ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
1η Εργασία - Τμήμα: Αρτίων Αριθμών Μητρώου
K22: Λειτουργικά Συστήματα – Χειμερινό Εξάμηνο '23
Ημερομηνία Ανακοίνωσης: Τρίτη 3 Οκτωβρίου 2023
Ημερομηνία Υποβολής: Τρίτη 24 Οκτωβρίου 2023 Ώρα 23:59

Εισαγωγή στην Εργασία:

Ο στόχος αυτής της εργασίας είναι να εξοικειωθείτε με το περιβάλλον Linux και τα σχετικά εργαλεία προγραμματισμού και ανάπτυξης λογισμικού. Σε C/C++, θα αναπτύξετε μια εφαρμογή με όνομα `mvote`, η οποία επιτρέπει σε μια δημοσκόπο (pollster) να παρακολουθεί την εξέλιξη μιας ψηφοφορίας. Το πρόγραμμα σας με την βοήθεια μιας σύνθεσης δομών θα επιτρέπει την εισαγωγή/εξαγωγή πληροφοριών και στατιστικών στοιχείων. Τα βασικά στοιχεία των εκλεκτόρων στην ψηφοφορία πρέπει να εισάγονται και να αναζητούνται σε μια δομή που έχει κόστος προσπέλασης *O(1)*. Κάτι τέτοιο μπορείτε να το επιτύχετε με την χρήση γραμμικού κατακερματισμού (linear hashing)[1, 2, 3]. Επίσης, θα πρέπει να μπορείτε *δυναμικά* να απαντήσετε επερωτήσεις που έχουν να κάνουν με συγκεκριμένες ομάδες συμμετεχόντων, χρησιμοποιώντας μια δομή τύπου ανεστραμμένης διπλής λίστας που επιτρέπει την κατά ταχυδρομικό κωδικό παρακολούθηση της ψηφοφορίας.

Μερικές βασικές προϋποθέσεις για την παραπάνω εφαρμογή είναι οι εξής:

1. διαχειρίζεται ένα *τυχαίο αριθμό* από ψηφοφόρους. Η κάθε ψηφοφόρος/συμμετέχουσα διαθέτει ένα μοναδικό PIN (Poll ID Number), επίθετο, όνομα και το ταχυδρομικό κώδικα (T.K.) της κατοικίας της καθώς επίσης και αν έχει ή δεν έχει ψηφίσει μέχρι στιγμής. Αρχικά θεωρούμε ότι κανένας εκλέκτορας δεν έχει ψηφίσει.
2. χρησιμοποιεί ένα πίνακα βασισμένο σε γραμμικό κατακερματισμό [1, 2, 3] που επιτρέπει στην δημοσκόπο να αποθηκεύσει, προσπελάσει, διαχειριστεί την κατάσταση ψηφοφορίας βασισμένη στο PIN του κάθε εκλέκτορα. Ο πίνακας κατακερματισμού δεν είναι στατικός και θα μπορεί να μεγαλώνει ανάλογα με τις εισαγωγές εκλεκτόρων που δέχεται η `mvote`.
3. μπορεί γρήγορα να δώσει προσπέλαση στους εκλέκτορες που έχουν ήδη ψηφίσει σε κάθε T.K.
4. επιτρέπει στη διευθύνουσα της ψηφοφορίας να εισαγάγει και να αλλάζει δεδομένα για την κατάσταση κάθε εκλέκτορα.
5. θα πρέπει –όποτε αυτό απαιτείται– να ελευθερώνει όλη την μνήμη που έχει δεσμεύσει. Φυσικά αυτό ισχύει στον τερματισμό της εφαρμογής/ψηφοφορίας.

Διαδικαστικά:

Το πρόγραμμα 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 `https://www.alexdelis.eu/k22/` για επιπρόσθετες ανακοινώσεις.

