνες συντακτικής ανάλυσης της πρότυπης λύσης σε Prolog που σας δόθηκε παράγει το συντακτικό δένδρο της πρότασης. Τεκμηριώστε πειστικά τον κώδικά σας.

Θέμα 3ο (30 μονάδες)

Ανάπτυξη Σημασιολογικού Αναλυτή σε άλλη γλώσσα Προγραμματισμού. Αναζητείστε στο διαδίκτυο ή αναπτύξτε εσείς Σημασιολογικό Αναλυτή που με βάση τους κανόνες σημασιολογικής ανάλυσης της πρότυπης λύσης σε Prolog που σας δόθηκε παράγει τα σημαινόμενα της πρότασης (σχέσεις μεταξύ ρημάτων, ουσιαστικών, επιθέτων, κ.λπ). Τεκμηριώστε πειστικά τον κώδικά σας.

Θέμα 4ο (30 μονάδες)

Πρόγραμμα στην γλώσσα προγραμματισμού της επιλογής σας, για την ενημέρωση και πραγματοποίηση ερωταποκρίσεων σε Βάση Γνώσης που έχετε αναπτύξει γιαα αυτό τον σκοπό. Οι ερωτήσεις και απαντήσεις θα δίδονται σε φυσική γλώσσα.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1 Εισαγωγή 4
2 Περιγραφή του προγράμματος 4
3 Επίδειξη της λύσης 6
4 Βιβλιογραφικές Πηγές 13

1 Εισαγωγή

Αρχικά, μετά από προσεκτική μελέτη της εκφώνησης αποφάσισα να υλοποιήσω την εργασία επιλέγοντας το Β2.Δηλαδή να λύσω τα 4 θέματα που περιλαμβάνονται στο Β μέρος της εκφώνησης χρησιμοποιώντας άλλη γλώσσα προγραμματισμού και συγκεκριμένα την python.Επίσης ήταν αναγκαίο για την λειτουργικότητα του κάθε κώδικα να γίνουν install κάποιες libraries όπως η nltk. Ακόμη για την δημιουργία Βάσης Γνώσης χρησιμοποιήθηκε η SQLite όσον αφορά το 4ο θέμα. Τέλος ως εργαλείο ανάπτυξης του κάθε κώδικα επέλεξα την πιο νέα και πληρέστερη έκδοση του Visual Studio Code.Παρακάτω γίνεται αναλυτική παρουσίαση των θεμάτων με χαρακτηριστικά screenshots για την επίδειξη των λύσεων.

2 Περιγραφή του προγράμματος

Θέμα 1ο (20 μονάδες)

Για το πρώτο θέμα έπρεπε να φτιάξουμε λεκτικό αναλυτή ώστε διαβάζοντας μια μικρή ιστορία να μπορεί να παραχθεί μια λίστα από προτάσεις και στη συνέχεια μια λίστα από λέξεις που περιέχουν αυτές οι προτάσεις. Αρχικά κάναμε import το nltk που αποτελεί μια εργαλειοθήκη για την επεξεργασία φυσικής γλώσσας(natural language toolbox kit). Από αυτή την εργαλειοθήκη κάναμε import τα sent_tokenize και word_tokenize που χωρίζουν το αρχείο randomText.txt σε προτάσεις και λέξεις αντίστοιχα. Έτσι κάνουμε print τις προτάσεις και στη συνέχεια τις λέξεις κάθε πρότασης του αρχείου randomText.txt.

Θέμα 2ο (20 μονάδες)

Στο δεύτερο θέμα καλούμαστε να δημιουργήσουμε έναν συντακτικό αναλυτή που θα παράγει συντακτικό δέντρο. Όπως και στο πρώτο ερώτημα κάνουμε import το πακέτο nltk καθώς και τα sent_tokenize, word_tokenize και PunktSentenceTokenizer που ουσιαστικά διαιρεί το .txt αρχείο σε μια λίστα από προτάσεις. Επίσης συμπεριέλαβα σαν σχόλιο μέσα στον κώδικα την λίστα με τα tags που αντιστοιχούν σε κάθε μέρος του λόγου. Στη συνέχεια με την συνάρτηση process_content κάνω print κάθε λέξη μέσα στο randomText.txt μαζί με την συντακτική σημασία κάθε λέξης. Τέλος με την γραμμή chunked.draw() κάνουμε print το συντακτικό δέντρο προσθέτοντας κάθε πρόταση.

Θέμα 3ο (30 μονάδες)

Για τον σημασιολογικό αναλυτή χρησιμοποίησα 2 .txt αρχεία τα οποία περιείχαν θετικές και αρνητικές κριτικές ταινιών(negative.txt, positive.txt). Στη συνέχεια όρισα να λαμβάνω μόνο τα μέρη του λόγου που έιναι ρήματα, επίθετα και επιρρήματα για να περιορίσω λίγο τα δεδομένα και να λειτουργήσουν αποδοτικότερα οι αλγόριθμοι. Για την χρήση των αλγορίθμων έκανα import το sklearn που είναι ένα package που περιέχει διάφορους αλγόριθμους και έκανα train και test τα διάφορα δεδομένα. Έπειτα εμφανίζω τις λέξεις (ρήμα, επίθετο ή επίρρημα) που εμφανίζονται περισσότερες φορές στα θετικά σε σχέση με τα αρνητικά και το αντίστροφο.

