ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
1η Εργασία - Τμήμα: Περιττών Αριθμών Μητρώου
K22: Λειτουργικά Συστήματα – Χειμερινό Εξάμηνο '20
Ημερομηνία Ανακοίνωσης: Τρίτη 29 Σεπτεμβρίου 2020
Ημερομηνία Υποβολής: Σάββατο 17 Οκτωβρίου 2020 Ώρα 23:59

Εισαγωγή στην Εργασία:

Ο στόχος αυτής της εργασίας είναι να εξοικειωθείτε με το περιβάλλον `Linux` και σχετικά εργαλεία προγραμματισμού και ανάπτυξης λογισμικού.

Θα υλοποιήσετε ένα πρόγραμμα με όνομα mngstd, που με την βοήθεια μιας σύνθεσης δομών θα επιτρέπει την εξαγωγή πληροφοριών και στατιστικών στοιχείων για την γραμματεία του τμήματος. Θα πρέπει να υλοποιήσετε μια δομή πάνω στην οποία θα κάνετε εισαγωγές/διαγραφές αλλά και αναζητήσεις σε εγγραφές φοιτητών. Οι λειτουργίες αυτές θα πρέπει εν γένει να έχουν κόστος προσπέλασης *O(1)*. Κάτι τέτοιο μπορείτε να το επιτύχετε όταν χρησιμοποιείτε δομή κατακερματισμού (hashing). Επίσης, θα πρέπει να μπορείτε δυναμικά να απαντήσετε επερωτήσεις που έχουν να κάνουν με συγκεκριμένες ομάδες δεδομένων χρησιμοποιώντας μια δομή τύπου ανεστραμμένου καταλόγου (inverted index).

Μερικές βασικές προϋποθέσεις για την παραπάνω εφαρμογή είναι οι εξής:

1. Η βασική δομή δεδομένων είναι το *hashing* και οργανώνει τις εγγραφές των φοιτητών σύμφωνα με το StudentID που λειτουργεί σαν κλειδί.
2. Η δομή δέχεται εισαγωγές, διαγραφές και επερωτήσεις.
3. Η εγγραφή κάθε φοιτήτριας/φοιτητή αποτελείται από το StudentID της/του (κλειδί), όνομα, επίθετο, Τ.Κ. πόλης μόνιμης κατοικίας, έτος 1ης εγγραφής και μέσο όρο μαθημάτων μέχρι στιγμής που έχει πάρει μέχρι στιγμής.
4. Ταυτόχρονα η εφαρμογή χρησιμοποιεί έναν *inverted index* ώστε να ομαδοποιεί τις εγγραφές που υπάρχουν ανά έτος φοίτησης ώστε να μπορούμε να ρωτάμε την δομή για χαρακτηριστικά και μεγέθη που αφορούν στην συγκεκριμένη ομάδα.
5. Η εφαρμογή θα πρέπει να μπορεί να διαβάσει δεδομένα και στην εκκίνηση της από ένα αρχείο εισόδου.
6. Το πρόγραμμα σας θα πρέπει –όποτε αυτό απαιτείται– να ελευθερώνει όλη την μνήμη που έχει δεσμεύσει. Το ίδιο ισχύει στον τερματισμό της εφαρμογής.

Μέθοδοι προσπέλασης σαν και αυτές που αναφέρονται παραπάνω είναι πολύ κοινές σε υλοποιήσεις συστημάτων ανάκτησης πληροφορίας και μηχανών αναζήτησης.