- Υπεύθυνοι για την άσκηση αυτή (ερωτήσεις, αξιολόγηση, βαθμολόγηση κλπ.) είναι ο Δρ. Σαράντης Πασκαλής `paskalis+AT-di`, ο κ. Άγγελος Πουλής `sdi1900230+AT-di`, και ο κ. Δημήτρης Ροντογιάννης `dronto+AT-di`.
- Στην διάρκεια της 1ης εβδομάδας του εξαμήνου θα πρέπει να 'εγγραφείτε' στο μάθημα στη πλατφόρμα `https://eclass.uoa.gr/courses/DI634/`

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`. Στη σελίδα που βρίσκεται στο `https://piazza.com/uoa.gr/fall2023/197eb/home` μπορείτε να κάνετε ερωτήσεις και να δείτε απαντήσεις ή/και διευκρινήσεις που δίνονται σχετικά με τις προγραμματιστικές ασκήσεις του μαθήματος.

Παραδοχές για το mvote:

Θεωρείστε για το πρόγραμμα σας τα παρακάτω:

1. Τα διαθέσιμα δεδομένα για κάθε συμμετέχοντα στην ψηφοφορία είναι: PIN, επίθετο, όνομα και Τ.Κ.
2. Οι εκλέκτορες παρέχονται στο mvote με την μορφή ενός *ASCII* αρχείου στη γραμμή εντολής. Κάθε γραμμή στο εν λόγω αρχείο, παρέχει τις πληροφορίες για κάθε εκλέκτορα. Το μήκος του αρχείου δεν είναι γνωστό εξ' αρχής.
3. Η διευθύνουσα την ψηφοφορία θα μπορεί μέσω γραμμής εντολής να μεταβάλει την πληροφορία σχετικά με το αν ένας εκλέκτορας έχει ήδη ψηφίσει.
4. Το PIN χρησιμοποιείται σαν *κλειδί* για να εισαχθούν τα δεδομένα κάθε εκλέκτορα στη δομή γραμμικού κατακερματισμού. Αντίστοιχα ο Τ.Κ. χρησιμοποιείται σαν *κλειδί* ώστε να υπάρχει γρήγορη προσπέλαση στην πληροφορία των συμμετεχουσών στην ψηφοφορία για κάθε Τ.Κ.

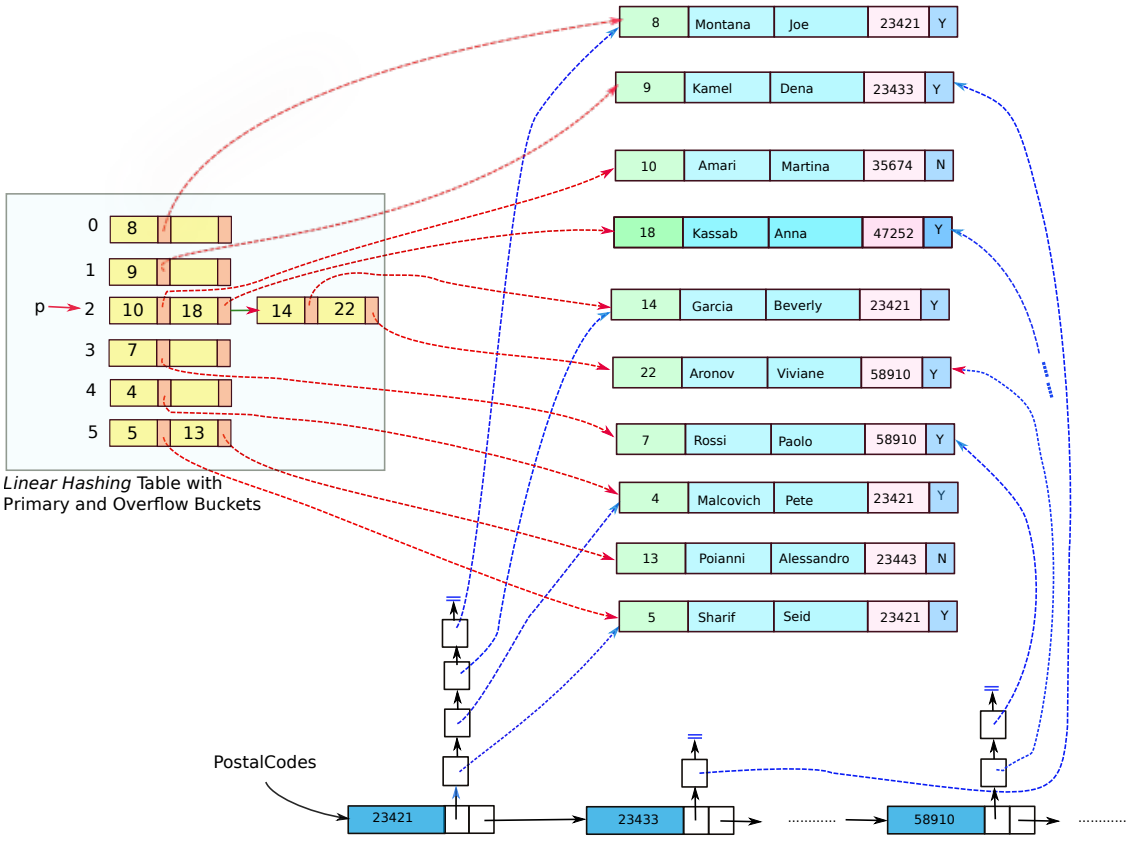
Σύνθεση Δομών για το mvote:

