ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
4η Εργασία - Τμήμα: Περιττών Αριθμών Μητρώου
K22: Λειτουργικά Συστήματα – Χειμερινό Εξάμηνο '20
Ημερομηνία Ανακοίνωσης: Δευτέρα 21/12
Ημερομηνία Υποβολής: Κυριακή 24/01 και Ώρα 23:59

Εισαγωγή στην Εργασία:

Ο στόχος αυτής της εργασίας είναι να δημιουργήσετε ένα πρόγραμμα που να 'αντιγράφει γρήγορα' ιεραρχίες καταλόγων/αρχείων όταν κάτι τέτοιο γίνεται κατ' επανάληψη. Το πρόγραμμα αυτό λέγεται *quick incremental copy* ή απλά quic. Για να επιτύχουμε κάτι τέτοιο θα πρέπει να εκμεταλλευτούμε το γεγονός ότι συνήθως όταν αντιγράφουμε μια ιεραρχία (κατάλογο), ένα μεγάλο μέρος της εν λόγω αντιγραφής μπορεί να προϋπάρχει και να μην έχει δεχτεί αλλαγές. Για παράδειγμα αν υποθέσουμε ότι εργαζόμαστε μέσα σε μια ιεραρχία που έχει να κάνει με την ανάπτυξη του κώδικα της άσκησης αυτής. Στην διάρκεια της ανάπτυξης προσθ-αφαιρούμε αρχεία και κατάλογους. Αν σε κάποια στιγμή θέλουμε να χρησιμοποιήσουμε το quic για να φτιάξουμε ένα αντίγραφο σε ένα USB-stick ή οποιδήποτε άλλο σημείο του συστήματος αρχείου, κάνουμε τα εξής:

```
ad@rhodes: du -sh ~/MyProject4/
1.4M /home/ad/MyProject4/
$> quic  MyProject4 /media/usbdisk/Project4-backup/
created directory /media/usbdisk/Project4-backup/
./
src/
src/myheaders.h
src/main.c
src/quic-routines.c
src/lists/
src/lists/mylists.c
src/lists/auxiliary.c
src/lists/documentation
bin/
bin/a.out
todo/
todo/things2do1.txt
todo/things2do2.txt
todo/tmp/
... < listing of 65 files/directories >

there are 65 files/directories in the hierarchy
number of entities copied is 65
copied 1,333,598 bytes in 0.866sec at  153,995.15 bytes/sec
```

Την πρώτη φορά που εκτελείται το quic συμπεριφέρεται ακριβώς όπως και το cp -r του Linux καθώς αντιγράφει ολόκληρη την ιεραρχία μεγέθους 1.4MB από το κατάλογο MyProject4 στο κατάλογο με όνομα Project4-backup που βρίσκεται στο κατάλογο /media/usbdisk/ του USB-stick.

Μετά από λίγο και αφότου η χρήστης έχει αποφασίσει να αλλάξει το περιεχόμενο 2 αρχείων και έχει προσθέσει ακόμα ένα κατάλογο 'ενημερώνουμε' το αντίγραφο με την χρήση του qicp. Το cp -r θα αντέγραφε ολόκληρη την δομή από την αρχή. Ωστόσο το quic θα πρέπει να αντιγράψει μόνο τα στοιχεία (αρχεία/κατάλογοι) που έχουν αλλάξει. Πιο συγκεκριμένα:

```
ad@rhodes: du -sh ~/MyProject4/
1.4M /home/ad/MyProject4/
ad@rhodes: quic  MyProject4 /media/usbdisk/Project4-backup/
created directory /media/usbdisk/Project4-backup/
src/stack.c
todo/things2do2.txt
todo/tmp/junk/

there are 68 files/directories in the hierarchy
number of entities copied is 3
copied 1,598 bytes in 0.076sec at  21,026.31 bytes/sec
```

Όπως μπορείτε να διαπιστώσετε, τη δεύτερη φορά που γίνεται η κλήση του quic μόνο 3 οντότητες (δύο αρχεία και ένας κατάλογος) που έχουν αλλάξει στην νέα ιεραρχία αντιγράφονται. Στην συγκεκριμένη περίπτωση απαιτείται αντιγραφή μόνο 1,598 bytes και γενικά η απόδοση του quic πρέπει να είναι σαφώς πιο γρήγορη. Μπορείτε εύκολα να διαπιστώσετε το εν λόγω γεγονός με την χρήση του προγράμματος συστήματος *time* κάθε φορά που εκτελείτε το quic.

Θα πρέπει να *επιδείξετε την ορθότητά* της λειτουργίας του quic.

Διαδικαστικά:

Το πρόγραμμά 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.

