ΑΝΑΦΟΡΑ ΕΡΓΑΣΙΑΣ 1 - ΥΛΟΠΟΙΗΣΗ ΤΕΛΕΣΤΩΝ

Δημήτριος Παγώνης

5 Απριλίου 2025

Μέρος 1 (merge-join)

Τρόπος σκέψης για την επίλυση της άσκησης

Αρχικά, όρισα τη συνάρτηση merge_join η οποία παίρνει σαν ορίσματα το πρώτο αρχείο (το έχω στο μυαλό μου σαν το βασικό μου αρχείο για τις συγκρίσεις), το δεύτερο αρχείο και το αρχείο στο οποίο θα γράψουμε τα αποτελέσματα του merge_join.

Η πρώτη μου σκέψη ήταν να ορίσω τρεις 'δείκτες' αλλά έπειτα από μερικές δοκιμές αντιλήφθηκα πως είχα λάθος στην υλοποίηση και έτσι όρισα μόνο δύο δείκτες: line1,line2.Ο δείκτης line1 "σκανάρει" γραμμή-γραμμή το βασικό αρχείο και ο δείκτης line2 "σκανάρει" γραμμή-γραμμή αποκλειστικά το δεύτερο αρχείο.Επίσης, όρισα δύο ακόμη μεταβλητές οι οποίες κρατάνε την προηγούμενη γραμμή του εκάστοτε αρχείου στις περιπτώσεις που το χρειάζομαι για την αποφυγή διπλοτύπων.

Έχω ορίσει ένα flag το οποίο ουσιαστικά θα με ειδοποιεί για το εάν το πρώτο στοιχείο της προηγούμενης γραμμής που διάβασα(key1) είναι ίσο με το παρόν πρώτο και έτσι να χρησιμοποιήσω τον buffer.Η λογική είναι η εξής:

→ flag = 1 : είναι η τιμή στην οποία αρχικοποιούμε το flag.

- Εάν βρισκόμαστε στην περίπτωση όπου η τιμή που θα συγκρίνουμε από το πρώτο αρχείο είναι μεγαλύτερη από αυτήν του δεύτερου (key1>key2), τότε δεν υπάρχει κάποια αλλαγή στο flag και έτσι αυτό παραμένει 1.

- Η επόμενη περίπτωση στην οποία η τιμή του flag επαναφέρεται στην τιμή 1 είναι όταν βρίσκεται στην περίπτωση όπου η τιμή που θα συγκρίνουμε από το πρώτο αρχείο είναι μεγαλύτερη από αυτήν του δεύτερου (key<key2) ώστε όταν γίνουν οι κατάλληλες συγκρίσεις που θα περιγραφούν παρακάτω ο buffer να αδειάσει.

→ flag = 0 :

- Το flag γίνεται μηδέν όταν βρίσκομαι σε περίπτωση όπου τα δύο στοιχεία είναι ίσα (key1 == key).Έτσι, αφού πρώτα γράψουμε το αποτέλεσμα του join στο νέο αρχείο, και γεμίσουμε τον buffer με τα στοιχεία που θα χρειαστούμε από το αρχείο S, θα θέσουμε το flag στην τιμή 0 ώστε να γνωρίζουμε πως ο buffer περιέχει στοιχεία τα οποία θα χρησιμοποιήσουμε στις επόμενες συγκρίσεις. Έτσι, όταν μπούμε στη συνθήκη key1 < key2 και το flag είναι 0, θα ξεκινήσει μια σειρά από ελέγχους για να χειριστεί τα edge cases.

