ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
K24: Προγραμματισμός Συστήματος – Εαρινό Εξάμηνο 2012
1η Προγραμματιστική Εργασία
Ημερομηνία Ανακοίνωσης: 6/3/2012
Ημερομηνία Υποβολής: 23/3/2012

Εισαγωγή στην Εργασία:

Ο στόχος αυτής της εργασίας είναι να εξοικειωθείτε με το προγραμματισμό σε C/Unix.

Θα υλοποιήσετε ένα μηχανισμό cache που μπορεί να βαστάει εγγραφές που θα προέρχονται από ένα αρχείο τηλεφωνικών εγγραφών. Η cache θα δουλεύει με την τεχνική *Least Recently Used (LRU)* και ο αριθμός των εγγραφών στην cache θα είναι περιορισμένος σε σχέση με εκείνες τις εγγραφές που βρίσκονται στο αρχείο.

Η cache θα πρέπει να έχει και τα εξής χαρακτηριστικά:

1. Αν προσπαθούμε να βρούμε μία εγγραφή που δεν υπάρχει στην cache θα πρέπει να την ανακτήσουμε από το αρχείο δεδομένων.
2. Αν προσπαθούμε να βρούμε μία εγγραφή που βρίσκεται ήδη στην cache το κόστος της ανάγνωσης θα πρέπει να είναι $O(1)$.
3. Το κόστος διαγραφής μια εγγραφής από την cache πρέπει να είναι $O(1)$. Η ανάγκη αυτή προκύπτει όταν ουσιαστικά φεύγει μια εγγραφή για να δημιουργήσει χώρο, για μια άλλη που εισέρχεται.
4. Οι εγγραφές που βρίσκονται στην cache (και είναι πάντα υποσύνολο εκείνων στο αρχείο δεδομένων) ακολουθούν την LRU. Η διατήρηση των ιδιοτήτων της LRU για μια έγγραφή που υπάρχει στο cache θα πρέπει να έχει κόστος $O(k)$ όπου k είναι ο αριθμός των εγγραφών που βρίσκονται στην cache.

Διαδικαστικά:

Το πρόγραμμά σας θα πρέπει να τρέχει στα μηχανήματα Linux/Unix της σχολής. Παρακολουθείτε τη λίστα του μαθήματος για επιπρόσθετες ανακοινώσεις στο URL: *www.di.uoa.gr/∼ad* και τη λίστα του μαθήματος.

- Υπεύθυνοι για την άσκηση αυτή είναι (ερωτήσεις, αξιολόγηση, βαθμολόγηση, κτλ) είναι οι: Χρήστος Πατσονάκης (grad1135), Παύλος Βινιεράτος (grad1123) και Νίκος Κατσιπουλάκης (katsip).
- Εγγραφείτε στην ηλεκτρονική λίστα (mailman) και παρακολουθείτε ερωτήσεις/απαντήσεις/διευκρινήσεις που δίνονται σχετικά με την άσκηση (σημείωση: η η-λίστα αυτή δεν έχει καμιά σχέση με την hard-copy λίστα που κυκλοφορεί/ησε στα πρώτα μαθήματα και στην ποία θα έπρεπε να γράψετε το όνομά σας και το Unix user-id σας).

Τι πρέπει να Παραδοθεί:

1. Μια σύντομη και περιεκτική εξήγηση για τις επιλογές που έχετε κάνει στο σχεδιασμό του προγράμματος σας (2-3 σελίδες κειμένου είναι αρκετές).
2. *Οποιαδήποτε πηγή πληροφορίας* συμπεριλαμβανομένου και κώδικα που μπορεί να βρείτε στο Διαδίκτυο θα πρέπει να αναφερθεί και στο πηγαίο κώδικα σας αλλά και στην παραπάνω αναφορά σας.
3. Ένα `tar file` με όλη σας τη δουλειά σε έναν κατάλογο που πιθανώς να φέρει το όνομά σας και θα περιέχει όλη σας τη δουλειά.

Διατύπωση του Προβλήματος:

Θεωρούμε ότι υπάρχει ένα ASCII αρχείο που έχει την μορφή του που φαίνεται στο παρακάτω Πίνακα. Το αρχείο περιέχει μια τηλεφωνική εγγραφή σε κάθε γραμμή. Η κάθε εγγραφή αποτελείται από το τηλεφωνικό νούμερο ενός χρήστη (μοναδικό κλειδί), το όνομα τού χρήστη και την διεύθυνση του/της. Θα πρέπει να σημειωθεί ότι οι εγγραφες δεν έχουν προκαθορισμένο μήκος.

