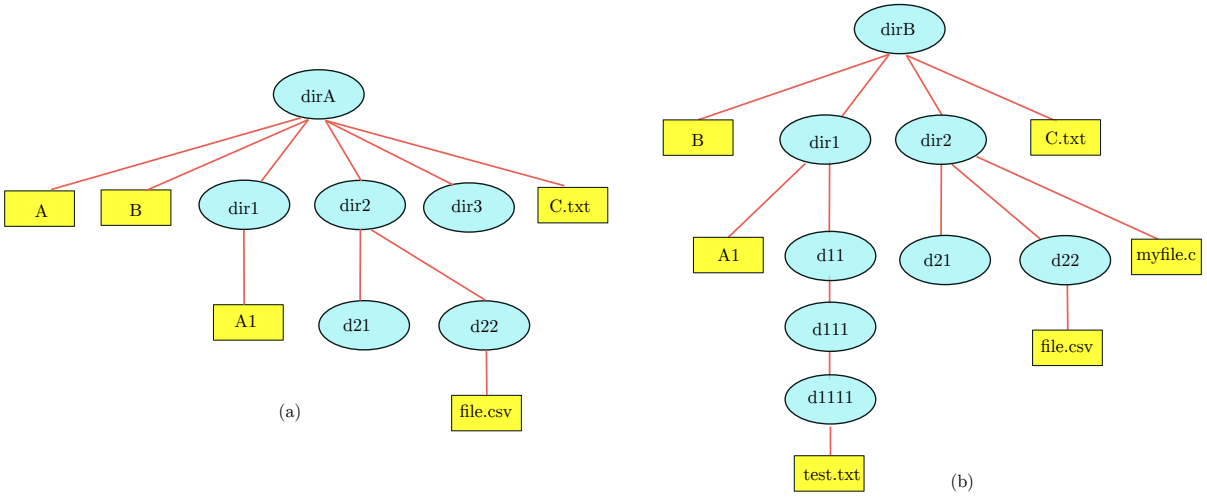
ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
4η Εργασία - Τμήμα: Αρτίων Αριθμών Μητρώου
K22: Λειτουργικά Συστήματα – Χειμερινό Εξάμηνο '23
Ημερομηνία Ανακοίνωσης: Δευτέρα 08 Ιανουαρίου 2024
Ημερομηνία Υποβολής: Δευτέρα 12 Φεβρουαρίου 2024 Ώρα 23:55

Εισαγωγή στην Εργασία: Στα πλαίσια αυτής της άσκησης πρέπει να γράψετε ένα πρόγραμμα γραμμής εντολών που συγκρίνει δύο ιεραρχίες καταλόγων μεταξύ τους. Το πρόγραμμα που ονομάζεται `cmpcats` θα μπορεί επίσης να δημιουργεί μια νέα ιεραρχία καταλόγων συγχωνεύοντας τις δύο υπό σύγκριση ιεραχίες σε ένα νέο κατάλογο.

Αν το `cmpcats` συγκρίνει δύο ίδιες ιεραρχίες καταλόγων δεν εκτυπώνεται τίποτα. Διαφορετικά, για κάθε διαφορά που παρατηρείται, θα πρέπει να εκτυπώνεται το αντίστοιχο μήνυμα, είτε πρόκειται για διαφορετικούς φακέλους, είτε για διαφορετικά αρχεία και συνδέσμους.

Το πρόγραμμα πρέπει να μπορεί (1) να συγκρίνει αναδρομικά δύο καταλόγους και να εμφανίζει τις διαφορές τους με ένα ευκολοδιάβαστο τρόπο, καθώς και (2) να δημιουργεί ένα νέο κατάλογο ο οποίος προκύπτει από τη συγχώνευση των δύο ιεραρχιών. Στη πρώτη περίπτωση η κλήση του προγράμματος γίνεται με ορίσματα τα ονόματα των δύο φακέλων, ενώ στη δεύτερη περίπτωση καλείται με ένα επιπλέον όρισμα, το όνομα του φακέλου που θέλουμε να δημιουργηθεί μετά τη συγχώνευση.



