ΗΥ-240: Δομές Δεδομένων
Εαρινό Εξάμηνο - Ακαδημαϊκό Έτος 2020
Διδάσκουσα: Παναγιώτα Φατούρου
Προγραμματιστική Εργασία - 1ο Μέρος

Ημερομηνία Παράδοσης: Δευτέρα, 30 Μαρτίου 2020

Τρόπος Παράδοσης: Χρησιμοποιώντας το πρόγραμμα turnin. Πληροφορίες για το πώς λειτουργεί το πρόγραμμα turnin παρέχονται στην ιστοσελίδα του μαθήματος.

Γενική Περιγραφή:
Στην εργασία αυτή καλείστε να υλοποιήσετε ένα πρόγραμμα που να προσομοιώνει τη λειτουργία ενός ομίλου αεροπορικών εταιρειών που εκτελούν πτήσεις σε διάφορες χώρες. Το πρόγραμμα που θα υλοποιήσετε θα πρέπει να επιτρέπει στον όμιλο να εντάσσει καινούριες εταιρείες (companies). Η κάθε εταιρεία έχει ένα στόλο από αεροπλάνα (airplanes) καθένα εκ των οποίων εκτελεί πτήσεις προς ένα συγκεκριμένο προορισμό. Ο όμιλος παρέχει πτήσεις προς συγκεκριμένους προορισμούς (destinations) μέσω των διαφορετικών εταιρειών που τον απαρτίζουν.

Αναλυτική Περιγραφή Ζητούμενης Υλοποίησης

Το σύστημα αποτελείται από ένα σύνολο εταιρειών που απαρτίζουν τον όμιλο και αποθηκεύονται σε μια κυκλική διπλά συνδεδεμένη λίστα (όπως αυτή που παρουσιάζεται στις διαφάνειες του μαθήματος η οποία έχει έναν εικονικό-dummy κόμβο) που ονομάζεται **λίστα αεροπορικών εταιρειών (airlines)**. Ο κάθε κόμβος της λίστας αντιστοιχεί σε μια αεροπορική εταιρεία και υλοποιείται ως μια εγγραφή τύπου struct airline με τα ακόλουθα πεδία:

- aId: Αναγνωριστικό (τύπου int) που χαρακτηρίζει μοναδικά την αεροπορική εταιρία.
- airplains: Δείκτης (τύπου struct airplanes *) στο πρώτο στοιχείο μιας **απλά συνδεδεμένης λίστας** που ονομάζεται **λίστα αεροπλάνων της αεροπορικής εταιρείας**.
- next: Δείκτης (τύπου struct airline *) στον επόμενο κόμβο της λίστας αεροπορικών εταιρειών.
- prev: Δείκτης (τύπου struct airline *) στον προηγούμενο κόμβο της λίστας αεροπορικών εταιρειών.

Η λίστα αεροπλάνων μιας αεροπορικής εταιρείας είναι ταξινομημένη με βάση το αναγνωριστικό του κάθε αεροπλάνου. Κάθε στοιχείο της λίστας αυτής αντιστοιχεί σε ένα αεροπλάνο της αεροπορικής εταιρείας και είναι μια εγγραφή τύπου struct airplane με τα παρακάτω πεδία:

- pId: Αναγνωριστικό (τύπου int) που χαρακτηρίζει μοναδικά το αεροπλάνο.
- dest: Αναγνωριστικό (τύπου int) που χαρακτηρίζει μοναδικά τον προορισμό στον οποίο ταξιδεύει το αεροπλάνο. Το εύρος τιμών αυτού του πεδίου είναι μεταξύ [0, 10].
- depart_time: Αναγνωριστικό (τύπου int) που αντιστοιχεί στην ώρα αναχώρησης του αεροπλάνου προς τον προορισμό του. Η ώρα θα έχει τη μορφή HHMM. Για παράδειγμα, η ώρα 1620 αντιστοιχεί στην ώρα 16:20.
- next: Δείκτης (τύπου struct airplane *) στον επόμενο κόμβο της λίστας αεροπλάνων.

