HY240: Δομές Δεδομένων
Χειμερινό Εξάμηνο – Ακαδημαϊκό Έτος 2013-14
Διδάσκουσα: Παναγιώτα Φατούρου

Προγραμματιστική Εργασία
1ο Μέρος

Ημερομηνία Παράδοσης: Παρασκευή, 29 Νοεμβρίου 2013, ώρα 23:59.

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

Copyright for photos by Martin Doerr

Γενική Περιγραφή

Στην εργασία αυτή καλείστε να υλοποιήσετε ένα πρόγραμμα που περιγράφει (προσομοιώνει) τη λειτουργία μίας υπηρεσίας διαμοιρασμού φωτογραφιών παρόμοιας με το Instagram. Κάθε χρήστης μπορεί να αναρτά τις φωτογραφίες που επιθυμεί ώστε να μπορούν άλλοι χρήστες να τις προβάλουν. Για τις ανάγκες αυτής της εργασίας θεωρούμε ότι η υπηρεσία υποστηρίζει πολλαπλούς λογαριασμούς. Κάθε λογαριασμός ανήκει σε κάποιο χρήστη. Ο ίδιος χρήστης μπορεί να κατέχει περισσότερους του ενός λογαριασμούς. Ο χρήστης ενός λογαριασμού έχει τη δυνατότητα:

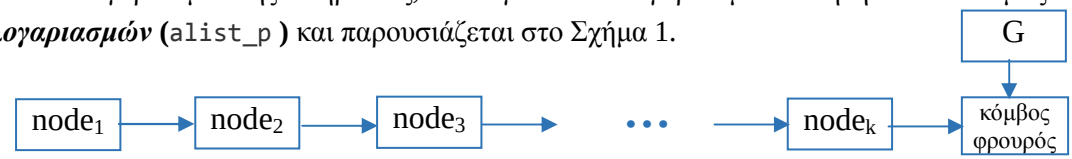
- να **αναρτά φωτογραφίες στο λογαριασμό**.
- να **δηλώνει πως του αρέσουν (like)** φωτογραφίες δικές του ή άλλων χρηστών.
- να **επισημαίνει (tag)** τον εαυτό του σε φωτογραφίες που τον απεικονίζουν ή να επισημαίνει άλλους χρήστες σε δικές του φωτογραφίες.

- να **ενώνει δύο λογαριασμούς που του ανήκουν**. Για παράδειγμα, ας υποθέσουμε ότι ένας χρήστης έχει δύο λογαριασμούς, έναν για οικογενειακές φωτογραφίες και έναν για επαγγελματικές. Ο χρήστης έχει τη δυνατότητα να αποφασίσει ότι θέλει να έχει μόνο ένα λογαριασμό όπου θα συγκεντρώσει τις φωτογραφίες των δύο άλλων λογαριασμών του.
- να **διαχωρίζει το σύνολο των φωτογραφιών που περιέχονται σε κάποιο λογαριασμό του** σε δύο ή περισσότερα σύνολα κάθε ένα εκ των οποίων θα ανήκει σε διαφορετικούς λογαριασμούς αυτού του χρήστη. Για παράδειγμα, κάποιος χρήστης μπορεί να επιθυμεί όλες οι φωτογραφίες που έχει τραβήξει στην Ήπειρο να τοποθετηθούν σε ένα νέο λογαριασμό.

Αναλυτική Περιγραφή Ζητούμενης Υλοποίησης

Για την υλοποίηση της παραπάνω υπηρεσίας θα χρειαστείτε τις ακόλουθες κεντρικές δομές δεδομένων:

1. Μία **μη-ταξινομημένη** απλά-συνδεδεμένη λίστα με κόμβο φρουρό που περιέχει τόσα στοιχεία όσοι και οι λογαριασμοί της υπηρεσίας, έναν για κάθε λογαριασμό. Αυτή η λίστα ονομάζεται ***λίστα λογαριασμών*** **(**`alist_p` **)** και παρουσιάζεται στο Σχήμα 1.



Σχήμα 1

2. Μία **ταξινομημένη, βάσει του πεδίου pid, διπλά συνδεδεμένη** λίστα με **όλες** τις φωτογραφίες που έχουν ανεβάσει όλοι οι χρήστες της υπηρεσίας. Αυτή η λίστα ονομάζεται ***λίστα φωτογραφιών*** (`plist_p`**)** και παρουσιάζεται στο Σχήμα 2.

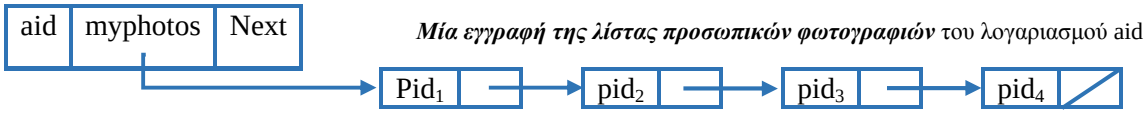
Σχήμα 2

Για την υλοποίηση των δομών αυτών χρειάζονται οι ακόλουθοι τύποι εγγραφών (structs):

1. Κάθε στοιχείο της ***λίστας λογαριασμών*** είναι μια εγγραφή (ένα struct) τύπου `account_t` με τα ακόλουθα πεδία:
   - **aid:** Αναγνωριστικό του λογαριασμού. Ένας μοναδικός αριθμός που αντιστοιχεί στο λογαριασμό.
   - **myphotos**: Δείκτης (τύπου `myphoto_t *`) στο πρώτο στοιχείο μιας **ταξινομημένης** (ως προς το πεδίο pid) **λίστας**, κάθε στοιχείο της οποίας αντιστοιχεί σε μια φωτογραφία που ανήκει στο λογαριασμό. Η λίστα αυτή ονομάζεται ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό aid. Κάθε στοιχείο της λίστας προσωπικών φωτογραφιών ενός λογαριασμού είναι μία εγγραφή (ένα struct) τύπου `myphoto_t` με τα ακόλουθα πεδία:
     - **pid**: Το αναγνωριστικό της φωτογραφίας.
     - **next**: Δείκτης (τύπου `myphoto_t *`) στον επόμενο κόμβο της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού.
   - **next**: Δείκτης (τύπου `account_t *`) στον επόμενο κόμβο της λίστας λογαριασμών.