Το Σχήμα 1 δείχνει μια μερική αλλά αντιπροσωπευτική κατάσταση της σύνθεσης δομών που θα δημιουργήσετε για την λειτουργία του mvote.

Στο πάνω δεξιά κομμάτι του Σχήματος 1, βλέπετε τον γραμμικό κατακερματισμό που χρησιμοποιεί μια σειρά από συναρτήσεις $h_i()$ επιλογής σας [1]. Η δομή (Σχήμα 1) δημιουργείται με την χρήση buckets που το καθένα έχει το πολύ 2 κλειδιά. Ο κατακερματισμός που προκύπτει προσφέρει γρήγορη προσπέλαση στις εγγραφές εκλεκτόρων με πολυπλοκότητα $O(1)$. Η σύνθεση της δομής αλλάζει δυναμικά καθώς νέες εγγραφές εκλεκτόρων εισάγονται. Οι διακοπτόμενες κόκκινες–καμπύλες δείχνουν την σύνδεση μεταξύ PINs και συγκεκριμένων εγγραφών εκλεκτόρων. Για παράδειγμα, τα κλειδιά για τις εγγραφές των χρηστών Amari, Kassab, Garcia και Aronov κατακερματίζονται στο bucket 2 και την επέκταση του.

Στο κάτω μέρος του Σχήματος 1, η εμφανιζόμενη δομή που είναι ένας *ανεστραμμένος κατάλογος*, ομαδοποιεί τους εκλέκτορες που έχουν ήδη ψηφίσει ανά Τ.Κ. Οι διακοπτόμενες μπλε–καμπύλες είναι pointers που 'συνδέουν' κάθε Τ.Κ. με ήδη-συμμετέχοντες στην ψηφοφορία στον εν λόγω Τ.Κ. Εάν η διευθύνουσα την ψηφοφορία αποφασίσει να μεταβάλλει την κατάσταση ψήφου για ένα εκλέκτορα, οι σχετικές αλλαγές πρέπει να εμφανιστούν σε όλη την οργάνωση δομών. Για παράδειγμα στο Τ.Κ. 58910, οι Rossi & Aronov φαίνεται να έχουν ήδη ψηφίσει ('Υ' στο σχετικό πεδίο των εγγραφών). Αν ο Konnos Kollias που διαμένει στον Τ.Κ. 58910 εισαχθεί και σημανθεί ότι έχει ήδη ψηφίσει, τότε θα πρέπει να υπάρξουν αλλαγές και στην δομή οργάνωσης για τον εν λόγω Τ.Κ.