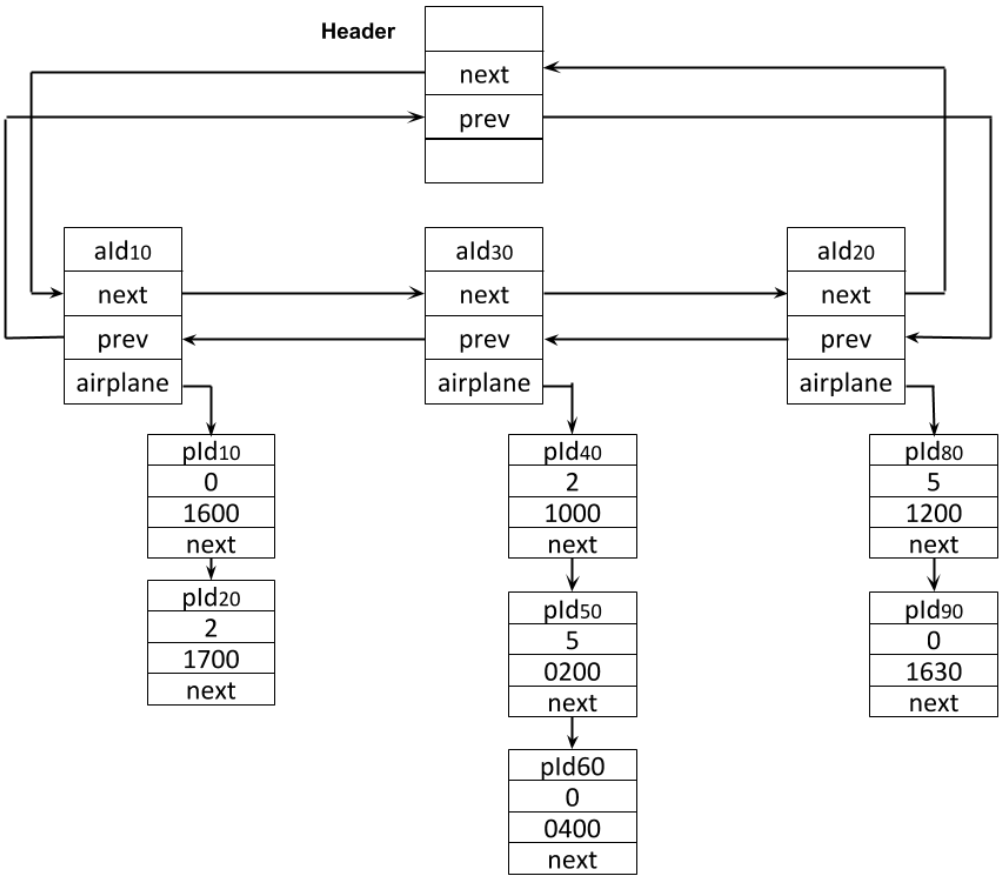
Στο Σχήμα 1 παρουσιάζεται η λίστα των αεροπορικών εταιρειών, ο κάθε κόμβος της οποίας περιέχει έναν δείκτη προς το πρώτο στοιχείο της λίστας αεροπλάνων της αντίστοιχης αεροπορικής εταιρείας.

Το σύστημα περιέχει επίσης ένα μηχανισμό για την κατηγοριοποίηση των αεροπλάνων στους προορισμούς που ταξιδεύουν. Ο μηχανισμός αυτός υλοποιείται με ένα πίνακα DESTINATIONS[MAX_DEST] που ονομάζεται πίνακας προορισμών. Θεωρούμε ότι υπάρχουν συνολικά 10 προορισμοί και έτσι ο πίνακας έχει 10 θέσεις, μια για κάθε προορισμό. Επομένως MAX_DEST = 10. Το στοιχείο i του πίνακα είναι ένας δείκτης σε μία απλά συνδεδεμένη ταξινομημένη λίστα που ονομάζεται **λίστα πτήσεων του προορισμού i** και περιέχει όλα τα αεροπλάνα που ταξιδεύουν με προορισμό τον προορισμό i. Η λίστα αυτή είναι ταξινομημένη με βάση την ώρα αναχώρησης και υλοποιείται με κόμβο φρουρό. Σε