   Μια εγγραφή τύπου `account_t` παρουσιάζεται στο Σχήμα 3.

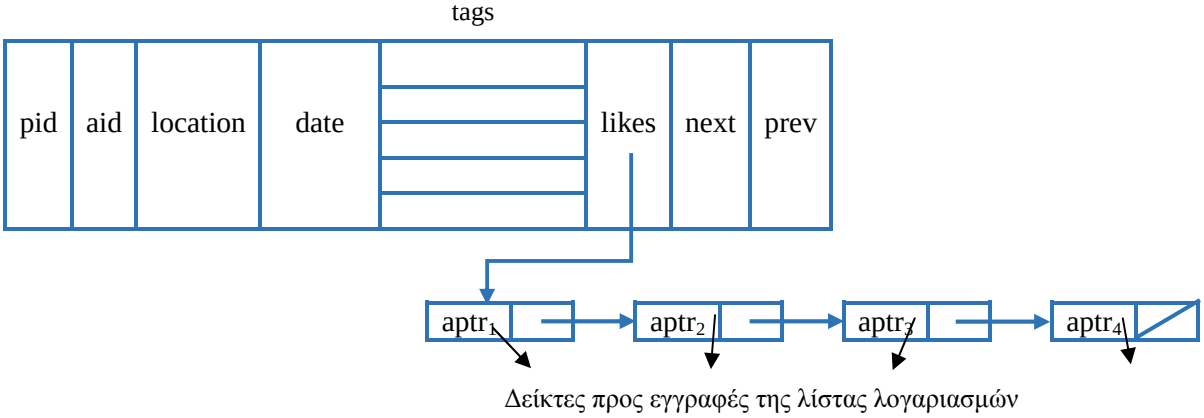
2. Κάθε στοιχείο της ***λίστας φωτογραφιών*** είναι μια εγγραφή (ένα struct) τύπου `photo_t` με τα ακόλουθα πεδία:



Σχήμα 3: Εγγραφή τύπου `account_t`.

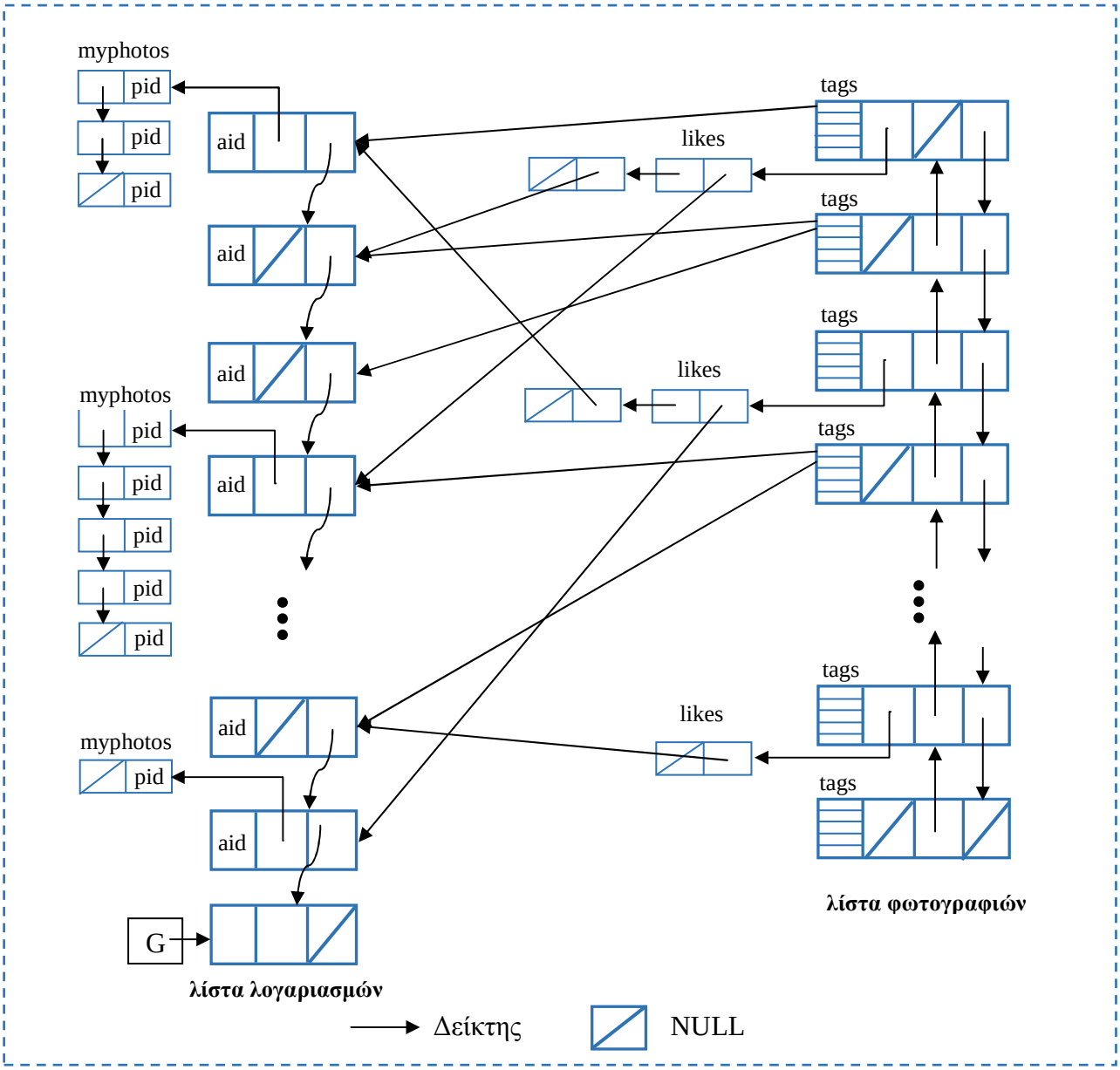
- **pid**: Αναγνωριστικό της φωτογραφίας. Είναι ένας μοναδικός αριθμός που αντιστοιχεί σε κάθε φωτογραφία.
- **aid**: Αναγνωριστικό του λογαριασμού στον οποίο ανήκει η φωτογραφία.
- **location**: Αναγνωριστικό της τοποθεσίας όπου τραβήχτηκε η φωτογραφία.
- **date**: Ένας αριθμός που αντιστοιχεί στην ημερομηνία στην οποία δημιουργήθηκε η φωτογραφία. Η ημερομηνία πρέπει να είναι της μορφής **YYYYMMDD**. Για παράδειγμα, η 28η Οκτωβρίου 2013 αναπαρίσταται ως 20131028 ενώ η 5η Ιανουαρίου 2011 αναπαρίσταται ως 20110105. Αυτή η αναπαράσταση μας επιτρέπει να συγκρίνουμε εύκολα ημερομηνίες για να δούμε πια προηγείται χρονικά. Για παράδειγμα το 20131028 είναι μεγαλύτερο του 20110105, άρα η 28η Οκτωβρίου 2013 έπεται της 5ης Ιανουαρίου 2011.
- **tags**: Πίνακας 5 θέσεων που ονομάζεται ***πίνακας επισημάνσεων***. Κάθε θέση του πίνακα περιέχει ένα δείκτη (τύπου `account_t *`) σε έναν κόμβο της λίστας λογαριασμών. Ο λογαριασμός αυτός αντιστοιχεί σε χρήστη που έχει επισημανθεί (γίνει tag) σε αυτή τη φωτογραφία.
- **likes**: Δείκτης στο πρώτο στοιχείο μιας λίστας, που ονομάζεται ***λίστα οπαδών*** της φωτογραφίας με αναγνωριστικό pid. Κάθε στοιχείο αυτής της λίστας είναι μία εγγραφή (ένα struct) τύπου `like_t` με τα ακόλουθα πεδία:
  - **aptr**: Δείκτης προς μια εγγραφή της λίστας λογαριασμών που αντιστοιχεί σε λογαριασμό που έχει δηλώσει πως του αρέσει η φωτογραφία (έχει κάνει like στην φωτογραφία).
  - **next**: Δείκτης (τύπου `like_t *`) στον επόμενο κόμβο της ***λίστα οπαδών*** της φωτογραφίας.
- **next**: Δείκτης (τύπου `photo_t *`) στον επόμενο κόμβο της ***λίστας φωτογραφιών***.
- **prev**: Δείκτης (τύπου `photo_t *`) στον προηγούμενο κόμβο της ***λίστας φωτογραφιών***.

