ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
3η Εργασία - Τμήμα: Αρτίων Αριθμών Μητρώου
K22: Λειτουργικά Συστήματα – Χειμερινό Εξάμηνο '15
Ημερομηνία Ανακοίνωσης: Δευτέρα 30 Νοεμβρίου
Ημερομηνία Υποβολής: Τρίτη 29 Δεκεμβρίου και Ώρα 23:59

Εισαγωγή στην Εργασία:

Ο στόχος αυτής της εργασίας είναι να δημιουργήσετε ένα αριθμό από ανεξάρτητα προγράμματα τα οποία μπορούν να τρέχουν *ταυτόχρονα* και να *υλοποιούν συνεργατικά* μια διαδικασία εκλογής.

Υπάρχουν τρεις τύποι προγραμμάτων που επιδιώκουν να εξομοιώσουν το πρόβλημα αυτοματοποιημένης εκλογής χωρίς να μπαίνουμε σε θέματα εμπιστευτικότητας και κρυπτογράφησης. Μια οποιαδήποτε διαδικασία εκλογής έχει: *1*) ένα οριζόμενο αριθμό από *ψηφοφόρους* (*voters*) που ο κάθε ένας μπορεί να δώσει την γνώμη του για ένα θέμα που έχει τεθεί σε ψηφοφορία, *2*) ένα συγκεκριμένο αριθμό από *ελεγκτές* (*checkers*) που ελέγχουν αν οι προσερχόμενες διεργασίες ανήκουν σε πιστοποιημένους ψηφοφόρους, και *3*) μια διαδικασία *καταμετρητή* (*collector*) που συλλέγει τους συγκεκριμένους ψήφους και κάνει την καταμέτρηση των ψήφων είτε στην διάρκεια είτε στο τέλος της ψηφοφορίας. Για να διευκολύνουμε τα πράγματα μια συγκεκριμένη εκλογή μπορεί να την ξεκινά και να την διαχειρίζεται μια διαδικασία *ενορχηστρωτής* (*orchestrator*). Το Σχήμα 1 δείχνει την συνολική εικόνα της εκλογικής διαδικασίας.

Η κάθε μία από τις παραπάνω διαδικασίες παίζει το (προφανή) ρόλο που της αποδίδεται και ο στόχος είναι οι ψηφοφόροι που μπορεί να εμφανίζονται σε διάφορες στιγμές στην διάρκειά της ψηφοφορίας και να μπορούν να εκφράσουν την γνώμη τους για το θέμα που έχει τεθεί. Όλα τα παραπάνω προγράμματα που προφανώς έχουν διαφορετικό κώδικα, θα πρέπει να μπορούν να εκτελούνται *ταυτόχρονα* και να *συγχρονίζονται* όπου αυτό κρίνεται απαραίτητο.

Η λίστα με τους εγγεγραμμένους ψηφοφόρους θα πρέπει να μπορεί να διαβάζεται σε ένα shared segment της μνήμης με την βοήθεια του οποίου οι πολλαπλοί ταυτόχρονοι ελεγκτές θα μπορούν να κάνουν σωστά την δουλειά τους. Επίσης, οι ταυτόχρονες διαδικασίες θα πρέπει να χρησιμοποιούν μηχανισμούς *buffering* όπου αυτό κρίνεται απαραίτητο. Στο τέλος της εκλογής, ο συλλέκτης ψήφων θα πρέπει να εκδώσει το αποτέλεσμα.

Σε αυτή την άσκηση θα πρέπει να:

1. χρησιμοποιήσετε ένα σετ από σηματοφόρους ώστε να έχετε μια επιτυχή συνεργασία μεταξύ των ανεξάρτητων διεργασιών, *shared memory segment* για την σωστή επεξεργασία της λίστας ψηφοφόρων, όπως και επίσης να δημιουργήσετε buffers όπου χρειάζονται,
2. έχετε όλες τις διεργασίες να προσαρτήσουν το παραπάνω *shared memory segment* ώστε να μπορούν να προσπελάσουν περιεχόμενα ενδιαφέροντός τους,
3. χρησιμοποιήστε *POSIX Semaphores* για να υλοποιήσετε τη λύση σας που θα εμπεριέχει σχετικές κλήσεις P() και V(),
4. παρέχετε τη δυνατότητα τα προγράμματά σας να παραμείνουν 'απασχολημένα' δηλ. να κάνουν `sleep()` για περιόδους χρόνου που καθορίζονται με παραμετρικό τρόπο.