Η συγκεκριμένη μορφή που παίρνουν ο πίνακας καταμερισμού και ο ανεστραμμένος κατάλογος έχουν να κάνουν και με επιλογές σχεδιασμού που θα πρέπει να πάρετε στην ανάπτυξη του mvote. Οι επιλογές αυτές θα πρέπει να περιγραφούν στο σύντομο αρχείο σχεδιασμού που θα υποβάλετε μαζί με τον κώδικα σας.



Σχήμα 1: Οργάνωση δομών για την εφαρμογή `mvote`

Η Κλήση της Εφαρμογής `mvote`

Η εφαρμογή θα πρέπει αυστηρά να μπορεί να κληθεί με βάση την παρακάτω γραμμή εντολής (`tty`):
mymachine-promt >> `./mvote -f registeredvoters -b bucketentries`

- `./mvote` είναι μονοπάτι για το πρόγραμμα σας,
- σημαία `-f` ονοματίζει το αρχείο registeredvoters που περιέχει το αρχικό σύνολο εκλεκτόρων,
- σημαία `-b` παρέχει τον μέγιστο αριθμό από κλειδιά που μπορεί να περιέχει το κάθε bucket.

Αν το κρίνετε απαραίτητο, μπορείτε να χρησιμοποιήσετε περεταίρω σημαίες/παραμέτρους γραμμής εντολών. Η σειρά με την οποία δίνονται οι διάφορες σημαίες/παράμετροι ως είσοδο στη γραμμή εντολών μπορεί να είναι τυχαία.

Όταν η εφαρμογή αρχικοποιηθεί, παρουσιάζει στη διευθύνουσα ένα *prompt*. Κάθε φορά που η διευθύνουσα δίνει μια εντολή, το `mvote` 'αντιδρά' και φέρνει σε πέρας την σχετική ενέργεια. Στο *prompt*, το πρόγραμμα σας πρέπει να μπορεί να διαχειρίζεται τις παρακάτω εντολές:

1. `l <pin>`
   προσπέλασε το πίνακα κατακερματισμού για την εκλέκτορα με PIN: `<pin>`. Εάν η εκλέκτορας βρεθεί, οι πληροφορίες της εγγραφής παρέχονται στο tty. Διαφορετικά μια ένδειξη λάθους παρέχεται.
2. `i <pin> <lname> <fname> <zip>`
   εισήγαγε τις σχετικές πληροφορίες για ένα συγκεκριμένο εκλέκτορα του οποίου το ID είναι `<pin>`, το επίθετο και όνομα του είναι `<lname>` και `<fname>` και ο Τ.Κ. της μόνιμης διαμονής του είναι `<zip>`.

Όσοι εκλέκτορες εγγράφονται στην ψηφοφορία έχουν αρχικά την ψηφοφορίας σαν *"N"*. Εάν το `<pin>` ήδη υπάρχει στο πίνακα κατακερματισμού, η εντολή εισαγωγής ακυρώνεται και ένα σχετικό μήνυμα λάθους τυπώνεται.