Μια εγγραφή τύπου `photo_t` παρουσιάζεται στο Σχήμα 4.



Σχήμα 4: Εγγραφή τύπου `photo_t`.

Το Σχήμα 5 παρουσιάζει τις δομές της προγραμματιστικής εργασίας με συγκεντρωτικό τρόπο όπου, για λόγους απλότητας, στις εγγραφές (structs) τύπου `photo_t` δεν εμφανίζονται τα στατικά πεδία (εμφανίζονται μόνο τα πεδία που αποθηκεύουν δείκτες). Συγκεκριμένα, τα πεδία κάθε κόμβου της λίστας φωτογραφιών που εμφανίζονται στο Σχήμα 5 είναι τα εξής: tags, likes, prev και next με αυτή τη σειρά.



Σχήμα 5: Δομές δεδομένων πρώτης φάσης προγραμματιστικής εργασίας.

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) που περιέχει γεγονότα των ακόλουθων μορφών:

- **R <aid>**: Γεγονός τύπου *Register* το οποίο σηματοδοτεί τη δημιουργία ενός νέου λογαριασμού (`account_t`) στο σύστημα με αναγνωριστικό <aid> και κενή ***λίστα προσωπικών φωτογραφιών*** (myphotos). Το γεγονός αυτό πρέπει να δημιουργεί το νέο λογαριασμό και να τον προσθέτει στη ***λίστα λογαριασμών***. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
R <aid> DONE
```

- **D <aid>**: Γεγονός τύπου *Delete account* το οποίο σηματοδοτεί τη διαγραφή του λογαριασμού με αναγνωριστικό <aid>. Κατά τη διαγραφή ενός λογαριασμού πρέπει να διαγράφονται και όλες οι φωτογραφίες που ανήκουν στο λογαριασμό (δηλαδή όλα τα στοιχεία της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού, καθώς και οι αντίστοιχες φωτογραφίες από τη ***λίστα φωτογραφιών***). Επίσης, θα πρέπει να διαγραφούν όλες οι πιθανές αναφορές σε αυτόν το λογαριασμό μέσω επισημάνσεων (tags) ή/και likes. Συγκεκριμένα, θα πρέπει να εξεταστούν όλα τα στοιχεία της ***λίστας φωτογραφιών*** και να γίνονται κατάλληλες αλλαγές στον ***πίνακα επισημάνσεων*** κάθε τέτοιου στοιχείου όποτε χρειάζεται, καθώς και να εξεταστεί η ***λίστα οπαδών*** της φωτογραφίας που αντιστοιχεί στο στοιχείο: αν κάποιο στοιχείο της ***λίστας οπαδών*** της φωτογραφίας αυτής δείχνει στο λογαριασμό με αναγνωριστικό <aid>, το στοιχείο αυτό θα πρέπει επίσης να διαγραφεί. Για λόγους ευκολότερης αποσφαλμάτωσης του κώδικα που θα δημιουργηθεί, συνίσταται ισχυρά κατά την διαγραφή να αναθέτετε την τιμή NULL στο πεδίο aptr του προς διαγραφή στοιχείου. Τέλος θα πρέπει να αφαιρείται ο κόμβος που κρατάει τα στοιχεία για αυτόν το λογαριασμό από την ***λίστα λογαριασμών***. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
D <aid>
    MYPHOTOS = <pid1, pid2, ... pidn>
    TAGS     = <tpid1, tpid2, ... tpidk>
    LIKES    = <lpid1, lpid2, ... lpidm>
DONE
```