Θα πρέπει να *επιδείξετε την ορθότητά* της λύσης σας.

Για να έχετε κάποια ευελιξία στην λύση που τελικά θα προτείνετε, θα μπορούσατε να υιοθετήσετε μια διεργασία ενορχηστρωτή που δημιουργεί με `fork()` άλλες και να ορίζει το κώδικα που χρειάζεται να τρέξουν με την βοήθεια `exec*()` κλήσεων. Η παραπάνω διαδικασία ενορχήστρωσης μπορεί να δημιουργεί τους semaphores, το shared segment, και οτιδήποτε άλλο χρειάζεται. Διαφορετικά, μπορείτε να χρησιμοποιείτε πολλαπλά ttys για να

ξεκινήσετε χειρωνακτικά ότι διαδικασία θέλετε να έχετε στην λύση σας.

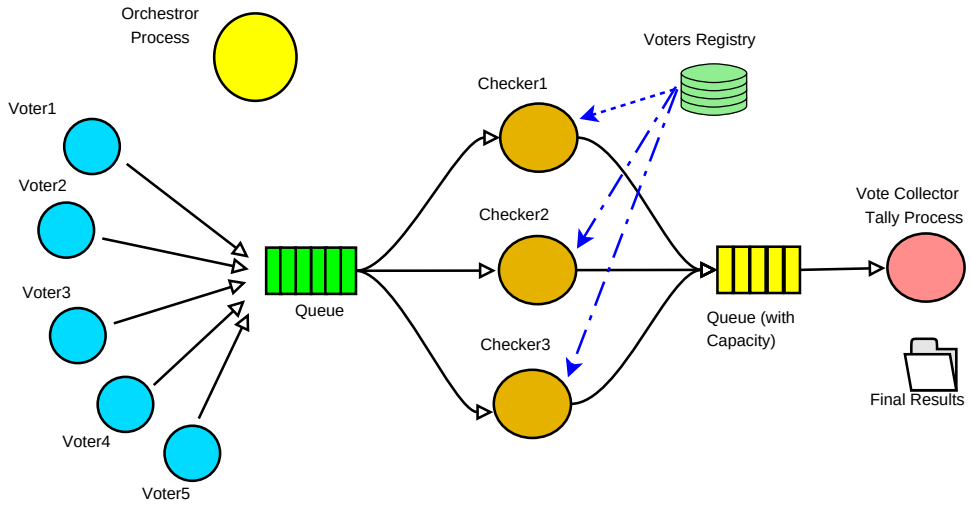
Διαδικαστικά:

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` του τμήματος. Παρακολουθείτε την ιστοσελίδα του μαθήματος για επιπρόσθετες ανακοινώσεις στο URL `http://www.di.uoa.gr/~ad/k22`.

- Υπεύθυνοι για την άσκηση αυτή (ερωτήσεις, αξιολόγηση, βαθμολόγηση κλπ.) είναι η κ. Αργυρώ Κοκοργιαννάκη (`argirok+AT-di`) και ο κ. Πάρης Ιωακειμίδης (`parioak+AT-di`).
- Μέσα από την σελίδα `piazza.com/uoa.gr/fall2015/k22sectiondelis/home`, θα μπορείτε να κάνετε ερωτήσεις και να δείτε απαντήσεις ή/και διευκρινήσεις που δίνονται σχετικά με την άσκηση (όπως έγινε με τις δύο προηγούμενες ασκήσεις).

Διατύπωση του Προβλήματος:

Το Σχήμα 1 παρέχει μια προτεινόμενη βασική οργάνωση διεργασιών που δουλεύουν ταυτόχρονα για να επιτύχουν συνεργατική εκλογή.



