Project 2

COMP211

Technical University of Crete

Odysseas Stavrou 2018030199

Compute Sales

Το σκριπτάκι computeSales.py περιληπτικά διαβάζει ένα αρχείο που περιέχονται μέσα διάφορες αποδείξεις και αθροίζει τις ολικές ποσότητες των προϊόντων, παράλληλα κρατώντας σε μια λίστα τα προϊόντα που πούλησε η κάθε επιχείριση(ΑΦΜ).

Αφού ο χρήστης τρέξει το πρόγραμμα για πρώτη φορά, του παρουσιάζεται ένα menu επιλογών με 4 διαθέσιμες επιλογές. Ανάλογα με το τι επιλέξει ο χρήστης, εκτελείτε και αφού τελειώσει η εκτέλεση επιστρέφει πάλι στο κυρίως menu. (Γίνονται οι κατάλληλοι έλεγχοι έτσι ώστε αν ο χρήστης δώσει κάτι άλλο εκτός από τις 4 δυνατές επιλογές 1,2,3,4 τότε να του ξαναζητείτε η είσοδος)

Το πρόγραμμα χωρίζεται σε 3 μέρη

1. Ανάγνωση και επεξεργασία αρχείου
2. Εκτύπωση ενός προϊόντος για όλα τα ΑΦΜ (και ολικό πόσο)
3. Εκτύπωση όλων των προϊόντων για ένα ΑΦΜ (και ολικό πόσο)

<u>Εκτύπωση ενός προϊόντος για όλα τα ΑΦΜ (και ολικό ποσό):</u>

- Ζητείτε από τον χρήστη το όνομα του προϊόντος
- Καλείτε η συνάρτηση GetItemSum με όρισμα το προϊόν που επέλεξε ο χρήστης
- Δημιουργούνται 2 κενές λίστες (μια για τα ΑΦΜ και μια για το ολικό ποσό του προϊόντος για κάθε ΑΦΜ) και μια μεταβλητή boolean αρχικοποιείτε σε True
- Διασχίζουμε το λεξικό και για κάθε ένα ΑΦΜ παίρνουμε τα προϊόντα του
- Ελέγχουμε αν στα προϊόντα υπάρχει το ζητούμενο, αν ναι τότε προσθέτουμε το ΑΦΜ στη λίστα για τα ΑΦΜ
- Για να πάρουμε το ολικό ποσό για το προϊόν για το συγκεκριμένο ΑΦΜ πρέπει να πιάσουμε τη θέση του στη λίστα τον προϊόντων της επιχείρησης και ακολούθως να πιάσουμε την τιμή που βρίσκετε σε αυτήν τη θέση της λίστας με τις τιμές όλων των προϊόντων της επιχείρησης
- Τότε μπορούμε να προσθέσουμε και το ολικό ποσό στη λίστα προς εκτύπωση
- Αν βρήκαμε το προϊόν σε τουλάχιστο 1 ΑΦΜ τότε μπορούμε να τα τυπώσουμε και ΔΕΝ μεταβαίνουμε απευθείας στο κυρίως menu (flag = False)

- Ταξινομούμε τις λίστες παράλληλα και τυπώνουμε τις δυο λίστες και επιστρέφουμε στο κυρίως menu

<u>Εκτύπωση όλων των προϊόντων για ένα ΑΦΜ (και ολικό ποσό)</u>

- Ζητείτε από τον χρήστη το ΑΦΜ
- Καλείτε η συνάρτηση GetAfmItemSum με όρισμα το ΑΦΜ
- Γίνεται έλεγχος αν υπάρχει το ΑΦΜ στο λεξικό, αν όχι τότε δεν τυπώνουμε τίποτα και επιστρέφουμε το κυρίως menu
- Εξάγουμε τις δυο λίστες που περιέχουν η μια τα προϊόντα και η άλλη τα ολικά ποσά για κάθε προϊόν
- Τις ταξινομούμε και τυπώνουμε προϊόν και ολικό ποσό
- Αν οι λίστες την ώρα της ταξινόμησης είναι κενές τότε δεν τυπώνουμε κάτι
- Επιστρέφουμε στο κυρίως menu