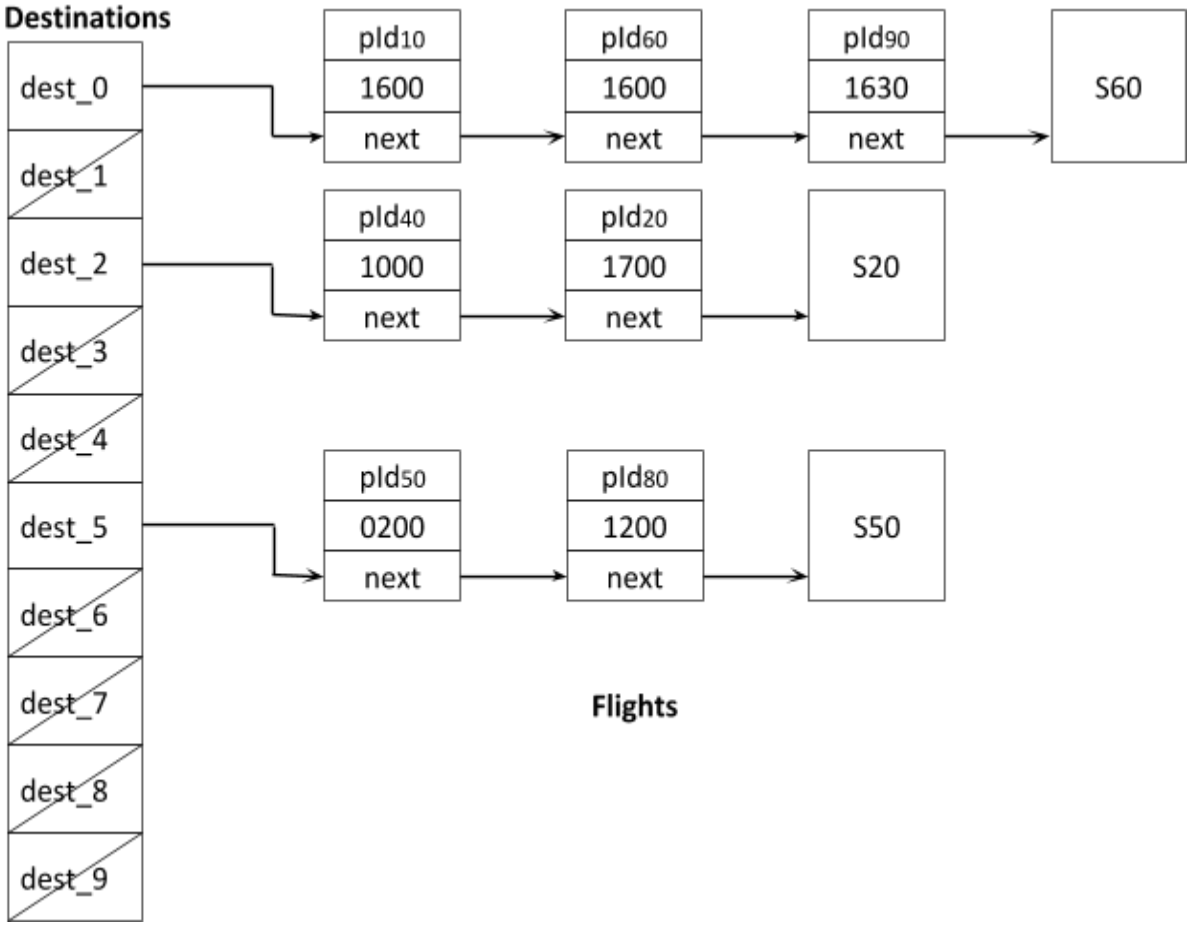
περίπτωση που δύο πτήσεις αντιστοιχούν στην ίδια ώρα, η σειρά εμφάνισής τους στη λίστα θα είναι με βάση το αναγνωριστικό του αεροπλάνου. Ο κάθε κόμβος της λίστας είναι μια εγγραφή τύπου struct flight με τα ακόλουθα πεδία:

- pId: Αναγνωριστικό (τύπου int) που χαρακτηρίζει μοναδικά το αεροπλάνο
- depart_time: Αναγνωριστικό (τύπου int) που αντιστοιχεί στην ώρα αναχώρησης του αεροπλάνου προς τον προορισμό του. Η ώρα θα έχει τη μορφή HHMM. Για παράδειγμα, η ώρα 1620 αντιστοιχεί στην ώρα 16:20. Υποθέτουμε, για λόγους απλούστευσης της εργασίας, ότι κάθε αεροπλάνο πετάει καθημερινά μία φορά προς το μοναδικό προορισμό του.
- next: Δείκτης (τύπου struct flight *) στον επόμενο κόμβο της λίστας πτήσεων.

