Εργασία 4 – Αυτόματος έλεγχος και βαθμολόγηση προγραμμάτων C

Αναπτύξτε μια εφαρμογή για τον αυτόματο έλεγχο και βαθμολόγηση προγραμμάτων C. Η επιθυμητή λειτουργικότητα περιγράφεται παρακάτω. Αποθηκεύστε το κυρίως πρόγραμμα της εφαρμογής σε αρχείο με όνομα `project4.c`.

Ορίσματα

Το κυρίως πρόγραμμα της εφαρμογής δέχεται ως ορίσματα:

(α) Το όνομα ενός αρχείου, στη μορφή `<progname>.c`, που περιέχει τον κώδικα του προγράμματος προς βαθμολόγηση.

(β) Το όνομα ενός αρχείου κειμένου `<progname>.args` που περιέχει τα ορίσματα που απαιτούνται για την εκτέλεση του προγράμματος `<progname>.c`, σε μία γραμμή, με χαρακτήρα ' ' (space) ανάμεσα σε διαδοχικά ορίσματα. Δε μπορείτε να κάνετε υποθέσεις για το μήκος της γραμμής ούτε για τον αριθμό των ορισμάτων.

(γ) Το όνομα ενός αρχείου κειμένου `<progname>.in` που περιέχει τα δεδομένα που θα δεχτεί το πρόγραμμα `<progname>.c` από τη συμβατική είσοδο.

(δ) Το όνομα ενός αρχείου κειμένου `<progname>.out` που περιέχει την αναμενόμενη έξοδο που πρέπει να παράξει το πρόγραμμα `<progname>.c` όταν εκτελεστεί με τα ορίσματα που προσδιορίστηκαν στο αρχείο `<progname>.args` και την είσοδο που προσδιορίστηκε στο αρχείο `<progname>.in`.

Υπολογισμός τελικού βαθμού και έξοδος

Ο τελικός βαθμός που θα ανατεθεί από την εφαρμογή σας στο πρόγραμμα `<progname>.c` προκύπτει από το άθροισμα των επιμέρους βαθμών στις παρακάτω κατηγορίες:

(α) Μεταγλώττιση: -100 αν η μεταγλώττιση απέτυχε, -10 αν η μεταγλώττιση πέτυχε αλλά παρήγαγε κάποιο warning, διαφορετικά 0.

(β) Ομοιότητα εξόδου με την αναμενόμενη: το ακέραιο πηλίκο `<πλήθος όμοιων bytes> * 100 / max(<πλήθος bytes αναμενόμενης εξόδου>, <πλήθος bytes εξόδου>)`.

(γ) Ποινή λόγω λανθασμένης προσπέλασης μνήμης : -15 αν το πρόγραμμα τερμάτισε λόγω σήματος SIGSEGV ή SIGABRT ή SIGBUS.

Η εφαρμογή σας στο τέλος εκτυπώνει στη συμβατική έξοδο τα εξής μηνύματα:

```
Compilation: X

Output: Y

Memory access: Z

Total: T
```

με χαρακτήρα `'\n'` ~~πριν και~~ μετά από κάθε μήνυμα. Οι τιμές `X`, `Y`, `Z` είναι οι βαθμοί που υπολογίστηκαν στις αντίστοιχες επιμέρους κατηγορίες όπως περιγράφονται στην προηγούμενη παράγραφο, ενώ το `T` είναι ίσο με `max(X+Y+Z, 0)`. Τα μηνύματα εκτυπώνονται ακόμη κι αν οι αντίστοιχοι βαθμοί είναι μηδέν.

Λειτουργία

Αν δεν έχει δοθεί ο σωστός αριθμός ορισμάτων με βάση την παραπάνω περιγραφή, η εφαρμογή εκτυπώνει κατάλληλο μήνυμα λάθους στην έξοδο λαθών και τερματίζει επιστρέφοντας 2.