3. `m <pin>`
   σημείωσε ότι η συμμετέχουσα με ID `<pin>` έχει ήδη ψηφίσει και θέσε την τιμή της κατάστασης της σε *"Y"*. Αν η κατάσταση στην εν λόγω εγγραφή είναι ήδη *"Y"*, δεν χρειάζεται να γίνει κάτι παραπάνω και ένα σχετικό μήνυμα εμφανίζεται στην έξοδο.
4. `bv <fileofkeys>`
   για οσους εκλέκτορες τα κλειδιά των οποίων εμφανίζονται στο αρχείο `<fileofkeys>` θέσε την κατάσταση ψηφοφορίας τους σε *"Y"*. Για κάθε επιχειρούμενη αλλαγή κατάστασης ψηφοφορίας, η ίδια συμπεριφορά με την εντολή '`m <pin>`' πρέπει να ακολουθηθεί.
5. `v`
   δώσε τον αριθμό εκλεκτόρων που έχουν ψηφίσει μέχρι στιγμής.
6. `perc`
   παρουσίασε το ποσοστό εκλεκτόρων που έχουν ήδη ψηφίσει σε σχέση με τον αριθμό όλων των εκλεκτόρων.
7. `z <zipcode>`
   παρουσίασε τον αριθμό των συμμετεχόντων στη ψηφοφορία στο Τ.Κ. `<zipcode>` και τύπωσε τα `PINs` τους.
8. `o`
   παρουσίασε όλους τους Τ.Κ. για τους οποίους υπάρχουν ήδη συμμετέχοντες στην ψηφοφορία. Η εν λόγω έξοδος θα πρέπει να είναι ταξινομημένη σε φθίνουσα σειρά σε σχέση με τον αριθμό των ψηφοφόρων ανά Τ.Κ. Κάθε γραμμή στην έξοδο αποτελείται από ένα Τ.Κ. και των αριθμό εκλεκτόρων που η κατάσταση ψηφοφορίας τους είναι *"Y"*.
9. `exit`: Το πρόγραμμα τερματίζει αφού αποδεσμεύσει τη μνήμη που δέσμευσε για τη λειτουργία του.
10. Οποιαδήποτε άλλη εντολή νομίζετε ότι μπορεί να σας βοηθήσει στην ανάπτυξη της εφαρμογής. Θα μπορούσατε να εισάγετε εντολές που αναπαράγουν με τρόπο κατανοητό το περιεχόμενο την δομών του `mvote`.

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 με στατικό τρόπο 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 *δυναμικές* 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. 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 κειμένο 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/7z` αρχείο 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.

Βαθμολόγηση Προγραμματιστικής Άσκησης:

Σημεία Αξιολόγησης Άσκησης	*Ποσοστό Βαθμού (0–100)*
Ποιότητα στην Οργάνωση Κώδικα & Modularity	20%
Σωστή Εκτέλεση Εντολών `mvote`	30%
Γραμμικός Κατακερματισμός	25%
2-διαστατη Συνδεδεμένη Λίστα	08%
Χρήση Makefile & Separate Compilation	08%
Αποψύλλωση Λαθών με *Valgrind*	03%
Επαρκής/Κατανοητός Σχολιασμός Κώδικα	06%

Άλλες Σημαντικές Παρατηρήσεις:

1. Η εργασία είναι **ατομική**.
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 επίσης 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 ή με συστήματα αυτόματης παραγωγής λογισμικού ή με αποθηκευτήρια (repositories) οποιασδήποτε μορφής.
5. Αναμένουμε ότι όποια υποβάλει την εν λόγω άσκηση θα πρέπει να έχει πλήρη γνώση και δυνατότητα εξήγησης του κώδικα. Αδυναμία σε αυτό το σημείο οδηγεί σε μηδενισμό στην άσκηση.
6. Προγράμματα που **δεν χρησιμοποιούν** separate compilation χάνουν αυτόματα 8% του βαθμού.
7. Σε καμιά περίπτωση τα Windows *δεν είναι επιλέξιμη* πλατφόρμα για την παρουσίαση αυτής της άσκησης.

Αναφορές

[1] Y. Chronis and A. Delis, *Linear Hashing: A Running Example*, Technical Note, `https://www.alexdelis.eu/LinearHashing-CD21.pdf`. June 2021.

[2] D. Zhang et al., *Linear Hashing*. Technical Report, `https://www.alexdelis.eu/M149/e_ds_linearhashing.pdf`. October 2012.

[3] L. Witold, *Linear Hashing: A New Tool for File and Table Addressing*, Proc. of 6th Int. Conf. on Very Large Databases, Montreal, Canada, 1980.