Στο Σχήμα 2 παρουσιάζεται ο πίνακας προορισμών, το κάθε κελί του οποίου περιέχει μια λίστα από πτήσεις.



Σχήμα 1: Η διπλά συνδεδεμένη λίστα αεροπορικών εταιρειών, η οποία είναι μη-ταξινομημένη. Επίσης, η απλά συνδεδεμένη λίστα των airplanes για κάθε εταιρεία, η οποία είναι ταξινομημένη με βάση το αναγνωριστικό του κάθε αεροπλανου.



Σχήμα 2: Ο πίνακας προορισμών (destinations) όπου κάθε στοιχείο του πίνακα είναι ένας δείκτης σε μια ταξινομημένη απλά-συνδεδεμένη λίστα με κόμβο φρουρό, η οποία περιέχει πληροφορίες για τις πτήσεις που ταξιδεύουν σε αυτό τον προορισμό.

Τρόπος Λειτουργίας Προγράμματος

Το πρόγραμμα που θα δημιουργηθεί θα πρέπει να εκτελείται καλώντας την ακόλουθη εντολή:

<executable> <input-file>

όπου <executable> είναι το όνομα του εκτελέσιμου αρχείου του προγράμματος (π.χ. a.out) και <input-file> είναι το όνομα ενός αρχείου εισόδου (π.χ. testfile) το οποίο περιέχει γεγονότα των ακόλουθων μορφών:

R <aId>

Γεγονός τύπου Register company που υποδηλώνει την εισαγωγή μιας εταιρείας με αναγνωριστικό <aId> στον όμιλο. Κατά το γεγονός αυτό θα γίνεται εισαγωγή ενός νέου κόμβου τύπου struct airline στη λίστα των αεροπορικών εταιρειών. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
R <aId>
        Airlines = <aId1>, <aId2>,...,<aIdn>
DONE
```

όπου n είναι ο αριθμός των κόμβων στη λίστα αεροπορικών εταιρειών και κάθε $i \in \{1,..,n\}$,<aId$_i$> είναι το αναγνωριστικό της αεροπορικής που αντιστοιχεί στον i-οστό κόμβο της λίστας αυτής.

I <aId> <pId> <dest> <depart_time>

Γεγονός τύπου Insert airplane που υποδηλώνει την εισαγωγή ενός αεροπλάνου με αναγνωριστικό <pId> στη λίστα αεροπλάνων της εταιρείας με αναγνωριστικό <aId>. Το αεροπλάνο πετάει προς τον προορισμό με αναγνωριστικό <dest> και η ώρα αναχώρησης θα είναι η <depart_time>. Κατά το γεγονός αυτό θα πρέπει να πραγματοποιούνται οι ακόλουθες ενέργειες. Αρχικά, θα αναζητάτε στη λίστα των αεροπορικών εταιρειών την εταιρία με αναγνωριστικό <aId> και στη συνέχεια θα εισάγετε στη λίστα των αεροπλάνων της αεροπορικής εταιρείας ένα νέο κόμβο με αναγνωριστικό <pId>. Μετά από κάθε εισαγωγή, η λίστα των αεροπλάνων θα πρέπει να παραμένει ταξινομημένη. Τέλος, θα πρέπει να εισάγετε ένα νέο κόμβο στον πίνακα προορισμών. Για την εισαγωγή αυτή θα πρέπει να χρησιμοποιήσετε την παράμετρο <dest> του γεγονότος για να εντοπίσετε την κατάλληλη λίστα στον πίνακα προορισμών και έπειτα θα εισάγετε ένα νέο κόμβο που να αντιπροσωπεύει τη νέα πτήση. Η λίστα πτήσεων θα πρέπει να παραμένει ταξινομημένη μετά το πέρας της εισαγωγής. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
I <aId> <pId> <destination> <depart_time>
        Airline1 = <pId1,1:aId1,1>, <pId1,2:aId1,2>,...,<pId1,m1:aId1,m1>
        Airline2 = <pId2,1:aId2,1>, <pId2,2:aId2,2>,...,<pId2,m2:aId2,m2>
        ...
        Airlinen = <pIdn,1:aIdn,1>, <pIdn,2:aIdn,2>,...,<pIdn,mn:aIdn,mn>

DONE
```