<u>Ανάγνωση και επεξεργασία αρχείου</u>

- Ζητείτε από τον χρήστη το όνομα του αρχείου
- Ελέγχεται εάν υπάρχει, αν όχι τότε επιστρέφουμε στο κυρίως menu
- Καλείτε η συνάρτηση process με όρισμα το όνομα του αρχείου
- Ανοίγουμε το αρχείο με Unicode encoding διότι έχουμε ελληνικούς χαρακτήρες και διαβάζουμε την πρώτη γραμμή
- Η επεξεργασία καθορίζεται από το τι περιέχει η κάθε γραμμή:
  - **<u>Αν η γραμμή ξεκινάει από '-'</u>**
    1. Τότε ξέρουμε σίγουρα ότι ξεκινάει μια απόδειξη (χωρίς όμως να ξέρουμε αν είναι έγκυρη)
    2. Διαβάζουμε κάθε χαρακτήρα της γραμμής και ελέγχουμε αν είναι "-", αν έστω ένας χαρακτήρας δεν είναι τότε είναι λάθος απόδειξη, θέτουμε True τη μεταβλητή και βγαίνουμε έξω από τον βρόγχο
    3. Αν είναι σωστή η απόδειξη, επαναφέρουμε τον δείκτη λάθους απόδειξης, bad_rec σε False

4. Διαβάζουμε την επομένη γραμμή

- **<u>Αν οι 3 πρώτοι χαρακτήρες είναι “ΑΦΜ” και η απόδειξη είναι ορθή</u>**

1. Τότε χωρίζουμε τη γραμμή στο “ “
2. Αν δεν υπάρχει “space” μεταξύ “ΑΦΜ” και αριθμού τότε θεωρούμε ότι η απόδειξη είναι λανθασμένη και θέτουμε τον δείκτη σε True διαβάζουμε επομένη γραμμή μέχρι να βρούμε “-” για να επαναφέρουμε τον δείκτη
3. Αν το αφμ είναι σωστό τότε παίρνουμε μόνο τον αριθμό και ελέγχουμε αν αποτελείτε από μόνον 10 χαρακτήρες και μόνο αριθμούς. Αν όχι τότε επαναλαμβάνουμε τη διαδικασία όταν έχει λάθος η απόδειξη.
4. Τέλος, αν το ΑΦΜ δεν υπάρχει στο καθολικό λεξικό το προσθέτουμε ως κλειδί και για τιμή έχει ένα άλλο λεξικό με κλειδιά “items”, “total_item”, και για τιμές 1 λίστα το καθένα.

- **<u>Αν η απόδειξη μέχρι στιγμής είναι ορθή</u>**

1. Χωρίζουμε τη γραμμή και ελέγχουμε αν χωρίστηκε σε 4 κομμάτια(προϊόν, ποσότητα, κόστος ανά ποσότητα, ολικό προϊόντος), αν δεν τότε η απόδειξη είναι λάθος
2. Μετατρέπουμε τα strings σε integers και floats στα ανάλογα στοιχειά, αν υπάρχει εξαίρεση τότε περιέχονται γράμματα όποτε η απόδειξη είναι λανθασμένη
3. Ελέγχουμε αν το τελικό πόσο είναι ίσο με ποσότητα * κόστος ανά ποσότητα αν όχι τότε η απόδειξη είναι λανθασμένη
4. Προσθέτουμε την τιμή αυτή σε μια μεταβλητή που κράτα το κόστος όλης της απόδειξης.
5. Έχοντας δημιουργήσει 2 λίστες (μια για τις ολικές τιμές και μια για τα προϊόντα). Ελέγχουμε αν στα προϊόντα υπάρχει το τρέχον, αν υπάρχει τότε προσθέτουμε στην αντίστοιχη θέση της παράλληλης λίστας των τιμών την τρέχουσα τιμή, αν όχι τότε το προϊόν και η τιμή μπαίνουν στο τέλος και των 2 λιστών
6. Διαβάζουμε την επομένη γραμμή