όπου n είναι το πλήθος των φωτογραφιών που ανήκουν στον λογαριασμό με αναγνωριστικό <aid>, k είναι το πλήθος των φωτογραφιών στις οποίες ο λογαριασμός είχε επισημανθεί (είχε γίνει tag), m είναι το πλήθος των φωτογραφιών των οποίων ο προς διαγραφή λογαριασμός είναι οπαδός και:

- για κάθε j, $1 \le j \le n$, $<pid_j>$ είναι το αναγνωριστικό της j-οστής φωτογραφίας που ανήκει στον προς διαγραφή λογαριασμό.
- για κάθε j, $1 \le j \le k$, $<tpid_j>$ είναι το αναγνωριστικό της j-οστής φωτογραφίας που περιείχε επισήμανση (tag) προς τον προς διαγραφή λογαριασμό. Σημειώνεται ότι από τις εγγραφές στη ***λίστα φωτογραφιών*** που αντιστοιχούσαν στις φωτογραφίες αυτές αφαιρέθηκαν οι εν λόγω επισημάνσεις (tags).

- για κάθε j, $1 \leq j \leq m$, <$lpid_j$> είναι το αναγνωριστικό της j-οστής φωτογραφίας που είχε οπαδό τον προς διαγραφή λογαριασμό. Σημειώνεται ότι από τις εγγραφές στη ***λίστα φωτογραφιών*** που αντιστοιχούσαν στις φωτογραφίες αυτές αφαιρέθηκαν τα εν λόγω likes.

- **U <pid> <aid> <location> <date>**: Γεγονός τύπου *Upload photo* το οποίο σηματοδοτεί την ανάρτηση (upload) μίας νέας φωτογραφίας στο σύστημα. Η νέα αυτή φωτογραφία θα έχει αναγνωριστικό <pid>. Το αναγνωριστικό του λογαριασμού που έκανε την ανάρτηση θα είναι <aid>, η τοποθεσία <location>, η ημερομηνία λήψης <date>, η ***λίστα οπαδών*** της φωτογραφίας (likes) θα είναι κενή και ο ***πίνακας επισημάνσεων*** άδειος. Η νέα φωτογραφία πρέπει να τοποθετηθεί στη σωστή θέση της ***λίστας φωτογραφιών***. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
U <pid> <aid> <location> <date> DONE
PREDG = <pidG>, SUCCG = <sidG>, PREDL = <pidL>, SUCCL = <sidL>
```

όπου:

- <pidG> και <sidG> είναι τα αναγνωριστικά της προηγούμενης και της επόμενης φωτογραφίας, αντίστοιχα, της προς εισαγωγή φωτογραφίας στη ***λίστα φωτογραφιών***.
- <pidL> και <sidL> είναι τα αναγνωριστικά της προηγούμενης και της επόμενης φωτογραφίας, αντίστοιχα, της προς εισαγωγή φωτογραφίας στη ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid>.

- **B <pid>**: Γεγονός τύπου *Burn photo* το οποίο σηματοδοτεί την αφαίρεση της φωτογραφίας με αναγνωριστικό <pid> από το σύστημα. Κατά την αφαίρεση μίας φωτογραφίας από το σύστημα πρέπει να διαγράφονται και όλες οι πληροφορίες σχετικά με αυτήν και επομένως και όλα τα στοιχεία της λίστας οπαδών της φωτογραφίας αυτής. Ακόμη θα πρέπει να διαγράφεται ο αντίστοιχος κόμβος από την ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού που ανέβασε την προς-διαγραφή φωτογραφία. Επίσης, για ευκολότερη αποσφαλμάτωση του κώδικα που θα δημιουργήσετε συνίσταται ισχυρά να αναθέτετε την τιμή NULL σε κάθε στοιχείο του ***πίνακα επισημάνσεων*** της εγγραφής που αντιστοιχεί στην προς-διαγραφή φωτογραφία. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
B <pid>
  <aid> LIKES = <aid1, aid2, ... aidn>
DONE PREDG = <pidG>, SUCCG = <sidG> PREDL = <pidL> SUCCL = <sidL>
```

όπου n είναι το μέγεθος της ***λίστας οπαδών*** της φωτογραφίας με αναγνωριστικό <pid>:

- <aid> είναι το αναγνωριστικό του λογαριασμού στον οποίο ανήκει η προς διαγραφή φωτογραφία,
- για κάθε j, $1 \leq j \leq n$, <aid_j> είναι το αναγνωριστικό του λογαριασμού στον οποίο δείχνει το j-οστό στοιχείο της λίστας οπαδών της προς διαγραφή φωτογραφίας.
- <pidG> και <sidG> είναι τα αναγνωριστικά της προηγούμενης και της επόμενης φωτογραφίας, αντίστοιχα, της προς διαγραφή φωτογραφίας στη λίστα των φωτογραφιών.

- <pidL> και <sidL> είναι τα αναγνωριστικά της προηγούμενης και της επόμενης φωτογραφίας, αντίστοιχα, της προς διαγραφή φωτογραφίας στη ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid>.