για κάθε i, $1 \le i \le n$, mi είναι το πλήθος των κόμβων της λίστας των αεροπλάνων του i-οστής εταιρείας, και για κάθε $j \in \{1,\ldots,mi\}$, $pId_{i,j}$, $aId_{i,j}$ είναι το αναγνωριστικό του αεροπλάνου και το αναγνωριστικό της αεροπορικής εταιρείας, αντίστοιχα, που αντιστοιχεί στον j-οστό κόμβο της λίστας των αεροπλάνων της i-οστής εταιρείας.

C <aId> <pId> <dest>

Γεγονός τύπου Cancel flight που υποδηλώνει την απόσυρση του αεροπλάνου με αναγνωριστικό <pId> από το στόλο της αεροπορικής εταιρείας με αναγνωριστικό <aId>. Κατά το γεγονός αυτό θα πρέπει να αναζητήσετε την εταιρεία με αναγνωριστικό <aId> στη λίστα αεροπορικών εταιρειών και στη συνέχεια να αφαιρέσετε από τη λίστα αεροπλάνων της το αεροπλάνο με αναγνωριστικό <pId>. Επιπρόσθετα, θα πρέπει να χρησιμοποιήσετε την παράμετρο <dest> του γεγονότος για να εντοπίσετε την κατάλληλη λίστα στον πίνακα προορισμών και έπειτα θα πρέπει να αφαιρέσετε τον κόμβο με αναγνωριστικό <pId> που αντιπροσωπεύει το αεροπλάνο προς απόσυρση. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
C <aId> <pId> <dest>
        Airline1 = <pId1,1:aId1,1>, <pId1,2:aId1,2>,...,<pId1,m1:aId1,m1>
        Airline2 = <pId2,1:aId2,1>, <pId2,2:aId2,2>,...,<pId2,m2:aId2,m2>
        ...
        Airlinen = <pIdn,1:aIdn,1>, <pIdn,2:aIdn,2>,...,<pIdn,mn:aIdn,mn>