Σχήμα 1: Διάγραμμα Διαδικασίας Εκλογής για Υποψηφίους/Θέματα προς Ψήφιση

Η διεργασία ενορχηστρωτής μπορεί να δημιουργεί 3 ελεγκτές, ένα καταμετρητή, ένα shared segment για να αποθηκεύσει το κατάλογο με τους εγγεγραμμένους ψηφοφόρους, καθώς επίσης και τους απαραίτητους σηματοφόρους για την ορθή ταυτόχρονη εκτέλεση των ταυτοχρόνων διεργασιών.

Ένας η πιο πολλοί ψηφοφόροι εμφανίζονται σε τυχαίες χρονικές στιγμές (δηλ. μπορούν να δημιουργούνται από τον ενορχηστρωτή η τα ttys του χρήστη) και μπαίνουν με μία ουρά η οποία δεν έχει προκαθορισμένο μήκος. Οι τρεις ελεγκτές δουλεύουν με τους ψηφοφόρους που είναι διαθέσιμοι στην παραπάνω ουρά. Προφανώς αν υπάρχουν λιγότεροι ψηφοφόροι, μόνο οι απαραίτητοι ελεγκτές εργάζονται και υπόλοιποι απλά περιμένουν.

Ο κάθε ελεγκτής παίρνει το Αριθμό Μητρώου του ψηφοφόρου και ελέγχει να δει ότι: 1) μπορεί να ψηφίσει (δηλ. ότι πράγματι εμφανίζεται στο κατάλογο), και 2) δεν έχει ήδη ξαναψηφίσει στην ίδια εκλογή. Στην διάρκεια του ελέγχου αυτού, ο ελεγκτής ενημερώνει και την σχετική πληροφορία για την ψηφοφόρο στον κατάλογο. Αν όλα προχωρήσουν καλώς με τον έλεγχό, ο ψηφοφόρος προχωρά στην περιοχή όπου μπορεί να περιμένει για να ρίξει την προτίμησή του. Η εν λόγω περιοχή είναι περιορισμένη και μπορεί να φιλοξενήσει μόνο λ μέγιστο αριθμό διεργασιών ψηφοφόρων.

- Αν υπάρχει χώρος, ο εκλογέας αποδεσμεύεται απο τον ελέγκτή του και περιμένει σε μία απο της λ θέσεις. Όποτε ο καταμετρητής είναι διαθέσιμος, 'καλεί' το επόμενο εκλογέα να καταθέσει την ψήφο του.
- Αν δεν υπάρχει χώρος στις λ θέσεις, ο ψηφοφόρος παραμένει με τον ελεγκτή του μέχρι να υπάρξει τελικά χώρος (δηλ. κάποια άλλη διαδικασία να αναχωρήσει αφού προχωρήσει σε τελική ψηφοφορία). Στην διάρκεια αυτής της αναμονής, ο ελεγκτής δεν μπορεί να κάνει τίποτα άλλο εκτος απο το να περιμένει.

Η διαδικασία καταμετρητής απλά διεκπεραιώνει την επιθυμία του κάθε εκλογέα. Για αυτό το λόγο διατηρεί μια δομή καταμέτρησης και κάνει την σχετική καταχώρηση που επιθυμεί ο ψηφοφόρος. Στην εν λόγω δομή, ο καταμετρητής ουσιαστικά καταγράφει ποιος υποψήφιος/θέμα έχει πάρει πόσους ψήφους.

Οι ψηφοφόροι παραμένουν δηλ. κάνουν `sleep()` για τυχαία χρονικά διαστήματα, με τους ελεγκτές (για να επιτευχθεί ο έλεγχος) αλλά και με τον καταμετρητή. Όταν όλοι οι εκλογείς έχουν ελεχθεί και ολοκληρώσουν την εκλογή (ή έχουν αποκλειστεί), ο καταμετρητής παρουσιάζει το αποτέλεσμα της εκλογής, οι διεργασίες απελευθερώνουν τις δομές που χρησιμοποιούν και όλοι τερματίζουν.