Θέμα 4ο (30 μονάδες)

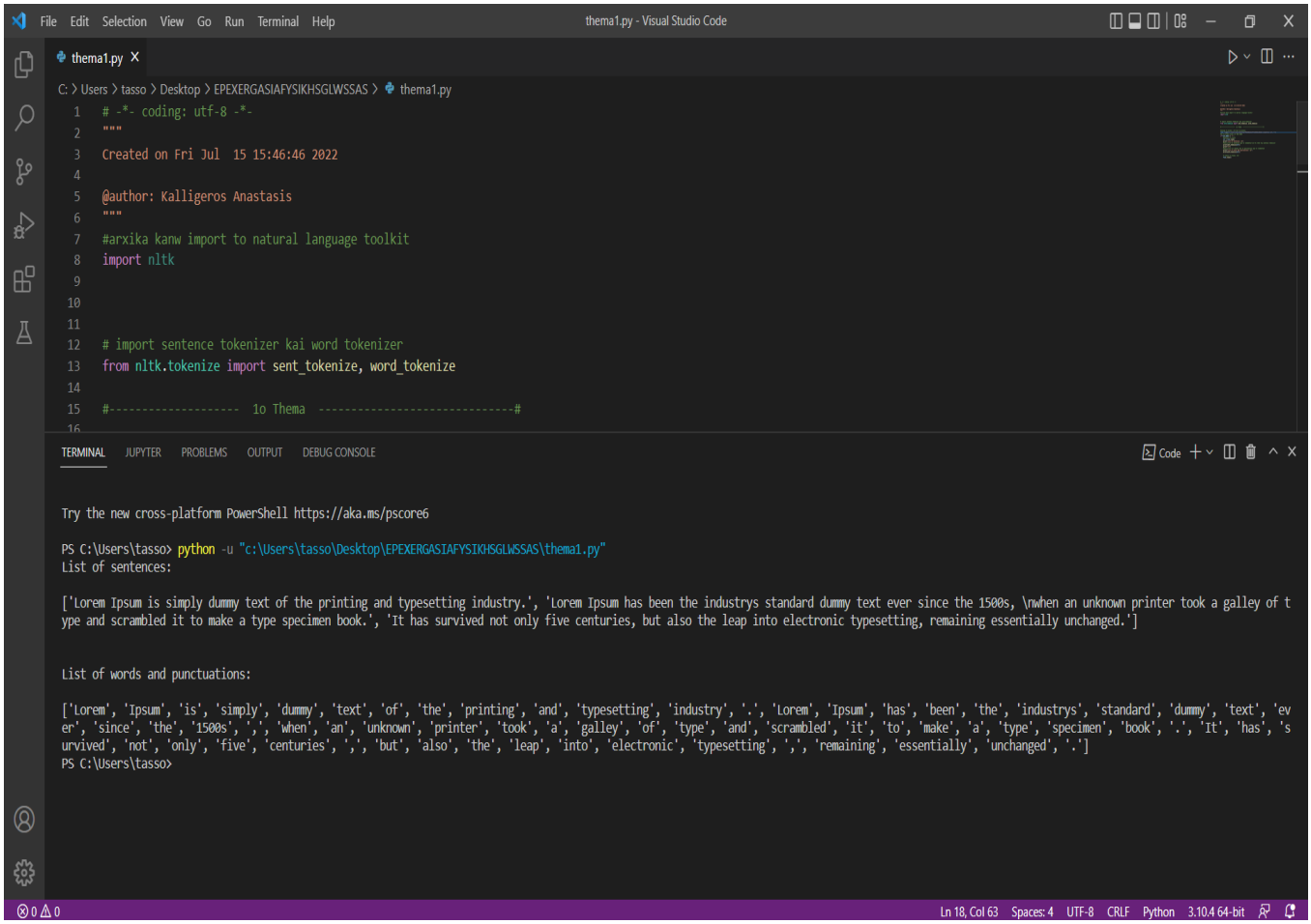
Το πρώτο πράγμα που έκανα είναι να κατασκευάσω μια βάση γνώσης. Η βάση γνώσης είναι στην ουσία μια Βάση δεδομένων (Database) που περιέχει πίνακες στους οποίους είναι καταχωρημένη όλη η σχετική γνώση. Για το παράδειγμα μας λοιπόν κατασκευάζουμε μια βάση (σε sqlite) που την ονομάζουμε Knowledge και καταχωρούμε την γνώση σε ένα πίνακα της βάσης που το ονομάζουμε knowledge.

3 Επίδειξη της λύσης

. Θέμα 1ο (20 μονάδες)

Δημιουργία Λεκτικού Αναλυτή:

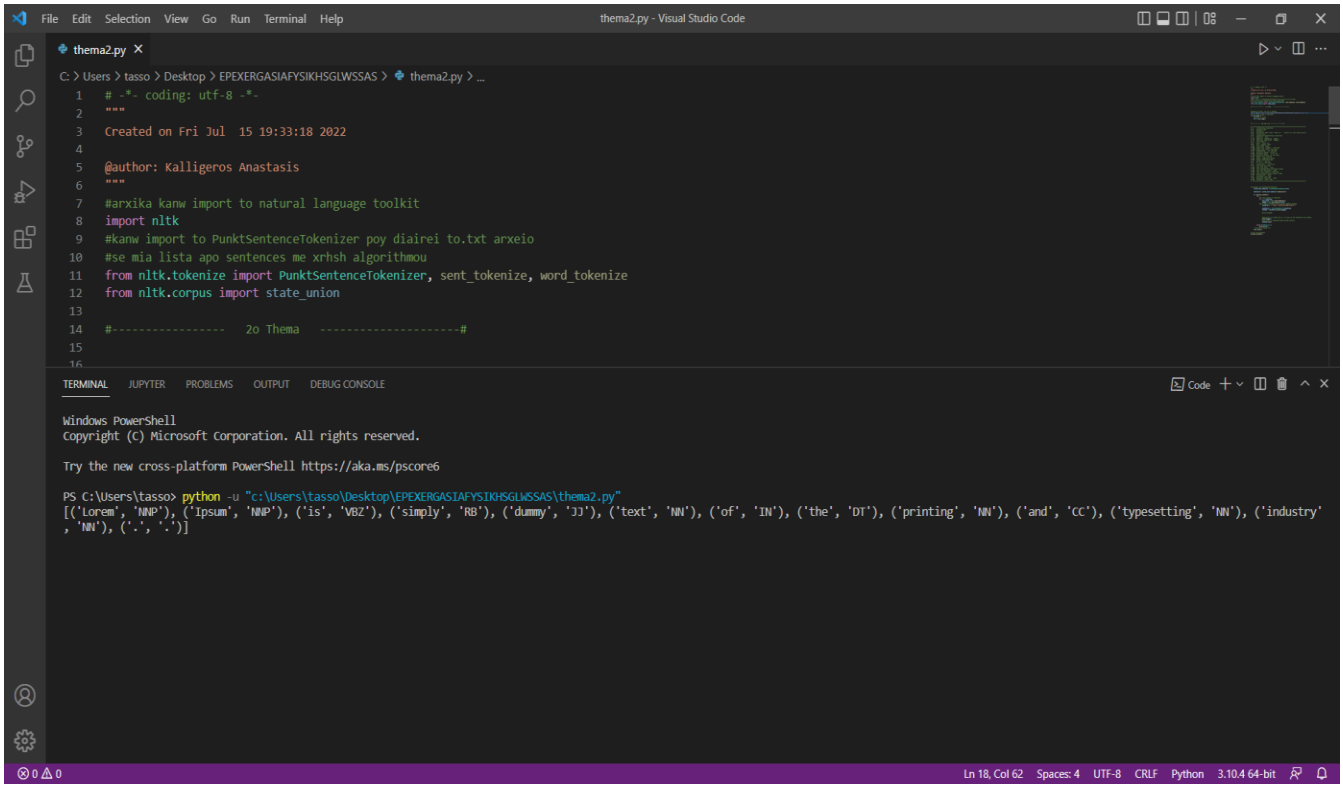
thema1.py



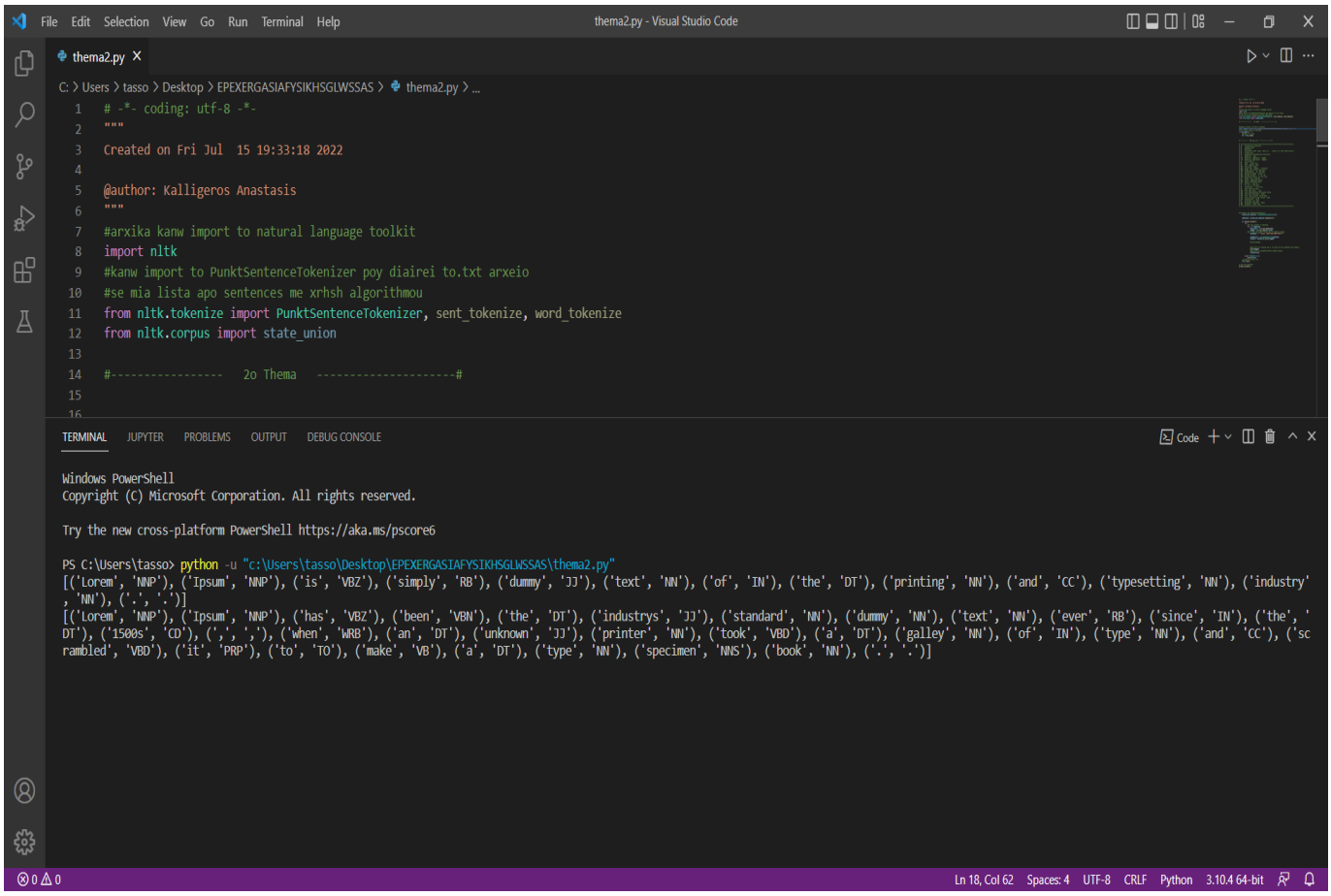
Θέμα 2ο (20 μονάδες)