DONE
```

για κάθε i, $1 \le i \le n$, mi είναι το πλήθος των κόμβων της λίστας των αεροπλάνων της i-οστής εταιρείας, και για κάθε $j \in \{1,\ldots,mi\}$, $pId_{i,j}$, $aId_{i,j}$ είναι το αναγνωριστικό του αεροπλάνου και το αναγνωριστικό της αεροπορικής εταιρείας, αντίστοιχα, που αντιστοιχεί στον j-οστό κόμβο της λίστας των αεροπλάνων της i-οστής εταιρείας.

D <aId>

Γεγονός τύπου Delete airline που υποδηλώνει τη διαγραφή της εταιρείας με αναγνωριστικό <aId> από τον όμιλο. Κατά το γεγονός αυτό, θα πρέπει να αναζητήσετε στη λίστα αεροπορικών εταιρειών την εταιρεία με αναγνωριστικό <aId> και στη συνέχεια να

διατρέξετε και να αποσύρετε όλα τα αεροπλάνα στη λίστα αεροπλάνων της. Για κάθε αεροπλάνο που βρίσκετε στη λίστα αεροπλάνων της εταιρείας θα πρέπει να το διαγράφετε από τον πίνακα προορισμών. Τέλος, θα πρέπει να διαγράψετε από τη λίστα αεροπορικών εταιρειών την εταιρεία με αναγνωριστικό <aId>. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

D <aId>
Airline$_1$ = <pId$_{1,1}$:aId$_{1,1}$>, <pId$_{1,2}$:aId$_{1,2}$>,...,<pId$_{1,m1}$:aId$_{1,m1}$>
Airline$_2$ = <pId$_{2,1}$:aId$_{2,1}$>, <pId$_{2,2}$:aId$_{2,2}$>,...,<pId$_{2,m2}$:aId$_{2,m2}$>
...
Airline$_n$ = <pId$_{n,1}$:aId$_{n,1}$>, <pId$_{n,2}$:aId$_{n,2}$>,...,<pId$_{n,mn}$:aId$_{n,mn}$>

DONE

για κάθε i, $1 \le i \le n$, mi είναι το πλήθος των κόμβων της λίστας των αεροπλάνων της i-οστής εταιρείας, και για κάθε $j \in \{1,\ldots,mi\}$, pId$_{i,j}$, aId$_{i,j}$ είναι το αναγνωριστικό του αεροπλάνου και το αναγνωριστικό της αεροπορικής εταιρείας, αντίστοιχα, που αντιστοιχεί στον j-οστό κόμβο της λίστας των αεροπλάνων της i-οστής εταιρείας.

A <aId1> <aId2>

Γεγονός τύπου Acquisition airline το οποίο σηματοδοτεί την εξαγορά της αεροπορικής εταιρείας <aId1> από την αεροπορική εταιρεία <aId2>. Κατά το γεγονός αυτό, θα πρέπει να αναζητήσετε στη λίστα των αεροπορικών εταιρειών τις εταιρείες με αναγνωριστικά <aId1> και <aId2>, αντίστοιχα και να συνενώσετε τις λίστες αεροπλάνων των εταιρειών. Προσοχή, η λίστα των αεροπλάνων της αεροπορικής εταιρείας με αναγνωριστικό <aId2> που θα προκύψει μετά τη συνένωση θα πρέπει να είναι ταξινομημένη, ενώ η λίστα αεροπλάνων της αεροπορικής εταιρείας <aId1> θα πρέπει να είναι κενή. Η χρονική πολυπλοκότητα αυτού του γεγονότος θα πρέπει να είναι O(n1+n2), όπου n1 και n2 είναι το πλήθος στοιχείων των λιστών αεροπλάνων των αεροπορικών εταιρειών <aId1> και <aId2>, αντίστοιχα. Τέλος θα πρέπει να διαγράφετε την εταιρεία με αναγνωριστικό <aId1> από τη λίστα των αεροπορικών εταιρειών. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

A <aId1> <aId2>
Airline$_1$ = <pId$_{1,1}$:aId$_{1,1}$>, <pId$_{1,2}$:aId$_{1,2}$>,...,<pId$_{1,m1}$:aId$_{1,m1}$>
Airline$_2$ = <pId$_{2,1}$:aId$_{2,1}$>, <pId$_{2,2}$:aId$_{2,2}$>,...,<pId$_{2,m2}$:aId$_{2,m2}$>
...
Airline$_n$ = <pId$_{n,1}$:aId$_{n,1}$>, <pId$_{n,2}$:aId$_{n,2}$>,...,<pId$_{n,mn}$:aId$_{n,mn}$>

DONE

για κάθε i, $1 \le i \le n$, mi είναι το πλήθος των κόμβων της λίστας των αεροπλάνων της i-οστής εταιρείας, και για κάθε $j \in \{1,\ldots,mi\}$, $pId_{i,j}$, $aId_{i,j}$ είναι το αναγνωριστικό του αεροπλάνου και το αναγνωριστικό της αεροπορικής εταιρείας, αντίστοιχα, που αντιστοιχεί στον j-οστό κόμβο της λίστας των αεροπλάνων της i-οστής εταιρείας.