Phone Number	Name	Address
7275212;	Alex Delis;	DI, Kessariani, Athens, Attica;
7275213;	Gus Tsavelas;	Megalopolis, Arcadia;
7275215;	John Kapnogianis;	Theodorou 23, Arkitsa, Ftiotida;
3456634;	Nikos Papas;	Kalamata, Messinia;
4563231;	Ted Kolokotronis;	Argos, Argolida;
4568712;	Andreas Karas;	Panepistimioy 40, Athens, Attica;
.....	.....	.....

Η κλήση του προγράμματος σας πρέπει να έχει την μορφή:

./mycache -n NumOfRecords -f FileWithPhones

Η παραπάνω κλήση ουσιαστικά λέει ότι κατά την διάρκεια της εκτέλεσης του προγράμματος θα πρέπει να χρησιμοποιηθούν εγγραφές από το αρχείο FileWithPhones και η cache μπορεί να αποθηκεύσει μέγιστο αριθμό από NumOfRecords εγγραφές. Η παραπάνω κλήση αυτόματα θα πρέπει να βάζει το χρήστη σε ένα prompt προγράμματος στο οποίο η χρήστης να μπορεί να κάνει απλές λειτουργίες. Το ρεπερτόριο αυτών των εντολών είναι:

1. read phonenum: βρες από το την cache (ή το αρχείο) την εγγραφή που έχει το συγκεκριμένο τηλεφωνικό νούμερο (σύντμηση εντολής r phonenum).
2. load NewFile: η χρήση του αρχείου που έχει δοθεί στην εντολή εκτέλεσης παύει. Το αρχείο με όνομα NewFile χρησιμοποιείται Επίσης η cache αδειάζει από όλα τα δεδομένα που είχαν προέλθει από το αρχείο fileWithPhones (σύντμηση εντολής l NewFile).
3. exit: η λειτουργία του προγράμματος τερματίζεται και όλες οι δομές ελευθερώνονται πριν το πρόγραμμα τερματιστεί (σύντμηση εντολής e).

Σε κάθε μία από τις παραπάνω λειτουργίες θα πρέπει να τυπώνεται και ο χρόνος που έχει διαρκέσει η εν-λόγω λειτουργία με τις κλήσεις χρονισμού που δίνονται στο τέλος της άσκησης.

Αλγοριθμικά Θέματα και Περιορισμοί:

Το πρόγραμμά σας για να παράγει τα αποτελέσματα που θέλουμε θα πρέπει να δουλεύει ως εξής:

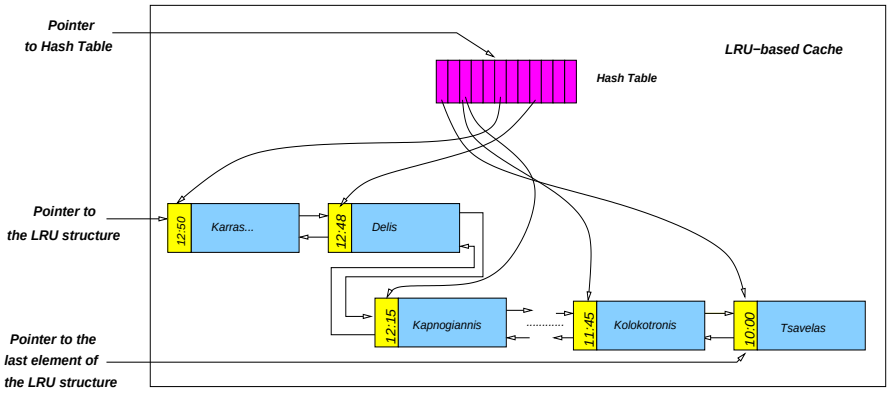
1. Όλα τα χαρακτηριστικά της δομής cache που αναφέρθηκαν προηγουμένως θα πρέπει να ισχύουν σε όλη την διάρκεια της εκτέλεσης του προγράμματός σας.

2. Εγγραφές που δεν είναι παρούσες στην cache θα πρέπει να ανασυρθούν από το αρχείο εγγραφών και να μπουν στην εν λόγω δομή. Αν δεν υπάρχει χώρος θα πρέπει να επιλεγεί μια εγγραφή θύμα (κατά τα γνωστά) και να αποχωρήσει από την δομή ώστε να δημιουργηθεί χώρος για νεοεισερχόμενη εγγραφή.