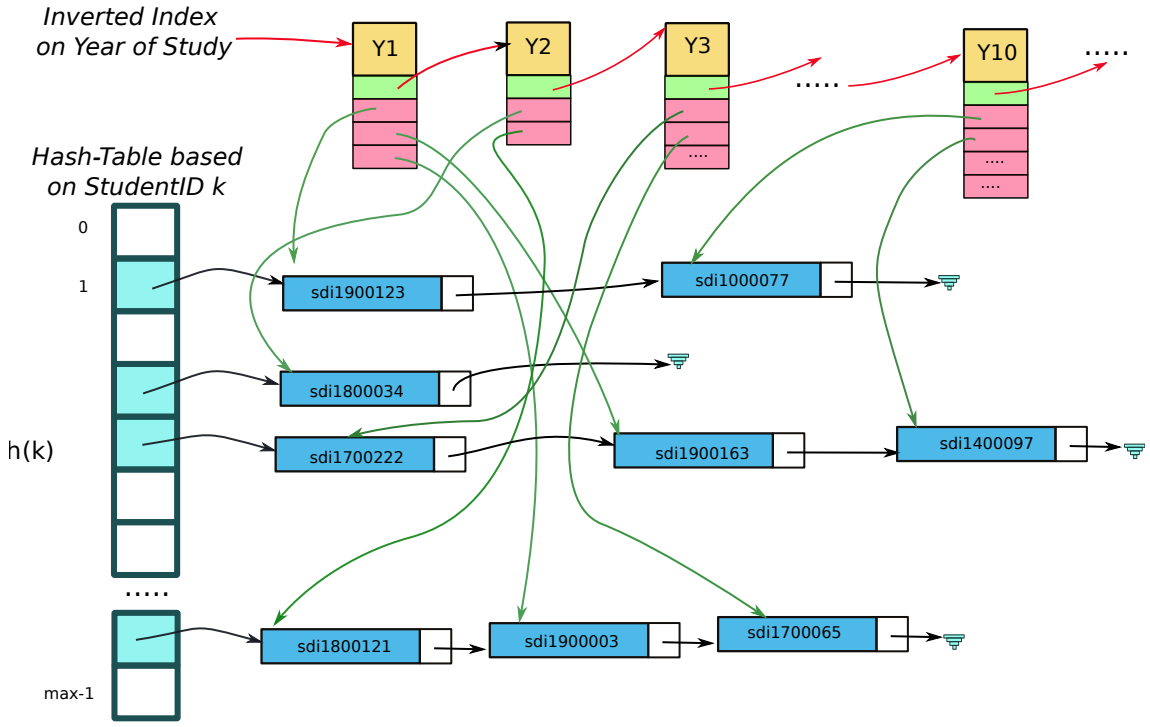
Διαδικαστικά:

Το προγραμμά σας θα πρέπει να γραφτεί σε *C* (ή *C++* αν θέλετε αλλά χωρίς την χρήση STL/Templates) και να τρέχει στις μηχανές `Linux workstations` του τμήματος.

Ο πηγαίος κώδικας σας (source code) πρέπει να αποτελείται από ***τουλάχιστον δυο*** (και κατά προτίμηση πιο πολλά) διαφορετικά αρχεία και θα πρέπει *απαραιτήτως να γίνεται χρήση* separate compilation .

Παρακολουθείτε την ιστοσελίδα του μαθήματος `http://www.di.uoa.gr/~ad/k22/` για επιπρόσθετες ανακοινώσεις αλλά και την ηλεκτρονική-λίστα (η-λίστα) του μαθήματος στο URL `https://piazza.com/uoa.gr/fall2020/k22/home`

- Υπεύθυνοι για την άσκηση αυτή (ερωτήσεις, αξιολόγηση, βαθμολόγηση κλπ.) είναι ο κ. Ξενοφών Κίτσιος `cs2190022+AT-di`, ο κ. Αλέξανδρος Κυριακού `cs2190001+AT-di`, και ο κ. Χρήστος Παπαδόπουλος `cs3190009+AT-di`.