- **L <aid> <pid>**: Γεγονός τύπου *Like photo* το οποίο σηματοδοτεί την προσθήκη ενός κόμβου στη ***λίστα οπαδών*** της φωτογραφίας με αναγνωριστικό <pid>. Ο κόμβος αυτός θα αποθηκεύει έναν δείκτη προς το στοιχείο με αναγνωριστικό <aid> της λίστας λογαριασμών. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
L <aid> <pid> DONE
```

- **T <aid> <pid>**: Γεγονός τύπου *Tag* το οποίο σηματοδοτεί την προσθήκη ενός δείκτη στον λογαριασμό με αναγνωριστικό <aid> στον ***πίνακα επισημάνσεων*** της φωτογραφίας με αναγνωριστικό <pid>. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
T <aid> <pid> DONE
```

- **M <aid1> <aid2> <aid3>**: Γεγονός τύπου *Merge accounts* το οποίο σηματοδοτεί τη συνένωση των λογαριασμών με αναγνωριστικά <aid1> και <aid2> σε ένα νέο λογαριασμό με αναγνωριστικό <aid3>. Κατά τη συνένωση θα πρέπει:

  - να συνενωθούν οι δύο λίστες προσωπικών φωτογραφιών σε μία νέα ταξινομημένη λίστα η οποία θα αποτελέσει την λίστα προσωπικών φωτογραφιών για το νέο λογαριασμό. Αυτό θα πρέπει να επιτευχθεί σε χρόνο O(n1+n2), όπου n1 και n2 είναι το πλήθος των στοιχείων στις λίστες προσωπικών φωτογραφιών των λογαριασμών <aid1> και <aid2>, αντίστοιχα,

  - να αντικατασταθούν οποιεσδήποτε αναφορές προς τους λογαριασμούς με αναγνωριστικά <aid1> και <aid2> με αναφορές στον λογαριασμό με αναγνωριστικό <aid3>. Αναφορές μπορεί να υπάρχουν στις ***λίστες οπαδών*** και στους ***πίνακες επισημάνσεων*** των διαφόρων φωτογραφιών. Στην περίπτωση όπου και οι δύο λογαριασμοί έχουν επισημανθεί ή/και έχουν δηλώσει την αρέσκεια τους (like) για μία φωτογραφία, καλό θα ήταν να αντικαθίσταται η πρώτη αναφορά με μία αναφορά στον λογαριασμό με αναγνωριστικό <aid3> και η δεύτερη να διαγράφεται. Όποτε αν μία φωτογραφία είχε δύο like από του λογαριασμούς με αναγνωριστικά <aid1> και <aid2>, μετά το πέρας της εκτέλεσης του γεγονότος θα πρέπει να έχει ένα like από τον λογαριασμό με αναγνωριστικό <aid3>.

  - να διαγραφούν οι λογαριασμοί με αναγνωριστικά <aid1> και <aid2> από την ***λίστα λογαριασμών.***

  Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
M <aid1> <aid2> <aid3>
      MYPHOTOS1 = <m1pid1, m1pid2, … , m1pidm>
      MYPHOTOS2 = <m2pid1, m2pid2, … , m2pidn>
      MYPHOTOS3 = <m3pid1, m3pid2, … , m3pidk>
      UPDATED  = <upid1, upid2, … upidr>
DONE
```

  όπου:

  - m είναι το μέγεθος της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό

<aid1>, n είναι το μέγεθος της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid2>, k είναι το μέγεθος της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid3> και r είναι το πλήθος των φωτογραφιών για τις οποίες ισχύει πως κάποιο στοιχείο του πίνακα επισημάνσεων τους ή κάποιο στοιχείο της λίστας οπαδών τους ενημερώθηκε μετά τη συνένωση και:

- για κάθε j , $1 \le j \le m$, <m1pid$_j$> είναι το αναγνωριστικό του λογαριασμού στον οποίο δείχνει το j-οστό στοιχείο της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid1>,
- για κάθε j , $1 \le j \le n$, <m2pid$_j$> είναι το αναγνωριστικό του λογαριασμού στον οποίο δείχνει το j-οστό στοιχείο της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid2>,
- για κάθε j , $1 \le j \le k$, <m3pid$_j$> είναι το αναγνωριστικό του λογαριασμού στον οποίο δείχνει το j-οστό στοιχείο της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid3>,
- για κάθε j , $1 \le j \le r$, <upid$_j$> είναι το αναγνωριστικό της φωτογραφίας της οποίας ανανεώθηκε η ***λίστα οπαδών*** της ή/και ο ***πίνακας επισημάνσεων*** της λόγω της συνένωσης των λογαριασμών.

- **S <aid1> <aid2> <aid3> <location>**: Γεγονός τύπου *Split account* το οποίο σηματοδοτεί το διαχωρισμό του λογαριασμού με αναγνωριστικό <aid1> σε δύο νέους λογαριασμούς με αναγνωριστικά <aid2> (για τον πρώτο από αυτούς) και <aid3> (για το δεύτερο). Όσες φωτογραφίες έχουν ληφθεί στη τοποθεσία <location> θα πρέπει να εισαχθούν (**ταξινομημένες βάσει του πεδίου pid**) στην ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid3> και όλες οι υπόλοιπες στην ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid2>. Ακόμη θα πρέπει να ενημερώνεται κατάλληλα και το πεδίο <aid> των στοιχείων της λίστας φωτογραφιών που αντιστοιχούν σε κάθε φωτογραφία που περιέχεται στις λίστες αυτές. Τυχόν αναφορές στον λογαριασμό με αναγνωριστικό <aid1> αντικαθίστανται με αναφορές στον λογαριασμό με αναγνωριστικό <aid2>. Αναφορές μπορεί να υπάρχουν στις ***λίστες οπαδών*** των στοιχείων της λίστας φωτογραφιών και στους ***πίνακες επισημάνσεων*** των στοιχείων αυτών. Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
S <aid1> <aid2> <aid3> <location> DONE
      MYPHOTOS1 = <m1pid_1, m1pid_2, … , m1pid_m>
      MYPHOTOS2 = <m2pid_1, m2pid_2, … , m2pid_n>
      MYPHOTOS3 = <m3pid_1, m3pid_2, … , m3pid_k>
      UPDATED   = <upid_1, upid_2, … upid_r>
```