Κανόνες Διεξαγωγής Εκλογής και Ανακοίνωσης Αποτελεσμάτων:

Οι κανόνες που τα συνεργαζόμενα προγράμματα πρέπει να εφαρμόσουν είναι οι παρακάτω:

1. Οι εκλογείς έχουν ένα μοναδικό 8-ψήφιο Αριθμό Μητρώου (ΑΜ) και αν έχουν εγγραφεί για την εκλογή οι εν λόγω αριθμοί μητρώου θα πρέπει να υπάρχουν στον σχετικό κατάλογο.
2. Εκλογείς που δεν είναι εγγεγραμμένοι στο κατάλογο δεν ψηφίζουν.
3. Εκλογείς που επιχειρούν να ψηφίσουν για δεύτερη φορά θα πρέπει να 'σημειωθούν' αναλόγως από τους ελεγκτές, να αποτραπούν από την ψηφοφορία για δεύτερη η πολλαπλή φορά (αν χρειαστεί) και στο τέλος να παραχθεί μια σχετική λίστα με όλους τους ψηφοφόρους που επιχείρησαν να πάρουν μέρος στην ψηφοφορία πολλαπλές φορές.
4. Οι εκλογείς μπορούν να επιλέξουν μία από τις επιλογές του ψηφοφορίας (π.χ. επιλογή για *«Τοποθέτηση 3»* ή για την *«Κοκογιαννάκη»*), ή απλά να επιλέξουν *«λευκό»*. Οι επιλογές των ψηφοφόρων είναι είτε έγκυρες ή «λευκές».
5. Ο κάθε εκλογέας παραμένει με τον ελεγκτή του για ένα τυχαίο χρονικό διάστημα την διάρκεια του οποίου ελέγχει ο ελεγκτής.
6. Ο κάθε εκλογέας παραμένει με τον καταμετρητή για ένα τυχαίο χρονικό διάστημα την διάρκεια του οποίου ελέγχει ο καταμετρητής.
7. Η εκλογική διαδικασία ολοκληρώνεται αφότου όλοι οι ψηφοφόροι που επιθυμούν να συμμετάσχουν περάσουν το έλεγχο και όσοι τον περάσουν επιτυχώς, ψηφίσουν.
8. Στο τέλος της ψηφοφορίας, είτε ο καταμετρητής είτε ο ενορχηστρωτής παρουσιάζουν τα αποτελέσματα δίνοντας τις επιλογές που πήραν ψήφους σε φθίνουσα σειρά δημοτικότητας αλλά και τον αριθμό των επιχειρούντων να ψηφίσουν, των ψηφισάντων, των έγκυρων ψήφων, των λευκών ψηφοδελτίων, και των σχετικών ποσοστώσεων για όλα τα παραπάνω.
9. Επίσης θα πρέπει να παραχθεί μια λίστα με όλους τους ψηφοφόρους που επιχείρησαν να ψηφίσουν πάνω από μία φορά.

Στοιχεία Εξόδου:

Τα προγράμματά σας θα πρέπει να παρέχουν τα εξής στοιχεία αποτελεσμάτων:

α) Αριθμός εκλογέων που επιχείρησαν να πάρουν μέρος στην ψηφοφορία.
β) Αριθμός ψηφοφόρων που πέρασαν τον έλεγχο, όσων δεν ήταν εγγεγραμμένοι και εκείνων που επιχείρησαν να διπλοψηφίσουν.
γ) Αριθμός ψήφων για κάθε επιλογή που υπήρξε στο ψηφοδέλτιο.
δ) Ποσοστώσεις για κάθε επιλογή της εκλογής καθώς επίσης και ποσοστό λευκών ψήφων.

ε) Λίστα με τους αριθμούς μητρώου και την ώρα που κάθε ένας από τους εν λόγω επιχείρησε να διπλοψήφισει.
ζ) Αριθμός ατόμων που προσήλθαν στην ψηφοφορία αλλά δεν είχαν γραφτεί, και τελικά δεν ψήφισαν.

Σχεδιασμός των Προγραμμάτων σας:

Έχετε ελευθερία να επιλέξετε οποιαδήποτε δομή επιθυμείτε για τα προγράμματα σας.

Θα πρέπει να υιοθετήσετε ένα (περιορισμένο) αριθμό από σηματοφόρους, βοηθητικές δομές και πιθανώς τουλάχιστον τρία προγράμματα. Όπως έχουμε αναφέρει, ίσως θα ήταν καλή ιδέα να αναπτύξετε ένα πρόγραμμα (τον ενορχηστρωτή) που δημιουργεί το shared segment, αρχικοποιεί δομές και καταστάσεις, και τέλος γνωστοποιεί το ID του κοινού τμήματος σε άλλα ενδιαφερόμενα εκτελέσιμα από την γραμμή εντολής τους. Το εν λόγω πρόγραμμα θα μπορούσε να αρχικοποιήσει και τους σηματοφόρους που είναι απαραίτητοι για την λύση που επιθυμούμε όπως επίσης και τους buffers και τα μεγέθη τους.