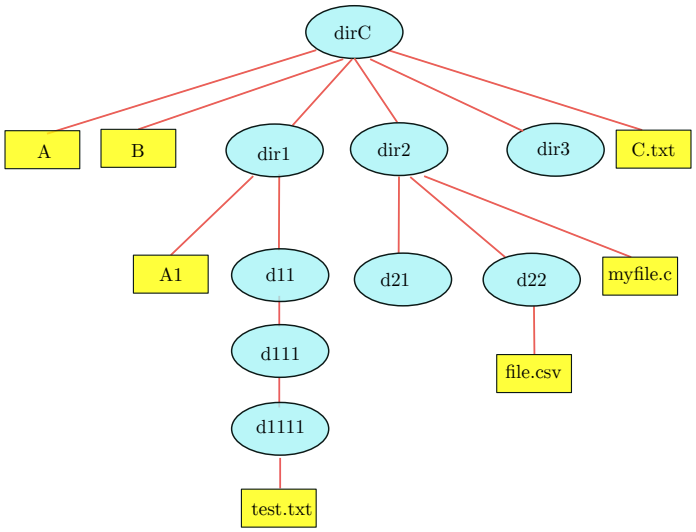
Σχήμα 1: Δύο κατάλογοι υπό σύγκριση `dirA & dirB`

Για παράδειγμα: για τις ιεραρχίες καταλόγων του Σχήματος 1, μία ενδεικτική εκτέλεση του προγράμματος για *σύγκριση των ιεραρχιών* είναι η εξής:

```
myprompt >>  cmpcats -d dirA dirB
In dirA :
        A
        dir3
In dirB:
        dir1/dir11
        dir1/dir11/dir111
        dir1/dir11/dir111/dir1111
        dir1/dir11/dir111/dir1111/test.txt
        dir2/myfile.c
```

Η παραπάνω έξοδος λέει ότι στο `dirA` υπάρχουν τα στοιχεία `A` και `dir3` που δεν υπάρχουν στο `dirB`. Παρομοίως στο `dirB` υπάρχουν τα στοιχεία `dir1/dir11`, `dir1/dir11/dir111`, `dir1/dir11/dir111/dir1111`, `dir1/dir11/dir111/dir1111/test.txt` και `dir2/myfile.c` που δεν υπάρχουν στο `dirA`. Η σειρά εμφάνισης των διαφορών δεν παίζει κάποιο ουσιαστικό ρόλο. Είστε ελεύθεροι να επιλέξετε ότι διαγνωστικά μηνύματα κρίνετε απαραίτητο ότι θα πρέπει να παρουσιαστούν καθώς επίσης και την σειρά τους (ωστόσο όλα να εμφανίζονται με κάποιο ευανάγνωστο και κατανοητό τρόπο).

Το αποτέλεσμα της εκτέλεσης:
`myprompt >> cmpcats -d dirA dirB -s dirC`
εμφανίζεται στο Σχήμα 2.



Σχήμα 2: Αποτέλεσμα συγχώνευσης των δύο καταλόγων `dirA & dirB` του Σχήματος 1 στο `dirC`

Διαδικαστικά:

- Το πρόγραμμα σας θα πρέπει να γραφτεί σε `C` (ή `C++` αν θέλετε αλλά χωρίς τη χρήση `STL/Templates`) και να τρέχει στα 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 είναι ο Δρ. Σαράντης Πασκαλής `paskalis+AT-di`, και ο κ. Σπύρος Τρυφωνίδης `+sdi1600175+AT-di`.

Διατύπωση του Προβλήματος:

Το πρόγραμμα σας θα πρέπει να δέχεται στη γραμμή εντολών τις αντίστοιχες παραμέτρους, ανάλογα αν η χρήση είναι για τον έλεγχο των διαφορών, ή και για την συγχώνευση αρχείων. Αν δε δοθούν σωστά τα ορίσματα αυτά θα πρέπει να τυπώνεται κάποιο μήνυμα λάθους και πληροφορίες για τη σωστή χρήση του προγράμματος. Στη περίπτωση της συγχώνευσης, αν ο κατάλογος προορισμού δεν υπάρχει, τότε το πρόγραμμα σας θα πρέπει να τον δημιουργεί.

Στην συνέχεια της περιγραφής θεωρείστε ότι οι δύο ιεραρχίες καταλόγων είναι οι `A` και `B`. Το πρόγραμμά σας θα πρέπει να διατρέχει τις δύο ιεραρχίες και να συγκρίνει καταλόγους και αρχεία, με το τρόπο που περιγράφεται παρακάτω, ώστε να βρίσκει διαφορές. Στην περίπτωση της σύγκρισης το πρόγραμμά σας απλά θα πρέπει να

εκτυπώνει τις διαφορές αυτές, ενώ στην περίπτωση της συγχώνευσης θα πρέπει να χειρίζεται κατάλληλα την πληροφορία αυτή ώστε να δημιουργεί τον νέο κατάλογο *χωρίς πλεονασμούς*.

