Μεταγλωττιστές 2017

Προγραμματιστική Εργασία #1

ΣΚΕΝΤΕΡΙΔΗΣ ΚΩΝΣΤΑΝΤΙΝΟΣ

Α.Μ.: Π2014221

Στα πλαίσια της πρώτης προγραμματιστικής εργασίας του μαθήματος “Μεταγλωττιστές” , έπρεπε να υλοποιήσουμε έναν κώδικα με τον οποίο να επιτυγχάνουμε την επεξεργασία των χρονικών τιμών, ενός αρχείου υποτίτλων .srt .

Ακολουθεί μια συνοπτική περιγραφή του κώδικα και ανάλυση της κανονικής έκφρασης:

Αρχικά έγινε η προσθήκη του module “re” , κάνοντας το import, έτσι ώστε να μπορούμε να χρησιμοποιήσουμε τα διάφορα εργαλεία που παρέχει για την υλοποίηση κανονικών εκφράσεων.

Ο κώδικας είχε προετοιμαστεί κατάλληλα έτσι ώστε να μπορεί να λαμβάνει ως input, ένα αρχείο .txt και μια τιμή offset για την αλλαγή στα δευτερόλεπτα. Αυτές οι λειτουργίες υλοποιήθηκαν με την χρήση του module “argparse”.

Στην συνέχεια ετοίμασα και αποθήκευσα την “regular expression” μου σε ένα string, ώστε να έχω τον κώδικα πιο συμμαζεμένο.

Την μεταβλητή την ονόμασα restr (*όπως φαίνεται να ακολουθεί κάποια σύμβαση*) και μέσα σαυτήν έγραψα την ακόλουθη κανονική έκφραση:

r'(\d{2}):(\d{2}):(\d{2}),(\d{3})\s-->\s(\d{2}):(\d{2}):(\d{2}),(\d{3})$'

Το r που βρίσκεται στην αρχή, δείχνει στην python ότι η έκφραση πρόκειται για ένα “raw string”. Με την χρήση του raw string η python θα καταλάβει τα escape sequences κανονικά (*π.χ.παραπάνω το \s δείχνει έναν κενό χαρακτήρα*).

Με το regular expression έπρεπε να αναγνωρίσω string της μορφής:

00:00:00,000 → 00:00:00,000

Στην συγκεκριμένη περίπτωση επέλεξα να κάνω χρήση του \d που είναι ίδιο με το να γράφαμε [0-9] . Με το \d κάνουμε matching οποιουδήποτε digit.

Με την χρήση του {2} (*που ακολουθεί το \d*) λέμε στην python ότι θα πρέπει να κάνει match 2 digits.

Μετά θέλουμε να κάνει match το σύμβολο “:”

Συνεχίζουμε με παρόμοιο τρόπο και για τις άλλες σειρές από digits και “:”, και για τον κενό χαρακτήρα κάνουμε χρήση του \s.

Οι παρενθέσεις που έχουν μπει (*π.χ. (\d{2})*), δημιουργούν μια σειρά από groups που θα μας φανούν χρήσιμα στην συνέχεια, όταν θα θέλουμε να χρησιμοποιήσουμε συγκεκριμένα κομμάτια.

Στο τέλος το $ κάνει match το τέλος του string ή κάνει match μέχρι πριν το newline.

Αυτό αποτελεί το regular expression που χρησιμοποιείται για την συγκεκριμένη εργασία.

Στην συνέχεια το βάζουμε μέσα στο rexp = re.compile(restr) ώστε να το ορίσουμε κανονικά.

Έπειτα, μέσα στην loop που διαβάζει το .txt αρχείο line by line, κάνουμε χρήση της συνάρτησης search και βάζουμε το αντικείμενο που προκύπτει από το search μέσα σε μια μεταβλητή με το όνομα m. (*m = rexp.search(line)*)

Μετά έχουμε μια σειρά από εντολές που έχουν ως στόχο την χρήση του offset για την αλλαγή των δευτερολέπτων του αρχείου .srt.

Με το left_seconds_old = m.group(3) , προσπαθούμε να βάλουμε μέσα σε μια μεταβλητή το matching που γίνεται στα digit των δευτερολέπτων.

Στην συνέχεια σε μια άλλη μεταβλητή βάζουμε το αποτέλεσμα της πρόσθεσης του offset (*το οποίο είναι float όπως δηλώνεται στην γραμμή 17 μέσα στον parser.add_argument*) με τα αρχικά δευτερόλεπτα (*τα οποία μετατρέπουμε σε float*).

Μετά για να βάλουμε πάλι πίσω στο .txt την καινούργια τιμή, την μετατρέπουμε σε string με την χρήση της str().

Και τέλος χρησιμοποιούμε την re.sub για να γίνει η αντικατάσταση της καινούργιας τιμής στα σημεία που θέλουμε.

Στο string αντικατάστασης, με το r’\3’ λέμε να κάνει αντικατάσταση στο group(3) την καινούργια τιμή για τα δευτερόλεπτα.

Η ίδια διαδικασία γίνεται και για τα δευτερόλεπτα του δεξιού τμήματος.

Τέλος τυπώνουμε τις γραμμές του .txt μία μία στην κονσόλα με την εντολή

Sys.stdout.write(line)

Και έπειτα κλείνουμε το αρχείο για να αποδεσμεύσουμε την μνήμη που καταλαμβάνει.

Πηγές:

https://docs.python.org/2/library/re.html

http://stackoverflow.com/questions/13444675/python-re-storing-multiple-matches-in-variables

https://www.youtube.com/watch?v=sZyAn2TW7GY

https://www.digitalocean.com/community/tutorials/how-to-convert-data-types-in-python-3

https://www.tutorialspoint.com/python3/python_variable_types.htm

http://www.python-course.eu/python3_re.php

https://docs.python.org/3.1/library/re.html

https://www.tutorialspoint.com/python3/python_reg_expressions.htm

https://docs.python.org/3/tutorial/inputoutput.html

https://www.digitalocean.com/community/tutorials/how-to-handle-plain-text-files-in-python-3

https://www.tutorialspoint.com/python3/python_files_io.htm

http://www.python-course.eu/python3_re_advanced.php

https://docs.python.org/2/library/re.html#match-objects

https://docs.python.org/2/howto/regex.html#regex-howto

https://docs.python.org/2/library/re.html#re.compile

http://code.runnable.com/UqV9JYh03AFGAACS/how-to-use-findall-finditer-split-sub-and-subn-in-regular-expressions-in-python-for-regex

https://www.youtube.com/watch?v=vzeeXoPh-hQ