Τα προγράμματά σας θα πρέπει να δημιουργούν εξόδους που με εύκολα κατανοητό τρόπο να μπορούν να δείξουν την ορθότητα αλλά και το ταυτόχρονο της εκτέλεσής τους με άλλες διεργασίες.

Στο τέλος της εκτέλεσης όλων των αναγνωστών/συγγραφέων, επιβάλλεται κάποιο πρόγραμμα να *καθαρίσει* και να *διαγράψει* το κοινό τμήμα μνήμης και τους σηματοφόρους που χρησιμοποιήθηκαν. Η απελευθέρωση (purging) τέτοιων πόρων είναι επιτακτική. Σε διαφορετική περίπτωση υπάρχει κίνδυνος ο πυρήνας να μην μπορεί να εξυπηρετήσει μέλλουσες ανάγκες σε κοινή μνήμη και σηματοφόρους.

Γραμμή Κλήσης των Προγραμμάτων:

Ο ελεγκτής μπορεί να κληθεί ως εξής:
./checker -d period -s shmid
όπου

- *checker* είναι το εκτελέσιμο του ελεγκτή,
- η σημαία *-d period* παρέχει την μέγιστη δυνατή χρονική διάρκεια ($t_{checker}$) για την οποία ο ελεγκτής θα παραμείνει 'εργαζόμενος' (δηλ. σε κατάσταση `sleep()`) αφότου έχει επιτυχώς αρχίσει να συνεργάζεται με ένα ψηφοφόρο. Στην πραγματικότητα θα πρέπει να επιλέξετε ένα τυχαίο αριθμό στο διάστημα $[1 \dots t_{checker}]$ το οποίο ο ψηφοφόρος θα δαπανήσει με τον ελεγκτή,
- η σημαία *-s shmid* δίνει το κλειδί που το κοινό τμήμα μνήμης έχει (και όπου βρίσκονται buffers, σηματοφόροι, και οποιαδήποτε άλλη βοηθητική δομή/μεταβλητή που απαιτείται).

Το πρόγραμμα για του καταμετρητή μπορεί να κληθεί εξής:
./counter -i registry -d period -o statsfile -s shmid
όπου

- *counter* είναι το εκτελέσιμο του καταμετρητή,
- η σημαία *-i* ορίζει το αρχείο *registry* με τους εγγεγραμμένους εκλογείς στην ψηφοφορία,
- η σημαία *-d period* παρέχει την μέγιστη χρονική διάρκεια ($t_{counter}$) για την οποία ένας ψηφοφόρος θα παραμείνει με το καταμετρητή (δηλ. σε κατάσταση `sleep()`) για να 'παραδώσει' την επιλογή του. Ο πραγματικός χρόνος αναμονής επιλέγεται τυχαία στο διάστημα $[1 \dots t_{counter}]$,
- η σημαία *-o statsfile* ορίζει το αρχείο εξόδου του προγράμματος,
- η σημαία *-s shmid* δίνει το κλειδί που το κοινό τμήμα μνήμης έχει (και όπου βρίσκονται buffers, σηματοφόροι, και οποιαδήποτε άλλη βοηθητική δομή/μεταβλητή που απαιτείται).

Ένα ψηφοφόρος μπορεί να καλεστεί ως εξής:
./voter -s regnum -c choice -s shmid
όπου

- *voter* είναι το εκτελέσιμο του κάθε ψηφοφόρου,
- η σημαία *-s regnum* καθορίζει τον αριθμό μητρώου (ΑΜ) που προτίθεται να χρησιμοποιήσει ο εκλογέας,
- η σημαία *-c choice* ένας αριθμός που δείχνει την επιλογή του εκλογέα. Το 0 μπορεί να είναι το Λευκό, το 1 να δίνει την πρώτη επιλογή του ψηφοδελτίου, το 2 την δεύτερη, κοκ.
- η σημαία *-s shmid* δίνει το κλειδί που το κοινό τμήμα μνήμης έχει (και όπου βρίσκονται buffers, σηματοφόροι, και οποιαδήποτε άλλη βοηθητική δομή/μεταβλητή που απαιτείται).