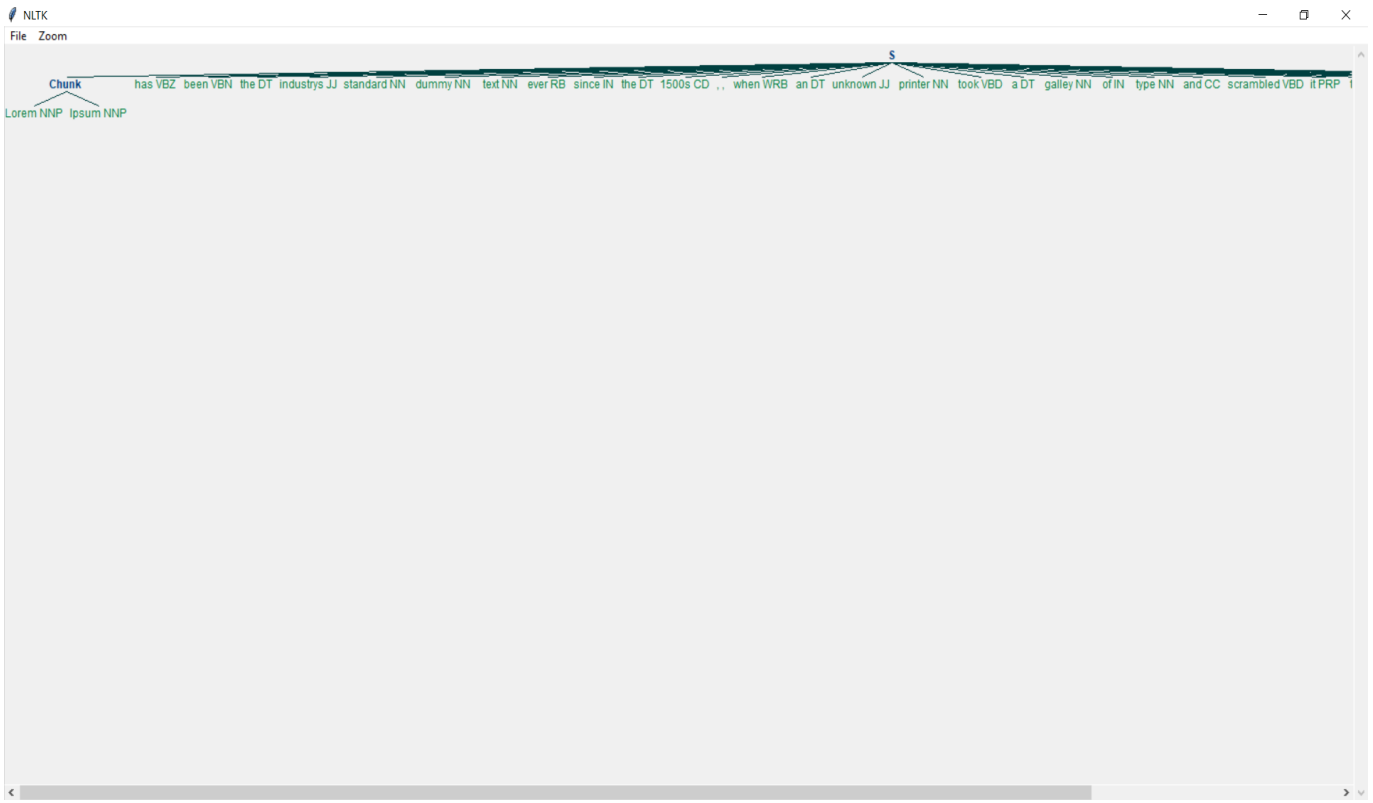
Δημιουργία Συντακτικού Αναλυτή:

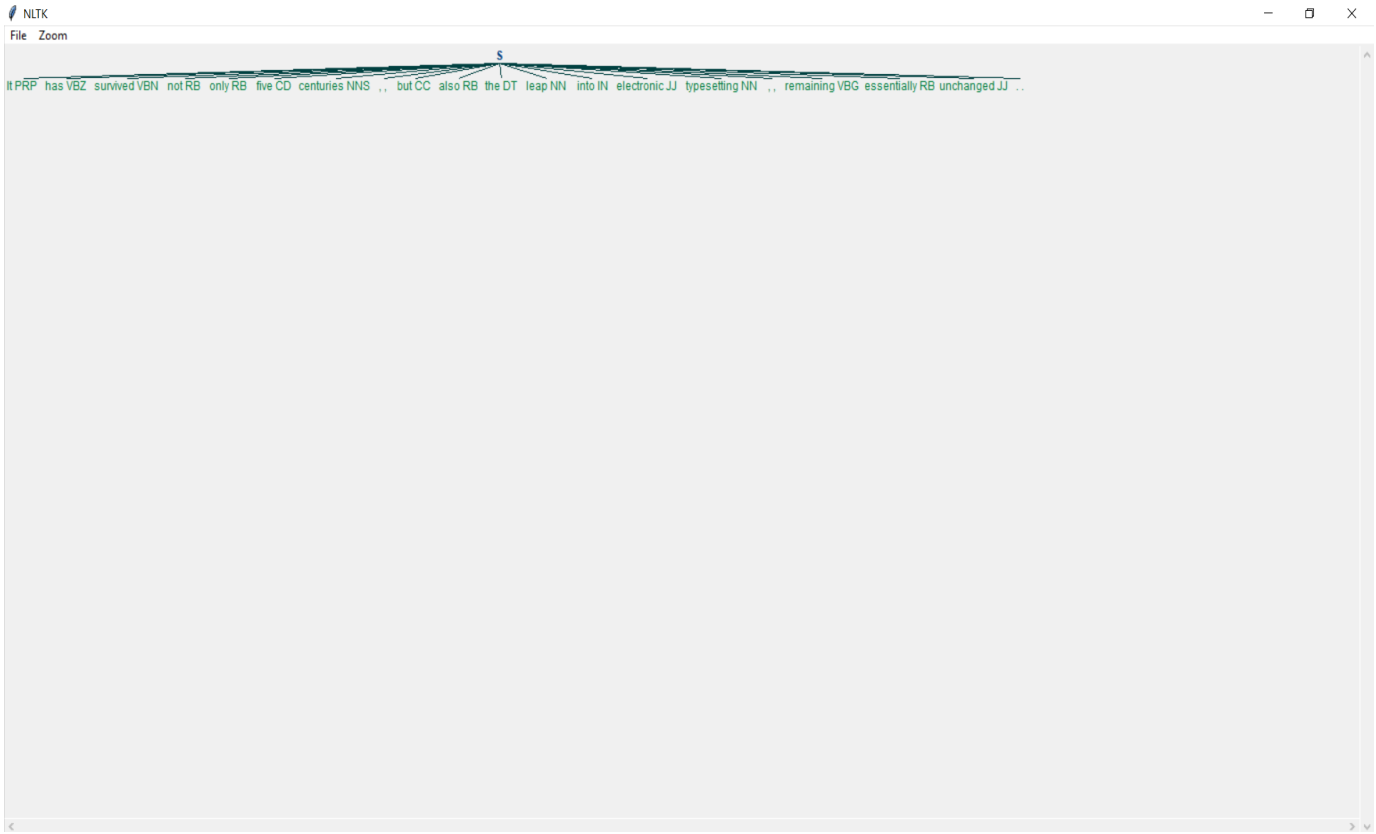
thema2.py

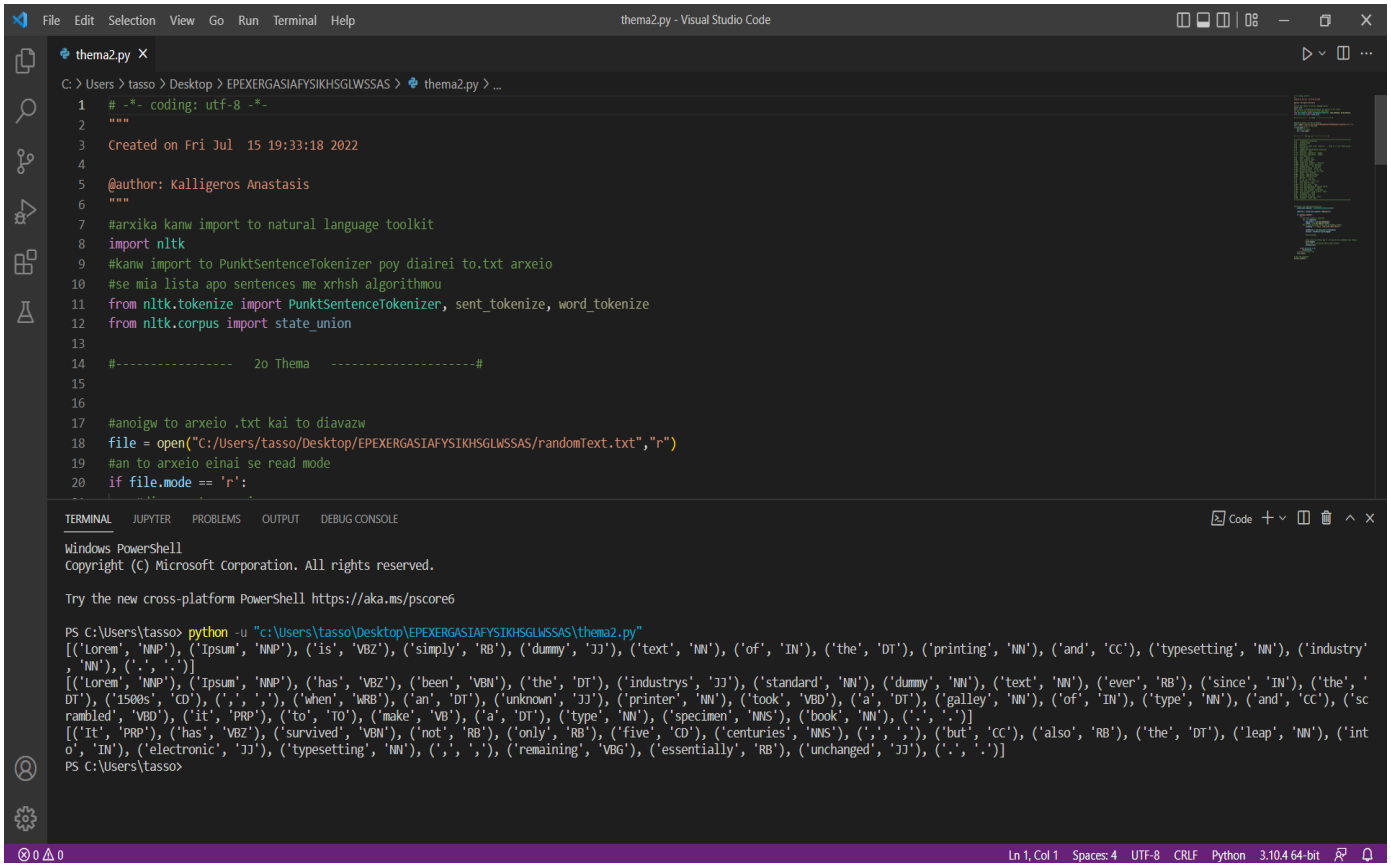



NLTK
File Zoom
S
Chunk
is VBZ simply RB dummy JJ text NN of IN the DT printing NN and CC typesetting NN industry NN ..
Lorem NNP Ipsum NNP


File Edit Selection View Go Run Terminal Help
thema2.py - Visual Studio Code
thema2.py
C: > Users > tasso > Desktop > EPEXERGASIAFYSIKHSGLWSSAS > thema2.py > ...
-*- coding: utf-8 -*-
"""
Created on Fri Jul 15 19:33:18 2022
@author: Kalligeros Anastasis
"""
#arxika kanw import to natural language toolkit
import nltk
#kanw import to PunktSentenceTokenizer poy diairei to.txt arxeio
#se mia lista apo sentences me xrhsh algorithmou
from nltk.tokenize import PunktSentenceTokenizer, sent_tokenize, word_tokenize
from nltk.corpus import state_union
#---------- 2o Thema ----------#
TERMINAL JUPYTER PROBLEMS OUTPUT DEBUG CONSOLE
Code
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\tasso> python -u "c:\Users\tasso\Desktop\EPEXERGASIAFYSIKHSGLWSSAS\thema2.py"
[('Lorem', 'NNP'), ('Ipsum', 'NNP'), ('is', 'VBZ'), ('simply', 'RB'), ('dummy', 'JJ'), ('text', 'NN'), ('of', 'IN'), ('the', 'DT'), ('printing', 'NN'), ('and', 'CC'), ('typesetting', 'NN'), ('industry', 'NN'), ('.', '.')]
[('Lorem', 'NNP'), ('Ipsum', 'NNP'), ('has', 'VBZ'), ('been', 'VBN'), ('the', 'DT'), ('industrys', 'JJ'), ('standard', 'NN'), ('dummy', 'NN'), ('text', 'NN'), ('ever', 'RB'), ('since', 'IN'), ('the', 'DT'), ('1500s', 'CD'), (',', ','), ('when', 'WRB'), ('an', 'DT'), ('unknown', 'JJ'), ('printer', 'NN'), ('took', 'VBD'), ('a', 'DT'), ('galley', 'NN'), ('of', 'IN'), ('type', 'NN'), ('and', 'CC'), ('scrambled', 'VBD'), ('it', 'PRP'), ('to', 'TO'), ('make', 'VB'), ('a', 'DT'), ('type', 'NN'), ('specimen', 'NNS'), ('book', 'NN'), ('.', '.')]
0 0
Ln 18, Col 62 Spaces: 4 UTF-8 CRLF Python 3.10.4 64-bit