- Υπεύθυνοι για την άσκηση αυτή (ερωτήσεις, αξιολόγηση, βαθμολόγηση κλπ.) είναι η κ. Ιωάννα Καραγεώργου `sdi1600057+AT-di`, και ο κ. Γιάννης Καλύβας `cs2190016+AT-di`.
- Παρακολουθείτε την ιστοσελίδα του μαθήματος `http://www.alexdelis.eu/k22/` για επιπρόσθετες ανακοινώσεις αλλά και την ηλεκτρονική-λίστα (η-λίστα) του μαθήματος στο URL `https://piazza.com/uoa.gr/fall2020/k22/home`
- Προφανώς η εργασία σας θα αποτελείται από πολλαπλά προγράμματα τα οποία θα πρέπει να συμβολομεταφραστούν με χρήση separate compilation.

Διαφοροποιήσεις στους Καταλόγους:

Το quic θα πρέπει να δέχεται δύο ορίσματα στη γραμμή εντολών: τον κατάλογο που θα πρέπει να αντιγραφεί (δηλ. κατάλογος προέλευσης) και τον κατάλογο προορισμού. Αν τα ορίσματα δεν δοθούν ορθά, το πρόγραμμα σας θα πρέπει να παράγει κάποιο μήνυμα λάθους. Αν ο κατάλογος προορισμού δεν υπάρχει, θα πρέπει να τον δημιουργήσετε.

Ας θεωρήσουμε ότι ένας αρχικός κατάλογος είναι ο A και ο κατάλογος τελικού προορισμού είναι ο T. Το πρόγραμμα θα πρέπει να διατρέχει τον A, να διαβάζει πληροφορίες που υπάρχουν στα εκεί `i-nodes` και για κάθε a `i-node` στον A θα πρέπει να αναζητείτε το αντίστοιχο τ `i-node` στο T (που βρίσκεται στο ίδιο σχετικό μονοπάτι). Διακρίνουμε 4 περιπτώσεις:

1. *Το τ δεν υπάρχει στο T*: σε αυτή την περίπτωση το τ θα πρέπει να δημιουργηθεί στην σωστή θέση του T και να αντιγραφεί το a στο τ. Αν το `i-node` αναφέρεται σε αρχείο, θα πρέπει να αντιγραφούν και τα δεδομένα του αρχείου.

2. *Το τ υπάρχει στο T αλλά δεν είναι ΙΔΙΟ με το a στο κατάλογο A*: εδώ θα πρέπει να ορίσουμε τι είναι το «ΙΔΙΟ» πράγμα που κάνουμε παρακάτω. Στην περίπτωση αυτή, πρέπει το περιεχόμενο του a `i-node` να αντιγραφεί στο τ. Αν τα `i-nodes` a και τ έχουν να κάνουν με αρχεία πρέπει στην αντιγραφή να περιληφθούν και τα δεδομένα αρχείων.

3. *Το τ υπάρχει στο T και είναι το ίδιο με το a στο A:* σε αυτή την περίπτωση δεν χρειάζεται να γίνει κάτι. Είναι η περίπτωση που το quic κερδίζει σε σχέση με το παραδοσιακό cp.

4. *Εάν υπάρχει ένα τ στο T που δεν έχει αντίστοιχο στο a στο κατάλογο A:* αυτή είναι η περίπτωση που κάποιο στοιχείο έχει διαγραφεί. Συνεπώς ο κατάλογος T θα πρέπει να καθαριστεί από το τ (διαγραφή).

Όταν τα `i-nodes`, a και τ αντιστοιχούν σε καταλόγους τα παραπάνω βήματα θα πρέπει να εφαρμοστούν *αναδρομικά* ώστε και τα υποκείμενα στοιχεία των καταλόγων να ελεγχθούν ενδελεχώς.

Ορισμός ΙΔΙΩΝ Αρχείων:

Για να διαπιστώσουμε ότι δύο αρχεία είναι «ίδια» θα πρέπει να ελέγξουμε την πληροφορία στα `i-nodes`. Δεν θα πρέπει να διαβάζετε το περιεχόμενο των εν λόγω αρχείων γιατί τότε η απόδοση του quic θα είναι χειρότερη του cp. Ο παραπάνω έλεγχος μπορεί να γίνει ως εξής:

- αν από τα a και τ, το ένα αντιστοιχεί σε αρχείο και το άλλο σε κατάλογο, τότε προφανώς έχουμε διαφορά.
- αν τα a και τ αναφέρονται σε καταλόγους, τότε θεωρούνται ίδια χωρίς όμως αυτό να σημαίνει και τα περιεχόμενα των καταλόγων είναι ίδια. Για να αποφασίσουμε κάτι τέτοιο, τα επιμέρους περιεχόμενα θα πρέπει να ελεγχθούν αναδρομικά.
- αν τα a και τ αναφέρονται σε αρχεία που έχουν διαφορετικό μέγεθος, είναι προφανές ότι τα εν λόγω αρχεία είναι ανόμοια.
- αν τα a και τ αναφέρονται σε αρχεία που έχουν το ίδιο μέγεθος αλλά το τ έχει ημερομηνία τροποποίησης πριν την ημερομηνία του a τότε τα a και τ δεν είναι ίδια.
- σε οποιαδήποτε άλλη περίπτωση, τα αρχεία με i-nodes a και τ θεωρούνται «ίδια».