S <aId1> <aId2> <dest>

Γεγονός τύπου Subsidiary company, όπου η εταιρεία με αναγνωριστικό <aId2> εξαγοράζει τα αεροπλάνα της αεροπορικής εταιρείας με αναγνωριστικό <aId1>, τα οποία εκτελούν πτήση προς ένα συγκεκριμένο προορισμό. Κατά το γεγονός αυτό, θα πρέπει να αναζητήσετε στην λίστα των αεροπορικών εταιρειών τις εταιρείες με αναγνωριστικά <aId1> και <aId2>, αντίστοιχα. Έπειτα, θα πρέπει να διατρέξετε τη λίστα των αεροπλάνων της <aId1>, και κάθε αεροπλάνο που εκτελεί πτήση προς τον προορισμό <dest>, θα το διαγράφετε από αυτή τη λίστα και θα το εισάγετε στη λίστα των αεροπλάνων της αεροπορικής εταιρείας με αναγνωριστικό <aId2>. Προσοχή: η λίστα των αεροπλάνων της εταιρείας με αναγνωριστικό <aId2> που θα προκύψει μετά την εκτέλεση του γεγονότος αυτού θα πρέπει να είναι ταξινομημένη. Η χρονική πολυπλοκότητα του γεγονότος θα πρέπει να είναι O(m), όπου m είναι ο συνολικός αριθμός των αεροπλάνων της εταιρίας της οποίας κάποια αεροπλάνα εξαγοράζονται. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

S <aId1> <aId2> <destinations>
Airline$_1$ = <$pId_{1,1}$:$aId_{1,1}$>, <$pId_{1,2}$:$aId_{1,2}$>,...,<$pId_{1,m1}$:$aId_{1,m1}$>
Airline$_2$ = <$pId_{2,1}$:$aId_{2,1}$>, <$pId_{2,2}$:$aId_{2,2}$>,...,<$pId_{2,m2}$:$aId_{2,m2}$>
...
Airline$_n$ = <$pId_{n,1}$:$aId_{n,1}$>, <$pId_{n,2}$:$aId_{n,2}$>,...,<$pId_{n,mn}$:$aId_{n,mn}$>

DONE

για κάθε i, $1 \le i \le n$, mi είναι το πλήθος των κόμβων της λίστας των αεροπλάνων της i-οστής εταιρείας, και για κάθε $j \in \{1,\ldots,mi\}$, $pId_{i,j}$, $aId_{i,j}$ είναι το αναγνωριστικό του αεροπλάνου και το αναγνωριστικό της αεροπορικής εταιρείας, αντίστοιχα, που αντιστοιχεί στον j-οστό κόμβο της λίστας των αεροπλάνων της i-οστής εταιρείας.

P <aid>

Γεγονός τύπου Partition airplanes equally, όπου η εταιρεία με αναγνωριστικό <aId> κλείνει και τα αεροπλάνα της αγοράζονται από (μοιράζονται ισότιμα στις) υπόλοιπες εταιρείες του ομίλου. Κατά το γεγονός αυτό, θα πρέπει να αναζητήσετε στη λίστα των αεροπορικών εταιρειών την εταιρεία με αναγνωριστικό <aId>. Έπειτα θα πρέπει να διατρέξετε τη λίστα των αεροπλάνων της εταιρείας αυτής και για κάθε αεροπλάνο που έχει στην κατοχή της θα πρέπει να κάνετε την κατάλληλη διάσχιση zig-zag στη λίστα εταιρειών προκειμένου να αποδοθεί ένα αεροπλάνο σε κάθε εταιρεία μέχρι η λίστα αεροπλάνων της συγκεκριμένης εταιρείας να αδειάσει. Η zig-zag διάσχιση θα πρέπει να γίνει ως εξής: Από τον κόμβο της

εταιρείας με αναγνωριστικό <aId> θα προχωράτε αρχικά ένα κόμβο δεξιά και ένα κόμβο αριστερά. Έπειτα πάλι από τον κόμβο της εταιρείας με αναγνωριστικό <aId> θα προχωράτε δύο κόμβους δεξία και δύο κόμβους αριστερά. Στη συνέχεια τρεις κόμβους δεξιά και τρεις κόμβους αριστερά κ.ο.κ Θα συνεχίσετε με αυτό τον βηματισμό μέχρι να δώσετε όλα τα αεροπλάνα. Έπειτα θα διαγράψετε την εταιρεία με αναγνωριστικό <aId> από τη λίστα αεροπορικών εταιρειών. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
P <aid>
        Airline1 = <pId1,1:aId1,1>, <pId1,2:aId1,2>,...,<pId1,m1:aId1,m1>
        Airline2 = <pId2,1:aId2,1>, <pId2,2:aId2,2>,...,<pId2,m2:aId2,m2>
        ...
        Airlinen = <pIdn,1:aIdn,1>, <pIdn,2:aIdn,2>,...,<pIdn,mn:aIdn,mn>

DONE
```