Σχήμα 1: Παράδειγμα πιθανής οργάνωσης δομών για την εφαρμογή `mngstd`

- Στην διάρκεια της 1ης εβδομάδας του εξαμήνου θα πρέπει να 'εγγραφείτε' στο μάθημα στη πλατφόρμα `https://eclass.uoa.gr`

  Με αυτό το τρόπο μπορούμε να γνωρίζουμε ποιος/-α προτίθεται να υποβάλλει την πρώτη άσκηση *ώστε να προβούμε στις κατάλληλες ενέργειες για την υποβολή της εργασίας σας*.
- Με βάση την παραπάνω λίστα, θα σας γράψουμε επίσης στο σύστημα `piazza.com` που βρίσκεται στο `https://piazza.com/uoa.gr/fall2020/k22/home`. Μόλις *αποδεχτείτε* ένα 'κωδικό' που θα σας σταλεί, μπορείτε να κάνετε ερωτήσεις και να δείτε απαντήσεις ή/και διευκρινήσεις που δίνονται σχετικά με την άσκηση.

Σύνθεση Δομών για το mngstd:

Το Σχήμα 1 δείχνει μια μερική αλλά αντιπροσωπευτική κατάσταση της σύνθετης δομής που θα δημιουργήσετε.

Οι εγγραφές φοιτητών είναι οργανωμένες με την βοήθεια ενός πίνακα κατακερματισμού (στο αριστερό κομμάτι του σχήματος). Η προσπέλαση στο πίνακα κατακερματισμού γίνεται με την βοήθεια μίας συνάρτησης κατακερματισμού $h(k)$ όπου k είναι το κλειδί `studentID` [1]. Κάθε εγγραφή έχει όλα τα στοιχεία πληροφορίας για την φοιτήτρια/φοιτητή όπως `studentID`, όνομα, επώνυμο, μέσο όρο, έτος εγγραφής, Τ.Κ. πόλης διαμονής, κλπ.

Η δομή του ανεστραμμένου ευρετηρίου οργανώνει την προσπέλαση στις εγγραφές με *ανά έτος σπουδών* που βρίσκεται ο κάθε φοιτητής και μπορεί να υλοποιηθεί με χρήση δυναμικά διασυνδεδεμένων λιστών. Έτσι το στοιχείο του ανεστραμμένου ευρετηρίου για τους πρωτοετείς στο Σχήμα 1 διαθέτει 3 εγγραφές με τις οποίες είναι άμεσα συνδεδεμένο.

Το μέγεθος του πίνακα κατακερματισμού ορίζεται στην αρχή της εκτέλεσης της εφαρμογής και έχει να κάνει

[1] Η δομή είναι απλή δομή κατακερματισμού και δεν έχει σχέση με 'δυναμικό κατακερματισμό'.

με το πλήθος των εγγραφών που αναμένεται να εισαχθούν στην δομή και παραμένει σταθερό κατά την διάρκεια εκτέλεσης του προγράμματος. Κάλλιστα μπορείτε να ελέγξετε τον αριθμό των εισαγωγών που γίνονται από το αρχείο εισόδου και να ορίσετε τα σχετικά μεγέθη του πίνακα κατακερματισμού ώστε να έχετε μικρό αριθμό από προσκρούσεις (collisions).

Η συγκεκριμένη μορφή που παίρνουν ο πίνακας καταμερισμού και ο ανεστραμμένος κατάλογος έχουν να κάνουν με επιλογές σχεδιασμού που θα πρέπει να πάρετε και να περιγράψετε στο σύντομο αρχείο σχεδιασμού που θα υποβάλετε μαζί με τον κώδικα σας. Οι δομές που τελικά θα χρησιμοποιήσετε θα πρέπει να είναι *δυναμικές*.

Γραμμή Κλήσης της Εφαρμογής:

Η εφαρμογή μπορεί να κληθεί με τον παρακάτω αυστηρό τρόπο στην γραμμή εντολής (`tty`):
`./mngstd -i inputfile -c configfile`
όπου

- `mngstd` είναι το εκτελέσιμο που θα δημιουργήσετε,
- `inputfile` είναι το αρχείο εισόδου δεδομένων. Το αρχείο περιέχει μια εγγραφή για κάθε φοιτητή στο σύστημα. Ο αριθμός των εγγραφών δεν είναι προκαθορισμένος.
- `configfile` είναι ένα προαιρετικό configuration αρχείο που μπορείτε να το χρησιμοποιήσετε ώστε να παραμετροποιήσετε εσείς όπως επιθυμείτε την εφαρμογή σας.