όπου:

- m είναι το μέγεθος της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid1>, n είναι το μέγεθος της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid2>, k είναι το μέγεθος της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid3> και r είναι το πλήθος των φωτογραφιών για τις οποίες ισχύει πως κάποιο στοιχείο του πίνακα επισημάνσεων τους ή κάποιο στοιχείο της λίστας οπαδών τους ενημερώθηκε μετά τη συνένωση και:
- για κάθε j , $1 \le j \le m$, <m1pid$_j$> είναι το αναγνωριστικό του λογαριασμού στον οποίο δείχνει το j-οστό στοιχείο της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid1>,
- για κάθε j , $1 \le j \le n$, <m2pid$_j$> είναι το αναγνωριστικό του λογαριασμού στον οποίο δείχνει το j-οστό στοιχείο της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid2>,
- για κάθε j , $1 \le j \le k$, <m3pid$_j$> είναι το αναγνωριστικό του λογαριασμού στον οποίο δείχνει το j-οστό στοιχείο της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό

<aid3>,

- για κάθε j , $1 \leq j \leq r$, > <$upid_j$> είναι το αναγνωριστικό της φωτογραφίας της οποίας ανανεώθηκε η ***λίστα οπαδών*** ή/και ο ***πίνακας επισημάνσεων*** της λόγω του διαχωρισμού του λογαριασμού με αναγνωριστικό <aid1>.

- **F <YYYYMMDD>:** Γεγονός τύπου *Find* το οποίο σηματοδοτεί την αναζήτηση **όλων των** φωτογραφιών που έχουν ληφθεί την ημερομηνία **<YYYYMMDD>**. Μετά το πέρας του γεγονότος θα πρέπει να τυπώνεται η παρακάτω πληροφορία:

```
F <YYYYMMDD>
    <pid1>
        TAGS:  <a1_id1> <a1_id2> <a1_id3> <a1_id4> <a1_id5>
        LIKES: <l1_aid1> <l1_aid2> … <l1_aidm1>
        LOCATION:  <location1>
        TIMESTAMP: <YYYYMMDD>
    <pid2>
        TAGS:  <a2_id1> <a2_id2> <a2_id3> <a2_id4> <a2_id5>
        LIKES: <l2_aid1> <l2_aid2> … <l2_aidm2>
        LOCATION:  <location2>
        TIMESTAMP: <YYYYMMDD>
        …
    <pidn>
        TAGS:  <an_id1> <an_id2> <an_id3> <an_id4> <an_id5>
        LIKES: <ln_aid1> <ln_aid2> … <ln_aidmn>
        LOCATION:  <locationn>
        TIMESTAMP: <YYYYMMDD>
DONE
```

όπου n είναι το πλήθος των φωτογραφιών που έχουν ληφθεί την ημερομηνία <YYYMMDD> και για κάθε i, $1 \leq i \leq n$:

- για κάθε j , $1 \leq j \leq 5$, > <a<i>_id_j> είναι το αναγνωριστικό του λογαριασμού ο οποίος έχει επισημανθεί στην φωτογραφία με αναγνωριστικό <pid_i> (δηλαδή υπάρχει δείκτης προς τον λογαριασμό αυτό στον ***πίνακα επισημάνσεων*** του στοιχείου που αντιστοιχεί στη φωτογραφία στη ***λίστα φωτογραφιών***)
- m_i είναι το μέγεθος της λίστας οπαδών της φωτογραφίας με αναγνωριστικό <pid_i> και για κάθε j , $1 \leq j \leq m_i$, > <l<i>_aid_j> είναι το αναγνωριστικό του j-οστού στοιχείου της λίστας οπαδών της φωτογραφίας με αναγνωριστικό <pidi>
- <$location_i$> είναι η τοποθεσία στην οποία ελήφθη η φωτογραφία με αναγνωριστικό <pid_i>.