για κάθε i, $1 \le i \le n$, mi είναι το πλήθος των κόμβων της λίστας των αεροπλάνων της i-οστής εταιρείας, και για κάθε $j \in \{1,\ldots,mi\}$, $pId_{i,j}$, $aId_{i,j}$ είναι το αναγνωριστικό του αεροπλάνου και το αναγνωριστικό της αεροπορικής εταιρείας, αντίστοιχα, που αντιστοιχεί στον j-οστό κόμβο της λίστας των αεροπλάνων της i-οστής εταιρείας.

T <dest> <timestamp>

Γεγονός τύπου travel όπου ένας χρήστης θέλει να αναζητήσει όλες τις πτήσεις που ταξιδεύουν προς τον προορισμό <destination> και αναχωρούν μετά την ώρα <timestamp>. Κατά το γεγονός αυτό, θα διατρέχετε τη λίστα πτήσεων του προορισμού <dest> και θα εκτυπώνετε όλες τις πτήσεις που έχουν ώρα αναχώρησης μετά την ώρα <timestamp>. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
T <destination> <timestamp>
        Destination = <pId1:dt1>, ..., <pIdn:dtn>
DONE
```

για κάθε i, $1 \le i \le n$ όπου n είναι το πλήθος των κόμβων στην λίστα των πτήσεων του προορισμού και για κάθε $i \in \{1,\ldots,n\}$, pId_i, dt_i είναι το αναγνωριστικό του αεροπλάνου που εκτελεί την πτήση και το depart_time της πτήσης που αντιστοιχεί στον ι-οστό κόμβο της λίστας των πτήσεων του προορισμού.

X

Γεγονός τύπου print airlines το οποίο σηματοδοτεί την εκτύπωση όλων των αεροπορικών εταιρειών από τη λίστα αεροπορικών εταιρειών. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

X
Airline$_1$ = aId$_1$
Airplanes$_1$ = <pId$_{1,}$>, <pId$_{,2}$>,...,<pId$_{n1}$>
...
Airline$_k$ = aId$_k$
Airplanes$_k$ = <pId$_{1,}$>, <pId$_{,2}$>,...,<pId$_{nk}$>
DONE

όπου k είναι το πλήθος των κόμβων στην λίστα των αεροπορικών εταιρειών, για κάθε i, $1 \le i \le k$, n_i είναι το πλήθος των κόμβων της λίστας των αεροπλάνων της i-οστής αεροπορικής εταιρείας και για κάθε $j \in \{1,\ldots,Э\}$, pId_j είναι το αναγνωριστικό του j-οστού κόμβου της λίστας των αεροπλάνων του i-οστής εταιρείας.

Y

Γεγονός τύπου print destinations το οποίο σηματοδοτεί την εκτύπωση του πίνακα Destinations. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος, το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

Y
Destination$_1$ = <pId$_{1,1}$:dt$_{1,1}$>, ..., <pId$_{1,n1}$:dt$_{1,n1}$>
Destination$_2$ = <pId$_{2,1}$:dt$_{2,1}$>, ..., <pId$_{2,n2}$:dt$_{2,n2}$>
...
Destination$_k$ = <pId$_{k,1}$:dt$_{k,1}$>, ..., <pId$_{k,nk}$:dt$_{k,nk}$>
DONE

για κάθε j, $1 \le j \le k$ όπου n_j είναι το πλήθος των κόμβων στην λίστα των πτήσεων του j-οστού προορισμού και για κάθε $i \in \{1,\ldots,Э\}$, $pId_{j,i,}$ $dt_{j,i}$ είναι το αναγνωριστικό του αεροπλάνου που εκτελεί την πτήση και το depart_time της πτήσης που αντιστοιχεί στον ι-οστό κόμβο της λίστας των πτήσεων του j-οστού προορισμού.

Βαθμολογία Γεγονότων

R	10
I	10
C	12
D	12
A	15
S	15
T	5
P	15
X	3
Y	3

Δομές Δεδομένων

Στην υλοποίησή σας δεν επιτρέπεται να χρησιμοποιήσετε έτοιμες δομές δεδομένων (π.χ., ArrayList στη Java, κ.α.). Στη συνέχεια παρουσιάζονται οι δομές σε C που πρέπει να χρησιμοποιηθούν για την υλοποίηση της παρούσας εργασίας.

```
struct airlines {
        int aId;
        struct airplane *p_root;
        struct airlines *next;
        struct airlines *prev;
}

struct airplanes {
        int pId;
        int dest;
        int depart_time;
        struct airplanes *next;
}

struct  flights {
        int pId;
        int depart_time;
        struct flights *next;
}
/* global variable, pointer to the head of the airline list */
struct airlines *airlinesList;

/* global variable, array of lists of destinations  */
struct flights DESTINATIONS[10]
```

ΚΑΘΕ ΕΠΙΤΥΧΙΑ