Οι παραπάνω in-line παράμετροι (και αντίστοιχες σημαίες) είναι προαιρετικές όσον αφορά την κλήση τους στην γραμμή εντολής. Ωστόσο η υλοποίησή τους είναι υποχρεωτική. Οι σημαίες μπορούν να εμφανίζονται με *οποιαδήποτε σειρά*.

Περιγραφή της Διεπαφής του Προγράμματος:

Αφού το πρόγραμμα κληθεί, και οι σχετικές δομές για όλους τους φοιτητές εισαχθούν, η/ο χρήστης μπορεί να αλληλεπιδράσει με τα δεδομένα του `mngstd` μέσω ενός prompt. Μέσω του τελευταίου οι παρακάτω εντολές μπορούν να κληθούν:

1. `i(nsert) studentid lastname firstname zip year gpa`: εισήγαγε στην δομή ένα φοιτητή με κλειδί `studentid` που έχει εισαχθεί το έτος `year`, έχει μόνιμη κατοικία στον Τ.Κ. `zip` και έχει μέσο όρο μέχρι στιγμής `gpa`. Επανάληψη εισαγωγής εγγραφής με υπάρχον κλειδί `studentid` στην δομή δεν είναι εφικτή και σχετική ένδειξη λάθους εμφανίζεται στην έξοδο.
2. `l(ook-up) studentid`: ανάκτησε και τύπωσε στην εγγραφή του εν λόγω φοιτητή στο `tty`. Αν δεν υπάρχει, τύπωσε σχετικό μήνυμα.
3. `d(elete) studentid`: διέγραψε από την δομή την/τον φοιτήτρια/φοιτητή με αριθμό `studentid`. Αν δεν υπάρχει τέτοια εγγραφή, τύπωσε σχετικό μήνυμα.
4. `n(umber) year`: για την ακαδημαϊκή χρονιά `year` βρες πόσοι παραμένουν εγγεγραμμένοι. Στο ΕΚΠΑ εγγεγραμμένος μπορεί να παραμείνει κάποιος χωρίς περιορισμό. Η εντολή αυτή βρίσκει πόσοι φοιτητές συνεχίζουν να δραστηριοποιούνται πχ. στο 6ο χρόνο, κλπ.
5. `t(op) num year`: για την ακαδημαϊκή χρονιά `year` βρες τους `num` φοιτητές με την καλύτερη απόδοση.
6. `a(verage) year`: για την ακαδημαϊκή χρονιά `year` βρες τους τον μέσο όρο.
7. `m(inimum) year`: για την ακαδημαϊκή χρονιά `year` βρες την/τον φοιτήτρια/φοιτητή με το μικρότερο μέσο όρο.
8. `c(ount)`: για κάθε έτος φοίτησης βρες τον αριθμό των φοιτητών που υπάρχουν στην δομή.
9. `p(ostal code) rank`: βρες την πόλη (δηλ. Τ.Κ.) που είναι ή `rank`th πιο 'δημοφιλής' όσον αφορά το μέρος μόνιμης κατοικίας των φοιτητών.
10. `exit`: το πρόγραμμα τερματίζει αφού απελευθερώσει με συγκροτημένο τρόπο όλο το χώρο που έχει καταλάβει στην μνήμη.

Η ακριβής αναμενόμενη *μορφή εξόδου* της κάθε μία από τις παραπάνω εντολές δίνεται με *λεπτομέρεια* στην σελίδα του μαθήματος.