- **P <aid>**: Γεγονός τύπου *Print* το οποίο σηματοδοτεί το τύπωμα των στοιχείων της ***λίστας προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid>. Για κάθε τέτοια φωτογραφία θα πρέπει επιπρόσθετα να τυπώνονται και οι πληροφορίες που αφορούν τη φωτογραφία, όπως οι επισημάνσεις της, η τοποθεσία και η ημερομηνία που ελήφθη η φωτογραφία, καθώς και τα στοιχεία της ***λίστας οπαδών*** της. Σε αυτό το γεγονός το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
P <aid>
    <pid1>
        TAGS:  <a1_id1> <a1_id2> <a1_id3> <a1_id4> <a1_id5>
        LIKES: <l1_aid1> <l1_aid2> … <l1_aaidm1>
        LOCATION:  <location1>
        TIMESTAMP: <YYYYMMDD1>
    <pid2>
        TAGS:  <a2_id1> <a2_id2> <a2_id3> <a2_id4> <a2_id5>
        LIKES: <l2_aid1> <l2_aid2> … <l2_aidm2>
        LOCATION:  <location2>
        TIMESTAMP: <YYYYMMDD2>
        …
    <pidn>
        TAGS:  <an_id1> <an_id2> <an_id3> <an_id4> <an_id5>
        LIKES: <ln_aid1> <ln_aid2> … <ln_aidmn>
        LOCATION:  <locationn>
        TIMESTAMP: <YYYYMMDDn>
DONE
```

όπου n είναι το πλήθος των στοιχείων στη ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid>και για κάθε i, $1 \leq i \leq n$:

- για κάθε j , $1 \leq j \leq 5$, > <a<i>_id_j> είναι το αναγνωριστικό του λογαριασμού ο οποίος έχει επισημανθεί στην φωτογραφία με αναγνωριστικό <pid_i>
- m_i είναι το μέγεθος της ***λίστας οπαδών*** της φωτογραφίας με αναγνωριστικό <pid_i> και για κάθε j, $1 \leq j \leq m_n$, > <l<i>_aid_j> είναι το αναγνωριστικό του j-οστού στοιχείου της ***λίστας οπαδών*** της φωτογραφίας με αναγνωριστικό <pid_i>, και
- <$location_i$> είναι η τοποθεσία στην οποία ελήφθη η φωτογραφία με αναγνωριστικό <pid_i>,
- <$YYYYMMDD_i$> είναι η ημερομηνία στην οποία ελήφθη η φωτογραφία με αναγνωριστικό <pid_i> (στη μορφή που περιγράφεται παραπάνω).

- **A**: Γεγονός τύπου *Acounts print* το οποίο σηματοδοτεί το τύπωμα των στοιχείων της ***λίστας λογαριασμών*** και της λίστας ***προσωπικών φωτογραφιών*** που περιέχεται σε κάθε λογαριασμό. Σε αυτό το γεγονός το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
A
    <aid1>
        MYPHOTOS:  <p1_id1> <p1_id2> ... <p1_idm1>
    <aid2>
        MYPHOTOS:  <p2_id1> <p2_id2>... <p2_idm2>
    ...
    <aidn>
        MYPHOTOS:  <pn_id1> <pn_id2>... <pn_idmn>
DONE
```

όπου n είναι το πλήθος των στοιχείων στη ***λίστα λογαριασμών*** και για κάθε i, $1 \leq i \leq n$::

- $<aid_i>$ είναι το αναγνωριστικό του i-οστού λογαριασμού στη ***λίστα λογαριασμών***,
- m_i είναι το μέγεθος της ***λίστας των προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό $<aid_i>$ και για κάθε j , $1 \leq j \leq m_i$, > $<p<i>\_id_j>$ είναι το αναγνωριστικό της j-οστής φωτογραφίας στη ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό $<aid_i>$.

- **E**: Γεγονός τύπου *Expose photos* το οποίο σηματοδοτεί το τύπωμα των στοιχείων της ***λίστας φωτογραφιών.*** Για κάθε φωτογραφία θα πρέπει να τυπώνονται και οι πληροφορίες που αφορούν τη φωτογραφία, όπως οι επισημάνσεις της, η τοποθεσία και η ημερομηνία που ελήφθη η φωτογραφία, καθώς και τα στοιχεία της ***λίστας οπαδών*** της. Σε αυτό το γεγονός το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
E
    <pid1>
        TAGS:  <a1_id1> <a1_id2> <a1_id3> <a1_id4> <a1_id5>
        LIKES: <l1_aid1> <l1_aid2> … <l1_aaidm1>
        LOCATION:  <location1>
        TIMESTAMP: <YYYYMMDD1>
    <pid2>
        TAGS:  <a2_id1> <a2_id2> <a2_id3> <a2_id4> <a2_id5>
        LIKES: <l2_aid1> <l2_aid2> … <l2_aidm2>
        LOCATION:  <location2>
        TIMESTAMP: <YYYYMMDD2>
        …
    <pidn>
        TAGS:  <an_id1> <an_id2> <an_id3> <an_id4> <an_id5>
        LIKES: <ln_aid1> <ln_aid2> … <ln_aidmn>
        LOCATION:  <locationn>
        TIMESTAMP: <YYYYMMDDn>
DONE
```

όπου n είναι το πλήθος των στοιχείων στη λίστα φωτογραφιών και για κάθε i, $1 \leq i \leq n$:

- για κάθε j , $1 \leq j \leq 5$, > $<a<i>\_id_j>$ είναι το αναγνωριστικό του λογαριασμού ο οποίος έχει επισημανθεί στην φωτογραφία με αναγνωριστικό $<pid_i>$

- m_i είναι το μέγεθος της λίστας οπαδών της φωτογραφίας με αναγνωριστικό <pid_i> και για κάθε j, $1 \leq j \leq m_n$, > <l<i>_aid_j> είναι το αναγνωριστικό του j-οστού στοιχείου της λίστας οπαδών της φωτογραφίας με αναγνωριστικό <pid_i>,
- <$location_i$> είναι η τοποθεσία στην οποία ελήφθη η φωτογραφία με αναγνωριστικό <pid_i>,
- <$YYYYMMDD_i$> είναι η ημερομηνία στην οποία ελήφθη η φωτογραφία με αναγνωριστικό <pid_i> (στην μορφή που περιγράφεται παραπάνω).

- **W**: Γεγονός τύπου *World Print* το οποίο σηματοδοτεί το τύπωμα όλων των δομών δεδομένων που υπάρχουν στο σύστημα. Σε αυτό το γεγονός το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

```
W
    ACCOUNTS:
    <aid1>
        MYPHOTOS:  <p1_id1> <p1_id2>... <p1_idm1>
    <aid2>
        MYPHOTOS:  <p2_id1> <p2_id2>... <p2_idm2>
    ...
    <aidn>
        MYPHOTOS:  <pn_id1> <pn_id2>... <pn_idmn>