Εάν υιοθετήσετε τον ενορχηστρωτή σαν ανεξάρτητο πρόγραμμα τότε μπορεί να κληθεί ως εξής:
./orchestrator -i registry -n numvoters -d1 period1 d2 period2 -c choices -o statsfile -l seats
όπου

- *orchestrator* είναι το εκτελέσιμο του ενορχηστρωτή,
- η σημαία *-i* ορίζει το αρχείο *registry* με τους εγγεγραμμένους εκλογείς στην ψηφοφορία,
- η σημαία *-n* ορίζει τον αριθμό *numvoters* των ψηφοφόρων που θα πάρουν μέρος στην ψηφοφορία,
- η σημαία *-l* δίνει τον αριθμό των 'θέσεων' που ελεγμένοι ψηφοφόροι μπορούν να χρησιμοποιήσουν καθώς μπορεί να βρίσκονται σε αναμονή για ψηφοφορία,
- η σημαία *-o statsfile* ορίζει το αρχείο εξόδου του προγράμματος,
- η σημαία *-d1 period1* παρέχει τη μέγιστη χρονική διάρκεια ($t_{checker}$) για την οποία οι ελεγκτές θα εργάζονται με ψηφοφόρους δηλ. θα είναι σε κατάσταση `sleep()`,
- η σημαία *-d2 period2* παρέχει τη μέγιστη χρονική διάρκεια ($t_{counter}$) για την οποία ο καταχωρητής θα εργάζεται με κάθε ψηφοφόρο δηλ. θα είναι σε κατάσταση `sleep()`,
- η σημαία *-c choices* παρέχει το αρχείο με το ψηφοδέλτιο (και ουσιαστικά δίνει τις επιλογές). Μπορεί αν έχει για παράδειγμα τα εξής στοιχεία: *0) Λευκό 1) Κοκογιαννάκη 2) Χρόνης 3) Ιωακιμείδης 4) Δελής,*
- η σημαία *-s shmid* δίνει το κλειδί που το κοινό τμήμα μνήμης έχει (και όπου βρίσκονται buffers, σηματοφόροι, και οποιαδήποτε άλλη βοηθητική δομή/μεταβλητή που απαιτείται).

Η σειρά με την οποία εμφανίζονται οι σημαίες δεν είναι προκαθορισμένη. Προφανώς μπορείτε να χρησιμοποιήσετε πιο πολλές σημαίες στα παραπάνω προγράμματα ώστε να διευκολυνθείτε ή και να σχεδιάσετε τα προγράμματά σας *ριζικά διαφορετικά*.

Τέλος καλό υα ήταν –και για λόγους επιβεβαίωσης και φυσικά μόνο για το σκοπό της άσκησης– να υπάρχει ένα μηχανισμός logging με την μορφή ενός append-only αρχείου. Εδώ, οι ψηφοφόροι δηλώνουν τι προτίθενται να ψηφίσουν και την πρόθεση αυτή κάποιος ανεξάρτητος παρατηρητής μπορεί να τήν χρησιμοποιήσει σε αντιπαράθεση με το αποτέλεσμα της εξόδου του καταμετρητή για επαλήθευση.

Τι πρέπει να Παραδοθεί:

1. Μια σύντομη και περιεκτική εξήγηση για τις επιλογές που έχετε κάνει στο σχεδιασμό του προγράμματος σας (1-2 σελίδες σε ASCII κειμένου είναι αρκετές).
2. Ενα Makefile (που να μπορεί να χρησιμοποιηθεί για να γίνει αυτόματα το compile του προγράμματος σας).
3. Ένα tar-file με όλη σας την δουλειά σε έναν κατάλογο που πιθανώς να φέρει το όνομά 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 σε* `Linux` αλλιώς *δεν θα 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. Σε καμιά περίπτωση τα MS-Windows *δεν είναι επιλέξιμη* πλατφόρμα για την παρουσίαση αυτής της άσκησης.