NLTK
File Zoom
S
Chunk
has VBZ been VBN the DT industrys JJ standard NN dummy NN text NN ever RB since IN the DT 1500s CD , , when WRB an DT unknown JJ printer NN took VBD a DT galley NN of IN type NN and CC scrambled VBD it PRP
Lorem NNP Ipsum NNP

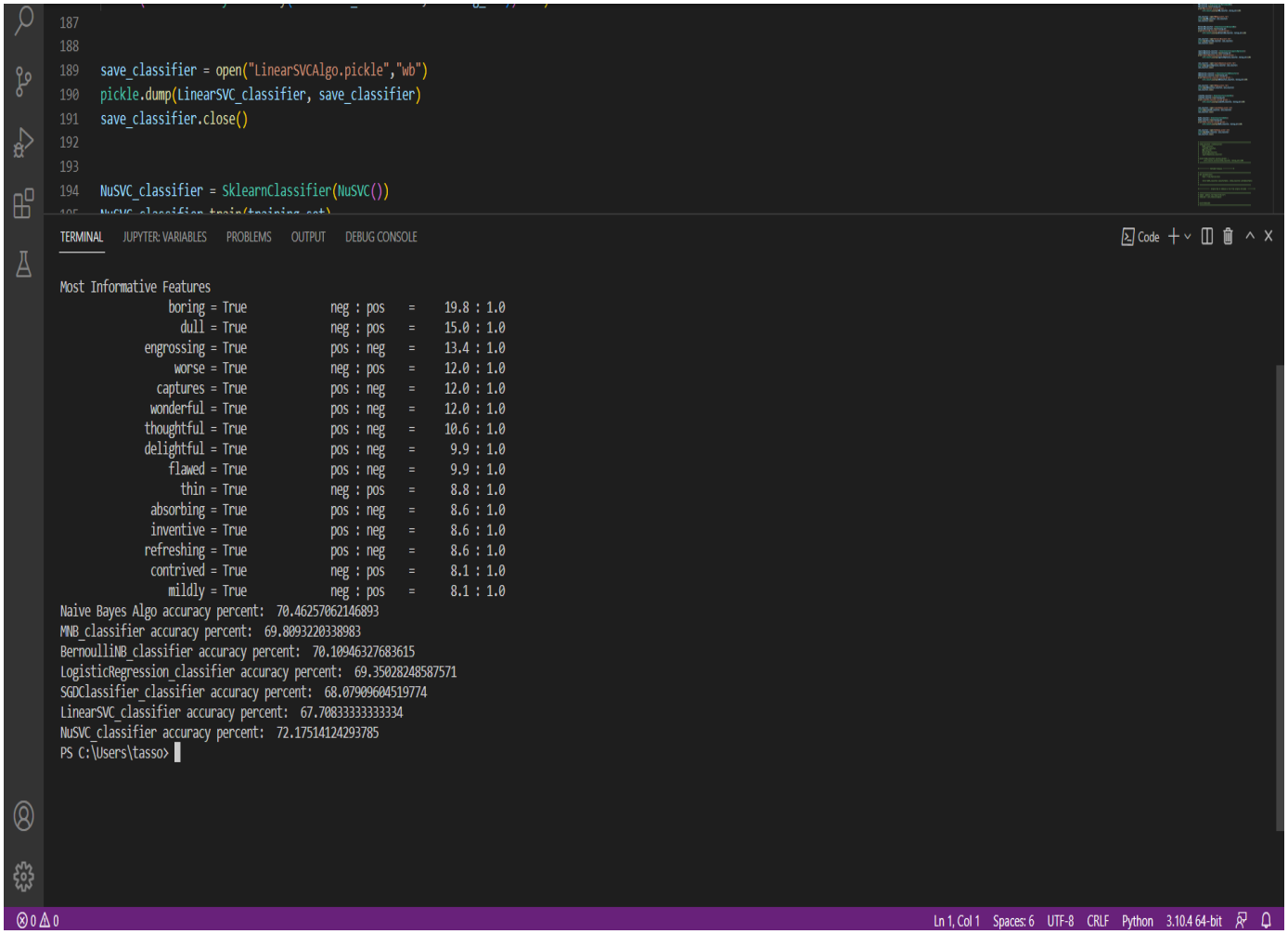

NLTK
File Zoom
S
It PRP has VBZ survived VBN not RB only RB five CD centuries NNS , , but CC also RB the DT leap NN into IN electronic JJ typesetting NN , , remaining VBG essentially RB unchanged JJ . .