Οι πληροφορίες που θα χρειαστούν για την σύγκριση των αρχείων και των καταλόγων θα αντλούνται από τα `i-nodes` με την κλήση συστήματος `stat`. Κάθε φορά πρέπει να εξετάζονται `i-nodes` που βρίσκονται στο *ίδιο σχετικό μονοπάτι*.

Ας θεωρήσουμε το α στοιχείο του συστήματος αρχείου βρίσκεται 'κάτω' από το κατάλογο `A` και αντίστοιχα το β να είναι 'κάτω' από το κατάλογο `B` με τους 2 καταλόγους `A` και `B` να βρίσκονται στο ίδιο σχετικό μονοπάτι. Για το συγκεκριμένο αυτό επίπεδο θεωρούμε ότι τα α και β είναι διαφορετικά, αν ισχύει μία από τις παρακάτω συνθήκες:

1. Αν το β δεν υπάρχει στο κατάλογο `A`.
2. Αν το α δεν υπάρχει στο κατάλογο `B`.
3. Αν α, β βρίσκονται στους καταλόγους `A` & `B` αλλά ανήκουν σε διαφορετικές κατηγορίες (κανονικά αρχεία, σύνδεσμοι, κατάλογοι).
4. Αν υπάρχουν α, β στους καταλόγους `A` & `B`, είναι στοιχεία του ιδίου τύπου (δηλ. ή αρχεία ή σύνδεσμοι ή κατάλογοι) αλλά είναι διαφορετικά σύμφωνα με τις συνθήκες που περιγράφονται παρακάτω για καταλόγους, αρχεία, και συνδέσμους.

Οι πληροφορίες που χρειάζονται για τις συγκρίσεις υπάρχουν στα *i-nodes*.

⇒ Δύο κατάλογοι θεωρούνται ίδιοι αν και μόνο αν:

1. Έχουν το ίδιο σχετικό μονοπάτι από την αρχή της ιεραρχίας.
2. Έχουν το ίδιο όνομα.

⇒ Δύο αρχεία είναι ίδια αν και μόνο αν:

1. Έχουν το ίδιο σχετικό μονοπάτι από την αρχή της ιεραρχίας.
2. Έχουν το ίδιο όνομα.
3. Έχουν το ίδιο μέγεθος.
4. Έχουν το ίδιο περιεχόμενο.

⇒ Δυο σύνδεσμοι (links) θεωρούνται ίδια αν και μόνο αν:

1. Έχουν το ίδιο σχετικό μονοπάτι από την αρχή της ιεραρχίας.
2. Έχουν το ίδιο όνομα.
3. Δείχνουν στα ίδια αρχεία.

Δεν χρειάζεται να λάβετε υπόψιν links που δείχνουν εκτός των ιεραρχιών για λόγους απλότητας.

⇒ Σε κάθε άλλη περίπτωση όλα τα παραπάνω (κατάλογοι, αρχεία, σύνδεσμοι) θεωρούνται διαφορετικά.

Στην περίπτωση που τα α και β αντιστοιχούν σε καταλόγους, πρέπει τα παραπάνω βήματα να εφαρμοστούν αναδρομικά και στα περιεχόμενα των ενφωλιασμένων καταλόγων.

Κατά τη διαδικασία της συγχώνευσης πρέπει να δημιουργήσετε μια νέα ιεραρχία αντιγράφοντας τα αρχεία που είναι κοινά και στις δύο ιεραρχίες και στην συνέχεια αντιγράφοντας τις αλλαγές που βρέθηκαν με τη προηγούμενη διαδικασία. Επίσης, θα πρέπει να χειριστείτε τις περιπτώσεις που δύο αρχεία ή σύνδεσμοι με το ίδιο όνομα **δεν είναι ίδια**. Στην περίπτωση αυτή θα πρέπει να κρατάτε μόνο αυτό που τροποποιήθηκε τελευταίο, δηλαδή έχει μεταγενέστερη ημερομηνία τροποποίησης.

Επιπλέον, θα πρέπει να χειριστείτε σωστά την περίπτωση των απλών και ισχυρών συνδέσμων (soft/hard links). Οι σύνδεσμοι θα πρέπει να διατηρούνται και στη νέα ιεραρχία. Πιο συγκεκριμένα, οι απλοί σύνδεσμοι πρέπει να δείχνουν στο αντίστοιχο μονοπάτι στη νέα ιεραρχία. Στη περίπτωση των ισχυρών συνδέσμων, πρέπει τα δεδομένα των αρχείων να μην αντιγράφονται στην τελική ιεραρχία πάνω από μία φορά.