Αν έχει δοθεί ο σωστός αριθμός ορισμάτων, τότε:

(α) Το κυρίως πρόγραμμα της εφαρμογής δημιουργεί μία νέα διεργασία P1 και ανακατευθύνει την έξοδο λαθών της σε ένα αρχείο με όνομα `<progname>.err` . Η P1 παράγει το εκτελέσιμο για το πρόγραμμα `<progname>.c` εκτελώντας την εντολή:

```
gcc -Wall <progname>.c -o <progname>
```

Το κυρίως πρόγραμμα περιμένει να τερματίσει η P1 και μετά ελέγχει αν το αρχείο `<progname>.err` περιλαμβάνει (τουλάχιστον μια φορά) ως μεμονωμένη λέξη τη συμβολοσειρά "`error:`" πράγμα που σημαίνει ότι η μεταγλώττιση απέτυχε ή, αν όχι, τη συμβολοσειρά "`warning:`" το οποίο σημαίνει ότι το πρόγραμμα `<progname>.c` μεταγλωττίζεται επιτυχώς αλλά με warnings.

(β) Εφόσον η μεταγλώττιση είναι επιτυχής, το κυρίως πρόγραμμα δημιουργεί δύο νέες διεργασίες P2 και P3 και έναν ανώνυμο αγωγό A, και ανακατευθύνει τη συμβατική είσοδο της P2 στο αρχείο κειμένου `<progname>.in`, τη συμβατική έξοδο της P2 στο άκρο εγγραφής του αγωγού A και τη συμβατική είσοδο της P3 στο άκρο ανάγνωσης του αγωγού A.

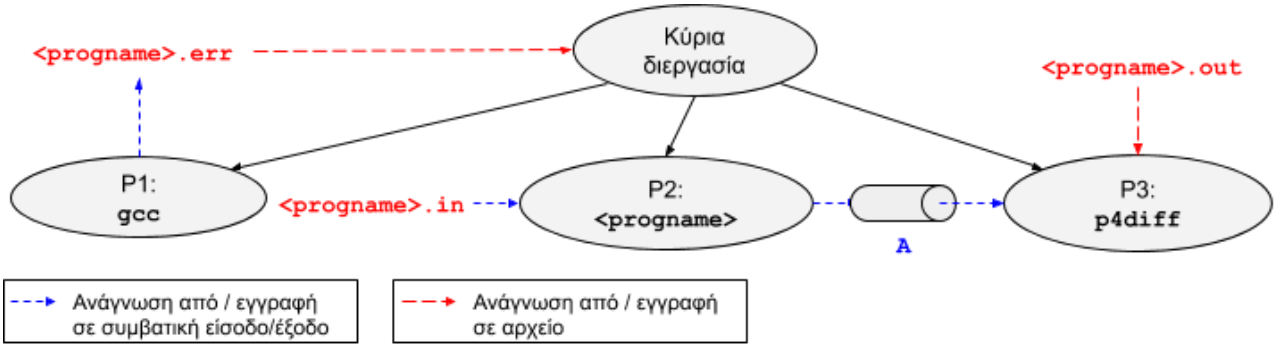
Η διεργασία P2 εκτελεί το πρόγραμμα `<progname>` με τα ορίσματα που προσδιορίζονται στο αρχείο `<progname>.args`.

Η διεργασία P3 εκτελεί το πρόγραμμα `p4diff` με όρισμα το όνομα του αρχείου `<progname>.out`. Το πρόγραμμα `p4diff` πρέπει να το υλοποιήσετε εσείς έτσι ώστε να συγκρίνει ένα προς ένα byte τα δεδομένα που διαβάζει από τη συμβατική του είσοδο με τα περιεχόμενα του αρχείου `<progname>.out`, να υπολογίζει το ποσοστό ομοιότητας όπως περιγράφεται στην προηγούμενη ενότητα και να το επιστρέφει ως αποτέλεσμα μέσω `return`. Αποθηκεύστε τον πηγαίο κώδικα του προγράμματος `p4diff` σε αρχείο με όνομα `p4diff.c`.