File Edit Selection View Go Run Terminal Help
thema2.py - Visual Studio Code
thema2.py
C: > Users > tasso > Desktop > EPEXERGASIAFYSIKHSGLWSSAS > thema2.py > ...
-*- coding: utf-8 -*-
"""
Created on Fri Jul 15 19:33:18 2022
@author: Kalligeros Anastasis
"""
#arxika kanw import to natural language toolkit
import nltk
#kanw import to PunktSentenceTokenizer poy diairei to.txt arxeio
#se mia lista apo sentences me xrhsh algorithmou
from nltk.tokenize import PunktSentenceTokenizer, sent_tokenize, word_tokenize
from nltk.corpus import state_union
#----------------- 2o Thema --------------------#
#anoigw to arxeio .txt kai to diavazw
file = open("C:/Users/tasso/Desktop/EPEXERGASIAFYSIKHSGLWSSAS/randomText.txt","r")
#an to arxeio einai se read mode
if file.mode == 'r':
TERMINAL JUPYTER PROBLEMS OUTPUT DEBUG CONSOLE
Code
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\tasso> python -u "c:\Users\tasso\Desktop\EPEXERGASIAFYSIKHSGLWSSAS\thema2.py"
[('Lorem', 'NNP'), ('Ipsum', 'NNP'), ('is', 'VBZ'), ('simply', 'RB'), ('dummy', 'JJ'), ('text', 'NN'), ('of', 'IN'), ('the', 'DT'), ('printing', 'NN'), ('and', 'CC'), ('typesetting', 'NN'), ('industry', 'NN'), ('.', '.')]
[('Lorem', 'NNP'), ('Ipsum', 'NNP'), ('has', 'VBZ'), ('been', 'VBN'), ('the', 'DT'), ('industrys', 'JJ'), ('standard', 'NN'), ('dummy', 'NN'), ('text', 'NN'), ('ever', 'RB'), ('since', 'IN'), ('the', 'DT'), ('1500s', 'CD'), (',', ','), ('when', 'WRB'), ('an', 'DT'), ('unknown', 'JJ'), ('printer', 'NN'), ('took', 'VBD'), ('a', 'DT'), ('galley', 'NN'), ('of', 'IN'), ('type', 'NN'), ('and', 'CC'), ('scrambled', 'VBD'), ('it', 'PRP'), ('to', 'TO'), ('make', 'VB'), ('a', 'DT'), ('type', 'NN'), ('specimen', 'NNS'), ('book', 'NN'), ('.', '.')]
[('It', 'PRP'), ('has', 'VBZ'), ('survived', 'VBN'), ('not', 'RB'), ('only', 'RB'), ('five', 'CD'), ('centuries', 'NNS'), (',', ','), ('but', 'CC'), ('also', 'RB'), ('the', 'DT'), ('leap', 'NN'), ('into', 'IN'), ('electronic', 'JJ'), ('typesetting', 'NN'), (',', ','), ('remaining', 'VBG'), ('essentially', 'RB'), ('unchanged', 'JJ'), ('.', '.')]
PS C:\Users\tasso>
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.10.4 64-bit

Θέμα 3ο (30 μονάδες)

Δημιουργία Σημασιολογικού Αναλυτή:

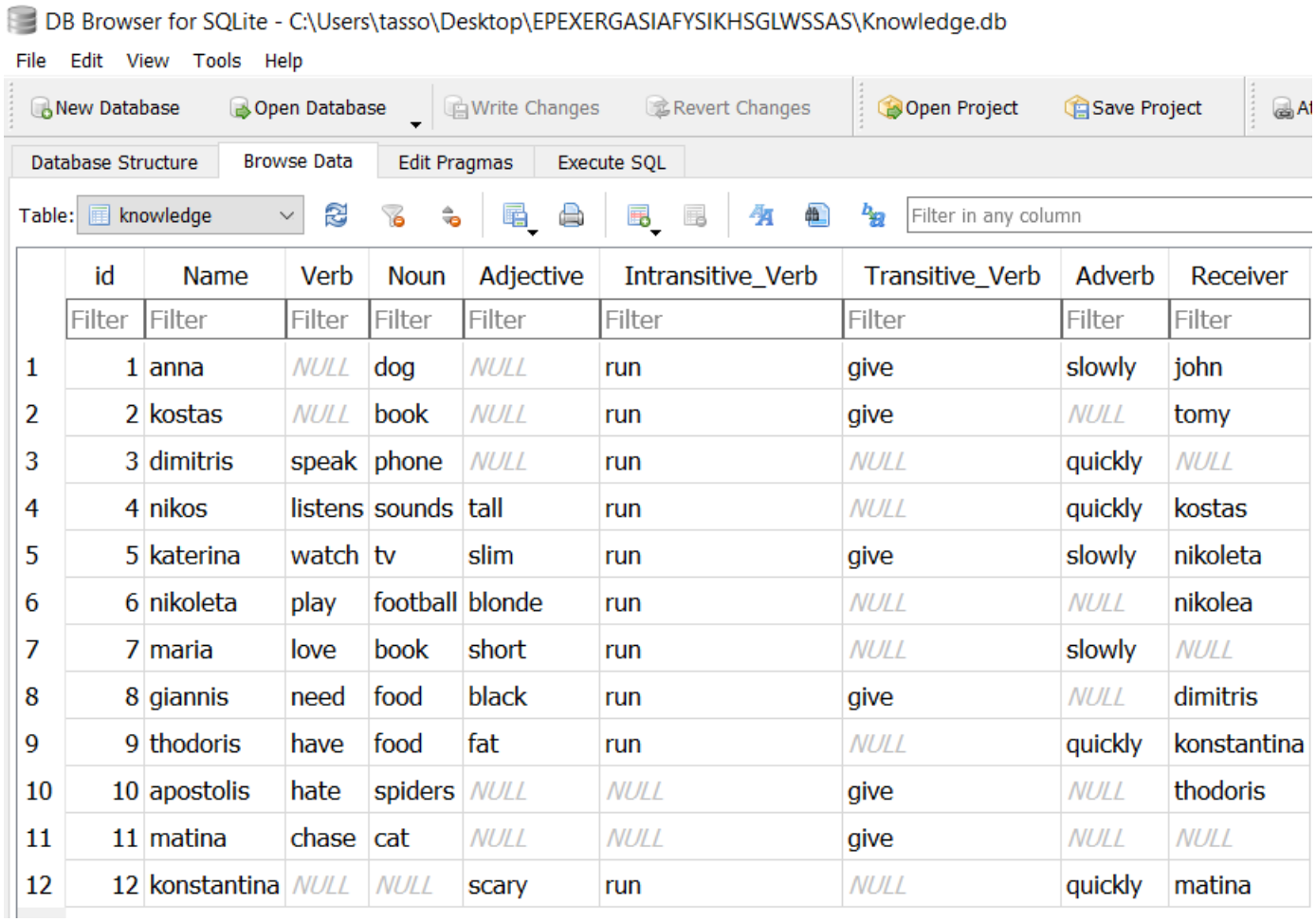
naturalLang.py



Όπως βλέπετε η λέξη dull θα εμφανιστεί 15 φορές στις αρνητικές κριτικές προς 1 που θα εμφανιστεί στις θετικές το οποίο είναι λογικό. Πιο κάτω βλέπουμε τους διάφορους αλγόριθμους και τις επί τοις 100 αποδόσεις τους. Πιο αποδοτικός στη συγκεκριμένη εκτέλεση είναι ο αλγόριθμος NuSVC(Nu-Support Vector Classification) με 72.17% απόδοση. Τέλος χρησιμοποιούμε κάνοντας import το pickle για να αποθηκεύουμε τους αλγορίθμους μας και να είναι πιο αποδοτικό και γρήγορο το πρόγραμμά μας.