Χαρακτηριστικά του Προγράμματος που Πρέπει να Γράψετε:

1. Δεν μπορείτε να κάνετε pre-allocate οποιοδήποτε χώρο αφού δομή(-ές) θα πρέπει να μπορεί(-ουν) να μεγαλώσει(-ουν) χωρίς ουσιαστικά κανέναν περιορισμό όσον αφορά στον αριθμό των εγγραφών που μπορούν να αποθηκεύσουν. Η χρήση στατικών πινάκων/δομών που δεσμεύονται στην διάρκεια της συμβολομετάφρασης του προγράμματος σας δεν είναι αποδεκτές επιλογές.
2. Για δομές που θα υιοθετήσετε, θα πρέπει να μπορείτε να εξηγήσετε (και να δικαιολογήσετε στην αναφορά σας) την πολυπλοκότητα που παρουσιάζουν.
3. Πριν να τερματίσει η εκτέλεση της εφαρμογής, το περιεχόμενο των δομών απελευθερώνεται με σταδιακό και ελεγχόμενο τρόπο.
4. Αν πιθανώς κάποια κομμάτια του κωδικά σας προέλθουν από κάποια δημόσια πηγή, θα πρέπει *να δώσετε αναφορά στη εν λόγω πηγή* είτε αυτή είναι βιβλίο, σημειώσεις, Internet URL κλπ. και να εξηγήσετε πως ακριβώς χρησιμοποιήσατε την εν λόγω αναφορά.
5. Έχετε πλήρη ελευθερία να επιλέξετε το τρόπο με τον οποίο τελικά θα υλοποιήσετε βοηθητικές δομές.

Τι πρέπει να Παραδοθεί:

1. Μια σύντομη και περιεκτική εξήγηση για τις επιλογές που έχετε κάνει στο σχεδιασμό του προγράμματος σας (2-3 σελίδες σε ASCII κειμένου είναι αρκετές).
2. Οπωσδήποτε ένα `Makefile` (που να μπορεί να χρησιμοποιηθεί για να γίνει αυτόματα το compile του προγράμματος σας). Πιο πολλές λεπτομέρειες για το (`Makefile`) και πως αυτό δημιουργείται δίνονται στην ιστοσελίδα του μαθήματος.
3. Ένα `zip/tar-file` με όλη σας την δουλειά σε έναν κατάλογο που πιθανώς να φέρει το όνομα σας και θα περιέχει όλη σας την δουλειά δηλ. `source files, header files, output files` (αν υπάρχουν) και οτιδήποτε άλλο χρειάζεται.

Άλλες Σημαντικές Παρατηρήσεις:

1. Οι εργασίες είναι **ατομικές**.
2. Το πρόγραμμα σας θα πρέπει να τρέχει στα `Linux` συστήματα του τμήματος αλλιώς **δεν μπορεί να βαθμολογηθεί**.
3. Αν και αναμένεται να συζητήσετε με φίλους και συνεργάτες το πως θα επιχειρήσετε να δώσετε λύση στο πρόβλημα, αντιγραφή κώδικα (οποιαδήποτε μορφής) είναι κάτι που **δεν επιτρέπεται** και δεν πρέπει να γίνει. Οποιοσδήποτε βρεθεί αναμεμειγμένος σε αντιγραφή κώδικά **απλά παίρνει μηδέν** στο μάθημα. Αυτό ισχύει για **όσους εμπλέκονται ανεξάρτητα** από το ποιος έδωσε/πήρε κλπ.
4. Το παραπάνω ισχύει αν διαπιστωθεί *έστω και μερική άγνοια* του κώδικα που έχετε υποβάλει ή *άπλα υπάρχει υποψία* ότι ο κώδικας είναι προϊόν συναλλαγής με τρίτο/-α άτομο/α.
5. Προγράμματα που **δεν χρησιμοποιούν** separate compilation χάνουν αυτόματα 5% του βαθμού.
6. Σε καμιά περίπτωση τα Windows *δεν είναι επιλέξιμη* πλατφόρμα για την παρουσίαση αυτής της άσκησης.