    PHOTOS:
    <pid1>
        TAGS:  <a1_id1> <a1_id2> <a1_id3> <a1_id4> <a1_id5>
        LIKES: <l1_aid1> <l1_aid2> ... <l1_aidk1>
        LOCATION:  <location1>
        TIMESTAMP: <YYYYMMDD1>
    <pid2>
        TAGS:  <a2_id1> <a2_id2> <a2_id3> <a2_id4> <a2_id5>
        LIKES: <l2_aid1> <l2_aid2> ... <l2_aidk2>
        LOCATION:  <location2>
        TIMESTAMP: <YYYYMMDD2>
        …
    <pidr>
        TAGS:  <an_id1> <an_id2> <an_id3> <an_id4> <an_id5>
        LIKES: <ln_aid1> <ln_aid2> ... <ln_aidkr>
        LOCATION:  <locationr>
        TIMESTAMP: <YYYYMMDDr>
DONE
```

όπου n είναι το πλήθος των στοιχείων στη ***λίστα λογαριασμών*** και r είναι το πλήθος των στοιχείων στη ***λίστα φωτογραφιών*** και:

- για κάθε i , $1 \le i \le n$, > <aid_i> είναι το αναγνωριστικό του i-οστού λογαριασμού στη ***λίστα λογαριασμών***, m_i είναι το πλήθος των στοιχείων στη ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid_i> και για κάθε j , $1 \le j \le m_i$, > <p<i>_id_j> είναι το αναγνωριστικό της j-οστής φωτογραφίας που υπάρχει στη ***λίστα προσωπικών φωτογραφιών*** του λογαριασμού με αναγνωριστικό <aid_i>,

- για κάθε i , $1 \le i \le r$, > (1) <pid_i> είναι το αναγνωριστικό της i-οστής φωτογραφίας στη ***λίστα φωτογραφιών***, (2) για κάθε j , $1 \le j \le 5$, > <a<i>_id_j> είναι το αναγνωριστικό του λογαριασμού ο οποίος δεικτοδοτείται από τη j-οστή θέση του ***πίνακα επισημάνσεων*** της φωτογραφίας με αναγνωριστικό <pid_i>, (3) m_i είναι το μέγεθος της ***λίστας οπαδών*** της φωτογραφίας με αναγνωριστικό <pid_i>, (4) για κάθε c, $1 \le c \le m_i$, > <l<i>_aid_c> είναι το αναγνωριστικό του λογαριασμού που δεικτοδοτείται από το j-οστό στοιχείο της ***λίστας οπαδών*** της φωτογραφίας με αναγνωριστικό <pid_i>, (5) <$location_i$> είναι η τοποθεσία στην οποία ελήφθη η φωτογραφία με αναγνωριστικό <pid_i> και (6) <$YYYYMMDD_i$> είναι η ημερομηνία στην οποία ελήφθη η φωτογραφία με αναγνωριστικό <pid_i> (στην μορφή που περιγράφεται παραπάνω).

Δομές Δεδομένων

Στη συνέχεια παρουσιάζονται οι δομές σε C που πρέπει να χρησιμοποιηθούν για την υλοποίηση της παρούσας εργασίας.

```
////////////////////////////////////////////////////////////////////////////
//                  Defining the needed structures
////////////////////////////////////////////////////////////////////////////

/** Enum and array doing the location to integer mapping */
typedef enum {
        attica                  = 1,
        central_greece          = 2,
        central_macedonia       = 3,
        crete                   = 4,
        east_macedonia_and_thrace = 5,
        epirus                  = 6,
        ionian_islands          = 7,
        north_aegean            = 8,
        peloponnese             = 9,
        south_aegean            = 10,
        thessaly                = 11,
        west_greece             = 12,
        west_macedonia          = 13
} location_e;

/**
 * Structure defining a node of the myphotos list (lista prosopikwn fwtografiwn)
 */
typedef struct myphoto {
        unsigned int    pid;  /**< The photo identifier. >0 */
        struct myphoto  *next;/**< Pointer to the next node in the myphotos list */
} myphoto_t;

/**
 * Structure defining a node of the account list (lista logariasmwn)
 */
typedef struct account {
        unsigned int    aid;      /**< The account's identifier  */
        myphoto_t       *myphotos;/**< A list with the photos uploaded by
                                   * this account */
        struct account  *next;    /**< Pointer to the next node of the
                                   * accounts list */
} account_t;

/**
 * Structure defining a node of the likes list (lista opadwn)
 */
typedef struct like {
        account_t       *aptr; /**< The account's identifier that liked
                                * the photo. >0 */
        struct like     *next; /**< Pointer to the next node in the likes list */
} like_t;
```

```
/**
 * Structure defining a node of the photos list (lista fwtografiwn)
 */
typedef struct photo {
        unsigned int    pid;            /**< The photo identifier. >0 */
        unsigned int    aid;            /**< The uploader's account identifier  */
        location_e      location;       /**< The location where this photo
                                         * was captured */
        unsigned int    date;           /**< The date the photo was
                                         * captured. (YYYYMMDD) */
        account_t       *tags[MAX_TAGS];/**< An array with pointers to the
                                         * tagged accounts in this
                                         * photo */
        like_t          *likes;         /**< A list with the accounts that
                                         * like the photo */
        struct photo    *next;          /**< Pointer to the next node of
                                         * the photos list */
        struct photo    *prev;          /**< Pointer to the previous node
                                         * of the photos list */
} photo_t;

// For simplicity we define the two major data structures as global variables
account_t* alist_p;   /**< The accounts list (lista logariasmwn) */
photo_t*   plist_p;   /**< The photos list (lista fwtografiwn)  */
////////////////////////////////////////////////////////////////////////////////
```

Συμβουλές

Για λόγους ευκολότερης αποσφαλμάτωσης του κώδικα που θα δημιουργήσετε, συνίσταται ισχυρά κατά την διαγραφή κόμβων να αναθέτετε την τιμή NULL στους δείκτες του προς διαγραφή στοιχείου. Ακόμη συνίσταται να αναθέτεται την τιμή 0 στα υπόλοιπα πεδία που είναι τύπου (unsigned int).