Υπόβαθρο:

Τα αρχεία στο λειτουργικό σύστημα Linux καταγράφονται σε δύο διαφορετικά επίπεδα, των ονομάτων και των κόμβων-δεικτών (i-nodes). Κάθε i-node συνδέεται με τα δεδομένα ενός και μόνο αρχείου και τηρεί όλες τις πληροφορίες εκτός από το όνομα του αρχείου (π.-χ. αριθμός i-node, ημερομηνία τελευταίας αλλαγής, μέγεθος). Η σχέση είναι αμφιμονοσήμαντη, καθώς κάθε αρχείο δεικτοδοτείται από ένα και μόνο i-node. Όμως ένα i-node μπορεί να είναι συνδεδεμένο με πολλαπλά ονόματα αρχείων (αυτό επιτυγχάνουν οι σύνδεσμοι, τα λεγόμενα hard links). Τα ονόματα δημιουργούν ένα κατευθυνόμενο άκυκλο γράφο (δέντρο) καθώς:

- Οι κατάλογοι, που και αυτοί είναι ονόματα, περιλαμβάνουν με τη σειρά τους άλλα ονόματα.
- Δεν επιτρέπονται οι σύνδεσμοι σε καταλόγους, οπότε αποκλείονται οι κύκλοι.

Ένας τρόπος να δούμε το i-node ενός αρχείου είναι η εντολή `ls -i` (σε επίπεδο προγράμματος συστήματος) όπως και η stat (σε επίπεδο κλήσης συστήματος). Ειδικά στο Linux υπάρχει και το πρόγραμμα συστήματος με το ίδιο όνομα stat που κάνει την ίδια δουλειά σε επίπεδο κελύφους.

Η δημιουργία ισχυρών συνδέσμων σε υπάρχον αρχείο γίνεται με την εντολή `ln <existing_name> <new_name>` και με παράμετρο `-s` για απλούς συνδέσμους. Για τον έλεγχο των απλών (συμβολικών) συνδέσμων μπορεί να χρησιμοποιηθεί η εντολή `readlink` τόσο σε επίπεδο προγράμματος συστήματος αλλά και κλήσης συστήματος.

Προφανώς, η χρήση οποιουδήποτε προγράμματος συστήματος (π.χ. `cp -R`) για την παροχή της απαιτούμενης λειτουργικότητας δεν επιτρέπεται.

Γραμμή Κλήσης της Εφαρμογής:

Η εφαρμογή μπορεί να κληθεί με τον παρακάτω αυστηρό τρόπο στη γραμμή εντολής (`tty`):

```
./cmpcat -d dirA DirB -s DirC
```

όπου `./cmpset` είναι το εκτελέσιμο του του προγράμματος σας που μπορεί να έχει τουλάχιστον τις παρακάτω σημαίες:

- `-d` δείχνει τα ονόματα 2 καταλόγων `dirA DirB` για τους οποίους θα πρέπει να δημιουργηθεί έξοδος διαφορών.
- `-s` υποδηλώνει το όνομα του κατάλογου που θα πρέπει να δημιουργηθεί `DirC` και θα είναι η συγχώνευση των 2 καταλόγων `dirA & DirB`. Η ύπαρξη `-s` προϋποθέτει την ύπαρξη του `-d`.

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

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 κείμενο είναι αρκετές).
2. Οπωσδήποτε ένα `Makefile` που να μπορεί να χρησιμοποιηθεί για να γίνει αυτόματα compile το πρόγραμμα σας.
3. Μια πλήρη σειρά από τεστ τα οποία θα πρέπει να δημιουργήσετε και να περιλαμβάνουν όλες τις περιπτώσεις διαφοροποίησης μεταξύ 2 υπαρχόντων καταλόγων.
4. Ένα `zip/7z` αρχείο με όλη σας τη δουλειά σε έναν κατάλογο που πιθανώς να φέρει το όνομα σας και θα περιέχει όλη σας τη 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%
Ορθότητα Αποτελέσματός για ιεραρχίες καταλόγων/αρχείων	40%
Διαχείριση λογικών συνδέσμων	10%
Διαχείρισή σκληρών συνδέσμων	15%
Χρήση Makefile & Separate Compilation	06%
Ορθή Χρήση Σημαιών σε Γραμμές Εντολών	04%
Επαρκής/Κατανοητός Σχολιασμός Κώδικα	05%

Άλλες Σημαντικές Παρατηρήσεις:

1. Η εργασία μπορεί να γίνει από ομάδες των **πολύ 2 ατόμων**. Αν θέλετε μπορείτε να την κάνετε και ατομικά.
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 όποιος/-οι υποβάλουν 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. Σε καμιά περίπτωση τα Windows *δεν είναι επιλέξιμη* πλατφόρμα για την υλοποίηση αυτής της άσκησης.