3. Γενικά δεν υπάρχει κανένας περιορισμός στο μέγεθος όσον αφορά στο μέγεθος του αρχείου εγγραφών.

4. Η ανεύρεση εγγραφών από το αρχείο μπορεί να γίνεται με απλό σειριακό τρόπο (δηλ. διαβάζουμε όλες τις εγγραφές από την αρχή μια-προς-μία μέχρι να βρούμε αυτή που χρειαζόμαστε).

5. Το Σχήμα 1 δίνει ένα πιθανό σχεδιασμό για την εσωτερική δομή του cache. Εγγραφές ανακτώνται με την βοήθεια του hash-table ώστε το κόστος να είναι $O(1)$. Οι εγγραφές μπορούν να διαταχθούν σε μια λίστα με βάση την τελευταία φορά που χρησιμοποιήθηκαν και η λίστα αυτή μπορεί να έχει μέχρι μέγιστο αριθμό από NumOfRecords στοιχεία. Τέλος, πάντα υπάρχει ένας δείκτης στο τελευταίο στοιχείο της λίστας ώστε οποιαδήποτε χρονική στιγμή να γνωρίζουμε την εγγραφή θύμα (στα πλαίσια του LRU).



Σχήμα 1: Πιθανός σχεδιασμός του cache.

Άλλες Παραδοχές:

1. Θεωρήστε ότι κάθε γραμμή δεδομένων δεν μπορεί να είναι πιο μεγάλη από 128 χαρακτήρες και ότι όλες οι εγγραφές αποτελούνται από ένα ακέραιο (το τηλέφωνο) και ένα alphanumeric string(υπόλοιπες πληροφορίες).
2. Κάθε γραμμή του αρχείου μπορεί να έχει μεταβλητό μέγεθος αλλά τελειώνει με ένα ειδικό χαρακτήρα (delimeter ‘‘\n’’). Έτσι ξέρουμε που τελειώνει η εν λόγω γραμμή.

Τι θα Βαθμολογηθεί:

1. Η συμμόρφωση του κωδικά σας με τις προδιαγραφές του mycache.
2. Η **οργάνωση** και η **αναγνωσιμότητα** (μαζί με την ύπαρξη σχολίων) του κώδικα που θα πρέπει να βρίσκεται σε πολλαπλά αρχεία.
3. Η χρήση Makefile και κομματιαστής συμβολομετάφρασης (separate compilation).
4. Η αναφορά (2-3 το πολύ σελίδες) που θα γράψετε και θα υποβάλετε μαζί με τον πηγαίο κώδικα σε μορφή PDF.

Άλλες Σημαντικές Παρατηρήσεις:

1. Οι εργασίες είναι **ατομικές.**
2. Όποιος υποβάλλει/ δείχνει κώδικα που δεν έχει γραφτεί από την ίδια/ίδιο **μηδενίζεται** στο μάθημα.
3. Αν και αναμένεται να συζητήσετε με φίλους και συνεργάτες το πώς 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 κώδικα απλά παίρνει μηδέν στο μάθημα. Αυτό ισχύει για **όλους όσους εμπλέκονται** ανεξάρτητα από το ποιος έδωσε/πήρε κλπ.
4. Το πρόγραμμα σας θα πρέπει να τρέχει σε Ubuntu-Linux ή Solaris αλλιώς **δεν θα βαθμολογηθεί**.
5. Σε καμιά περίπτωση τα MS-Windows **δεν είναι επιλογή** πλατφόρμας για την παρουσίαση αυτής της άσκησης.

Παράδειγμα Χρονομέτρησης στο Unix:

```
#include <stdio.h>      /* printf() */
#include <sys/times.h>  /* times() */
#include <unistd.h>     /* sysconf() */

int main( void ) {
    double t1, t2, cpu_time;
    struct tms tb1, tb2;
    double ticspersec;
    int    i, sum = 0;

    ticspersec = (double) sysconf(_SC_CLK_TCK);

    t1 = (double) times(&tb1);

    for (i = 0; i < 100000000; i++)
              sum += i;

    t2 = (double) times(&tb2);
    cpu_time = (double) ((tb2.tms_utime + tb2.tms_stime) -
                         (tb1.tms_utime + tb1.tms_stime));

    printf("Run time was %lf sec (REAL time) although
            we used the CPU for %lf sec (CPU time).\n",
            (t2 - t1) / ticspersec, cpu_time / ticspersec);
}
```