Θέμα 4ο (30 μονάδες)

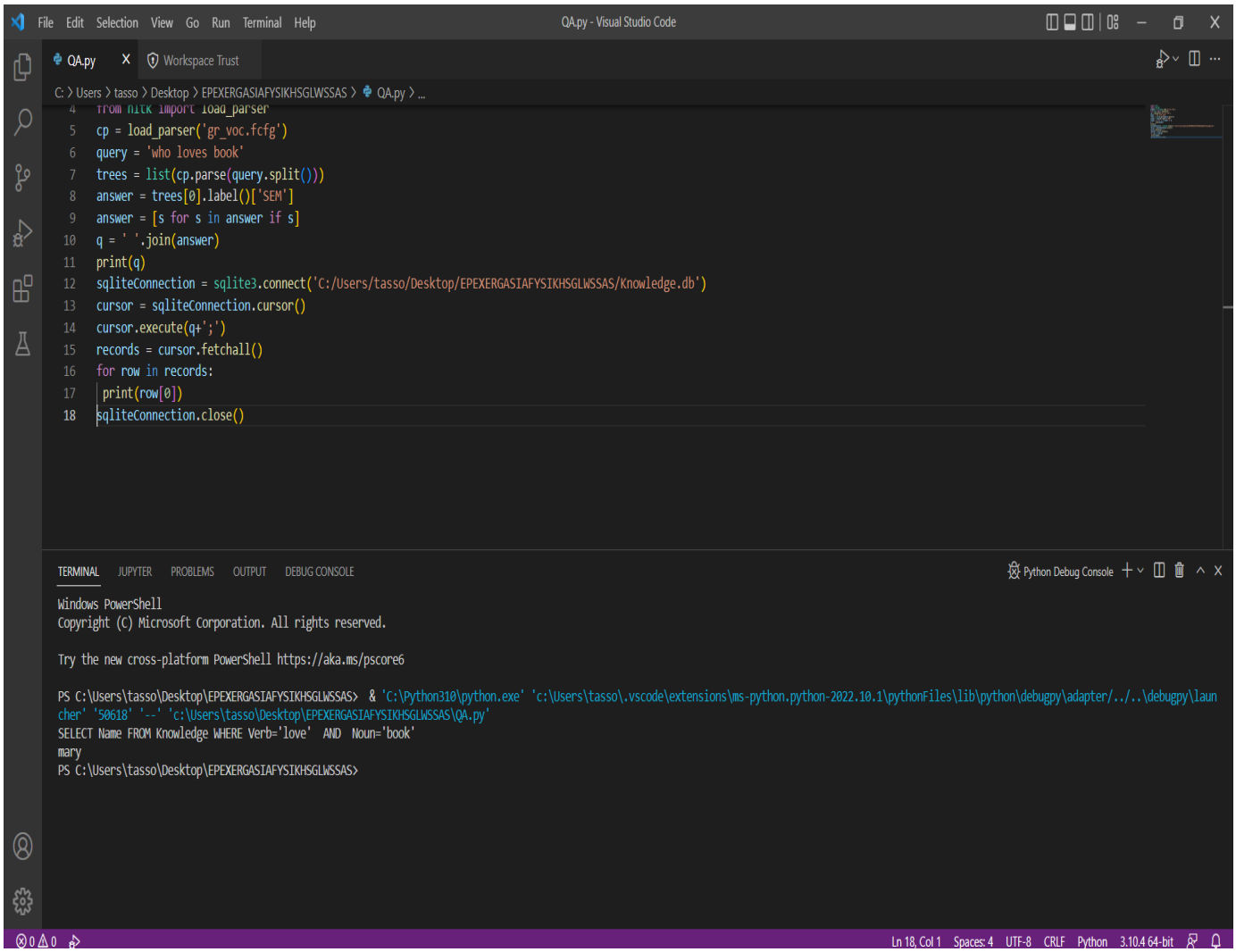
Ο πίνακας που κατασκευάστηκε είναι ο εξής:



	id	Name	Verb	Noun	Adjective	Intransitive_Verb	Transitive_Verb	Adverb	Receiver
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	anna	NULL	dog	NULL	run	give	slowly	john
2	2	kostas	NULL	book	NULL	run	give	NULL	tomy
3	3	dimitris	speak	phone	NULL	run	NULL	quickly	NULL
4	4	nikos	listens	sounds	tall	run	NULL	quickly	kostas
5	5	katerina	watch	tv	slim	run	give	slowly	nikoleta
6	6	nikoleta	play	football	blonde	run	NULL	NULL	nikolea
7	7	maria	love	book	short	run	NULL	slowly	NULL
8	8	giannis	need	food	black	run	give	NULL	dimitris
9	9	thodoris	have	food	fat	run	NULL	quickly	konstantina
10	10	apostolis	hate	spiders	NULL	NULL	give	NULL	thodoris
11	11	matina	chase	cat	NULL	NULL	give	NULL	NULL
12	12	konstantina	NULL	NULL	scary	run	NULL	quickly	matina

Στο αρχείο gr_voc.fcfg έχουμε δημιουργήσει μια γραμματική και ένα λεξιλόγιο. Η γραμματική μας όπως έχει σχεδιαστεί δημιουργεί ερωτήματα (Queries) για την βάση γνώσης (Knowledge Base) τα οποία αφορούν την απάντηση του εκάστοτε ερωτήματος που τίθεται από τον χρήστη.

QA.py



Εκτελούμε το πρόγραμμα μας για να δούμε την έξοδο που προκύπτει σύμφωνα με το ερώτημα που υποβάλλαμε.

4 Βιβλιογραφικές Πηγές

Σημειώσεις GUNET2 (ΕΠΕΞΕΡΓΑΣΙΑ ΦΥΣΙΚΗΣ ΓΛΩΣΣΑΣ).