Ο χειρισμός soft/hard links θα βαθμολογηθεί με επιπλέον μονάδες. Γενικά οι σύνδεσμοι θα πρέπει να διατηρούνται στο αντίγραφο. Πιο συγκεκριμένα, οι συμβολικοί σύνδεσμοι (soft links) θα πρέπει στο αντίγραφο να αναφέρονται στο ίδιο μονοπάτι. Το μονοπάτι αυτό μπορεί να είναι είτε απόλυτο είτε σχετικό. Στην περίπτωση των ισχυρών δεσμών (hard links) πρέπει τα δεδομένα των αρχείων να μην αντιγράφονται πάνω από μία φορά. Για παράδειγμα αν τα `i-nodes` *a1* και *a2* αναφέρονται στο ίδιο αρχείο, το αντίγραφο θα πρέπει να έχει κόμβους *τ1* και *τ2* οι οποίοι όμως θα αναφέρονται στα ίδια μπλοκς αρχείου.

Γραμμή Κλήσης του Προγράμματος:

Το πρόγραμμα σας θα μπορούσε να κληθεί ως εξής:
./quic -v -d -l origindir destdir όπου

- quick είναι το εκτελέσιμο,
- origindir ο αρχικός κατάλογος και destdir ο κατάλογος προορισμού.
- η σημαία -v (verbose) παρέχει την εκτύπωση διαγνωστικών για τις επιλογές/ενέργειες του προγράμματος όσον αφορά στην αντιγραφή/διαγραφή αρχείων.
- η σημαία -d (deleted) προσδιορίζει οτι στοιχεία του συστήματος αρχείου που έχουν διαγραφεί απο τον αρχικό κατάλογο origindir δεν θα πρέπει να υπάρχουν στο κατάλογο προορισμού destdir.
- η σημαία -l (links) καθορίζει εάν το πρόγραμμα σας λαμβάνει υπ' όψη και την διαχείριση των πιθανών συνδέσμων.

Έχετε την δυνατότητα να προσθέσετε οποιαδήποτε άλλη σημαία κρίνετε απαραίτητη.

Στο τέλος το πρόγραμμα σας θα πρέπει να παρέχει κάποια βασικά χαρακτηριστικά όσον αφορά την συμπεριφορά του όπως:

1. αριθμό οντοτήτων (αρχεία, καταλόγους και πιθανώς συνδέσμους) που το πρόγραμμα έχει «δει» στην συγκεκριμένη εκτέλεσή του,

2. αριθμό από τις παραπάνω οντότητες που τελικά αντιγράφησαν,
3. των αριθμό των bytes που αντιγράφησαν συνολικά, χρόνος που απαιτήθηκε για την εργασία του προγράμματος και το ρυθμό με τον οποίο γράφτηκαν τα παραπάνω bytes στο κατάλογου προορισμού.

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 `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. Τα πρόγραμματα σας θα πρέπει να τρέχουν στα `Linux` συστήματα του τμήματος αλλιώς **δεν μπορούν να βαθμολογηθούν**.
2. Αν και αναμένεται να συζητήσετε με φίλους και συνεργάτες το πως θα επιχειρήσετε να δώσετε λύση στο πρόβλημα, αντιγραφή κώδικα (οποιαδήποτε μορφής) μεταξύ ομάδων είναι κάτι που **δεν επιτρέπεται** και δεν πρέπει να γίνει. Οποιοσδήποτε ομάδα βρεθεί αναμεμειγμένη σε αντιγραφή κώδικά **απλά παίρνει μηδέν** στο μάθημα. Αυτό ισχύει για **όσους εμπλέκονται ανεξάρτητα** από το ποια ομάδα έδωσε/πήρε κλπ.
3. 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.
4. Προγράμματα που **δεν χρησιμοποιούν** separate compilation χάνουν αυτόματα 5% του βαθμού.
5. Σε καμιά περίπτωση τα Windows *δεν είναι επιτρεπτή* πλατφόρμα για την υλοποίηση ή παρουσίαση αυτής της άσκησης.