Εντολή εκτέλεσης κώδικα

```
python3 ask1.1.py R_sorted.tsv S_sorted.tsv RjoinS.tsv
```

Μέρος 2 (union)

Τρόπος σκέψης για την επίλυση της άσκησης

Ακολουθώντας τον ίδιο τρόπο σκέψης με την πρώτη άσκηση, χρησιμοποίησα τις εντολές: line1 = next(f1,None),line2 = next(f2,None) για να διαβάζω γραμμή-γραμμή από τα δύο αρχεία. Η διαφοροποίηση σε σχέση με το Μέρος 1 είναι πως χρησιμοποίησα 4 κλειδιά προκειμένου να ελέγχω αρχικά τα στοιχεία της πρώτης στήλης με βάση τη λεξικογραφική τους σειρά και αν αυτή είναι η ιδία τότε να ελέγχω με βάση τον αριθμό στη δεύτερη στήλη.Επίσης χρησιμοποίησα δύο μεταβλητές οι οποίες ελέγχουν το εκάστοτε αρχείο για διπλότυπα.Με αυτόν τον τρόπο εξασφαλίζω πως κατά την ένωση και την εμφάνιση τους στο τελικό αρχείο τα στοιχεία θα εμφανιστούν sorted και χωρίς διπλότυπα.

Τέλος, έχω χρησιμοποιήσει δύο if statements για να χειρίζομαι τις περιπτώσεις όπου ένα αρχείο τελειώνει πιο νωρίς από το άλλο και έτσι αυτό που ακόμα τρέχει να προσθέτει απλώς τις εναπομείνασες γραμμές του στο τελικό αρχείο και η ένωση να ολοκληρώνεται σωστά.

Εντολή εκτέλεσης κώδικα

```
python3 ask1.2.py R_sorted.tsv S_sorted.tsv RunionS.tsv
```

Μέρος 3 (intersection)

Τρόπος σκέψης για την επίλυση της άσκησης

Σε αυτήν την περίπτωση δεν χρειάστηκε να χρησιμοποιήσω κάποια επιπλέον μεταβλητή μιας και η σύγκριση αφορούσε ολόκληρη τη γραμμή.Συνεπώς , χρησιμοποιώντας την ίδια λογική για το διάβασμα γραμμών, υλοποίησα τον ίδιο αλγόριθμο.

Εντολή εκτέλεσης κώδικα

```
python3 ask1.3.py R_sorted.tsv S_sorted.tsv RintersectionS.tsv
```

Μέρος 4 (difference)

Τρόπος σκέψης για την επίλυση της άσκησης

Αρχικά, χρησιμοποίησα τον ίδιο αλγόριθμο για να διατρέξω τα δύο αρχεία. Έπειτα, χρησιμοποίησα μια μεταβλητή(prev_line1) προκειμένου να "κρατάω" την προηγούμενη γραμμή του πρώτου αρχείου σύγκρισης (στην προκειμένη περίπτωση το R_sorted.tsv) και με αυτόν τον τρόπο να μην έχω διπλότυπα στο τελικό αρχείο.

Εντολή εκτέλεσης κώδικα

```
python3 ask1.4.py R_sorted.tsv S_sorted.tsv RdifferenceS.tsv
```

Μέρος 5 (Ομαδοποίηση και συνάθροιση)

Τρόπος σκέψης για την επίλυση της άσκησης

Αρχικά, δημιούργησα μια συνάρτηση για να διαβάζει τα δεδομένα από ένα αρχείο .tsv και να τα καταχωρεί σε έναν πίνακα προκειμένου να μπορώ να τον αξιοποιήσω στον αλγόριθμο merge_join. Τα δεδομένα μου είναι κατανεμημένα στον πίνακα σε ζευγάρια κλειδιού-τιμής , έχω δηλαδή μικρούς πίνακες, δύο στοιχείων ο καθένας, μέσα στον κύριο πίνακά μου.Έπειτα, χρησιμοποίησα τις δύο κύριες συναρτήσεις που απαιτούνται για την υλοποίηση του αλγόριθμου merge-sort με τη μόνη διαφοροποίηση πως όταν έβρισκα ίδια "κλειδιά", προσέθετα τις τιμές τους, αφού πρώτα τις είχα μετατρέψει σε ακεραίους.

Εντολή εκτέλεσης κώδικα

```
python3 ask1.5.py R.tsv Rgroupby.tsv
```