- **<u>Αν η απόδειξη ξεκινάει από "ΣΥΝΟΛΟ" και είναι ορθή</u>**

    1. Μετατρέπουμε το σύνολο σε δεκαδικό αριθμό και αν υπάρχει εξαίρεση τότε η απόδειξη είναι λάθος
    2. Ελέγχουμε και την επομένη γραμμή αν είναι "-" (και αν αποτελείτε μόνο από "-") που σημαίνει ότι τερμάτισε η τρέχον, αν δεν τερματίζει ή αν δεν περιέχει μόνο "-", τότε και οι ΔΥΟ αποδείξεις είναι λάθος (η τρέχον γιατί δεν τερμάτισε, και η επομένη γιατί δεν ξεκίνησε)
    3. Αν όλοι οι προαναφερόμενοι έλεγχοι ήταν σωστοί τότε καλούμε τη συνάρτηση pushem με ορίσματα το ΑΦΜ, και τις 2 λίστες
    4. Επαναφέρουμε τις βοηθητικές μεταβλητές

- **<u>Αν δεν ισχύει τίποτα από τα προηγούμενα</u>**

    1. Αυτό σημαίνει πως η απόδειξη είναι εσφαλμένη άλλα ακόμα να φτάσουμε στο τέλος άρα απλά διαβάζουμε την επόμενη γραμμή μέχρι να βρούμε "-"

<u>Η συνάρτηση sortem(x,y):</u>

Η συνάρτηση αυτή παίρνει όρισμα 2 λίστες και ταξινομεί παράλληλα τη y με βάση τη θέση των στοιχείων της x.

<u>Η συνάρτηση pushem(afm, items, prices):</u>

Η συνάρτηση αυτή έχει ως ορίσματα το ΑΦΜ και τις 2 λίστες της οποίες θα "συγχωνεύσει" με τις εκάστοτε λίστες στο λεξικό. Ουσιαστικά παίρνει ένα-ένα τα προϊόντα από τα ορίσματα και βλέπει αν υπάρχει ήδη στα προϊόντα της επιχείρησης, αν όχι τότε το προσθέτει μαζί με την τιμή στο τέλος της αντίστοιχης λίστας, αν υπάρχει τότε προσθέτει στην τρέχουσα τιμή την καινούρια.

<u>Σημειώσεις - Δομές δεδομένων:</u>

Χρησιμοποίησα 2 λεξικά το ένα μέσα στο άλλο έτσι ώστε αν έχουμε πολλές αποδείξεις μπορώ να έχω σχεδόν άμεσα τις 2 λίστες για κάθε ΑΦΜ. Το "έξω" λεξικό έχει ως κλειδί κάθε ένα από τα ΑΦΜ, και για τιμή έχει ένα άλλο λεξικό όπου αυτό έχει ως κλειδιά τη λέξη "items", "item_total" και για τιμές 2 λίστες όπου περιέχουν η κάθε μια τα προϊόντα και της τιμές τους ανάλογα. Η ανάγνωση των αποδείξεων γίνεται σχετικά αργά (σε σχέση με της δυνατότητες του λεξικού), διότι διανύω τις λίστες εντός

των λεξικών για εισαγωγή των δεδομένων, και ταξινόμηση. Άλλα οι απαντήσεις στα ερωτήματα του χρήστη γίνονται άμεσα διότι τα δεδομένα είναι έτοιμα.

Το να διαβάζω κάθε γραμμή αν περιέχει μόνο “-” εκσφενδονίζει την πολυπλοκότητα και τον χρόνο ανάγνωσης ενός αρχείου (κάποιες φορές μέχρι και 300%). Επίσης διότι αντιμετώπισα αρκετά προβλήματα με δεκαδικούς, έπρεπε κάθε φορά που έκανα πράξεις με δεκαδικούς να στρογγυλεύω την τιμή, αν αυτή ήταν μέσα σε λίστα τότε η πολυπλοκότητα αυξάνεται ακόμα παραπάνω. Υπάρχουν διάφορες εντολές εκτυπωσης τις οποίες έχω βάλει σε σχολεία που δίνουν μια περαιτέρω εικόνα για το πως δούλεψα.