Το κυρίως πρόγραμμα περιμένει τον τερματισμό της P2 και ελέγχει τον λόγο τερματισμού. Αν η P2 τερματίσει λόγω σήματος SIGSEGV ή SIGABRT ή SIGBUS, υπολογίζεται η ποινή για λανθασμένη προσπέλαση μνήμης όπως περιγράφεται στην προηγούμενη ενότητα (διαφορετικά θεωρείται ότι το πρόγραμμα `<progname>` εκτελέστηκε χωρίς πρόβλημα οπότε δεν υπάρχει κάποια ποινή). Για απλότητα υποθέστε ότι το πρόγραμμα `<progname>` δεν έχει καμία ατέρμονη επανάληψη και άρα η διεργασία P2 τερματίζει πάντα (είτε κανονικά είτε λόγω σφάλματος προσπέλασης μνήμης).

Ακολούθως, το κυρίως πρόγραμμα περιμένει τον τερματισμό της P3, και παίρνει μέσω του exit status της το βαθμό που υπολόγισε η P3.

(γ) Τέλος, εκτυπώνει τα μηνύματα βαθμολογίας όπως περιγράφονται στην προηγούμενη ενότητα.



Απαιτήσεις/Υποθέσεις υλοποίησης

- Παρέχετε κατάλληλο `Makefile` για την εφαρμογή σας.
- Απαγορεύεται η χρήση καθολικών (global) ή static μεταβλητών, goto και η χρήση της εντολής system.
- Οι διεργασίες P2 και P3 τρέχουν ταυτόχρονα.
- Το πρόγραμμα `p4diff` διαβάζει δεδομένα από το αρχείο και τον αγωγό σε blocks των 64 bytes.
- Μπορείτε να υποθέσετε ότι το όνομα του αρχείου `<progname>.c` θα έχει πάντα τη σωστή κατάληξη (`.c`) καθώς και ότι τα ονόματα των αρχείων `<progname>.args`, `<progname>.in` και `<progname>.out` θα έχουν τη σωστή μορφή και κατάληξη.

Σημαντικές ημερομηνίες

Φροντιστήριο: *Τρίτη 19/5/2020* (στην ώρα του μαθήματος)

Προθεσμία υποβολής: *Κυριακή 7/6/2020, 23:59*

Πακετάρισμα και αποστολή εργασίας

1. Κατασκευάστε ένα κατάλογο με όνομα `project4submit`
2. Αντιγράψτε τα `project4.c, p4diff.c, Makefile` στον κατάλογο `project4submit`
3. Πακετάρετε και συμπιέστε τον κατάλογο `project4submit` κάνοντας δεξί κλικ κι επιλέγοντας `Compress here as .tar.gz`
4. Κάντε login στο autolab, κι επιλέξτε το `project4` του μαθήματος ECE116-S20
5. Αποδεχτείτε το μήνυμα ακαδημαϊκής ακεραιότητας και μετά κάντε κλικ στο `SUBMIT`.
6. Στο παράθυρο που θα εμφανιστεί εντοπίστε κι επιλέξτε το αρχείο `project4submit.tar.gz` που κατασκευάσατε ώστε να το ανεβάσετε στο autolab.
7. Μετά από ένα λεπτό, ανανεώσετε τη σελίδα για να δείτε το βαθμό σας στα επιμέρους τεστ.

Για όσους έχουν κέφι

- Αλλάξτε το κυρίως πρόγραμμα ώστε αντί για τα ορίσματα `<progname>.args`, `<progname>.in`, `<progname>.out` να λαμβάνει ως όρισμα το όνομα ενός καταλόγου μέσα στον οποίο βρίσκονται υποκατάλογοι, ένας για κάθε έλεγχο. Σε κάθε υποκατάλογο περιλαμβάνονται τα παρακάτω αρχεία κειμένου:
    - `test.args` με τα ορίσματα που απαιτούνται γι αυτόν τον έλεγχο.
    - `test.in` με τα δεδομένα που αποτελούν τη συμβατική είσοδο γι αυτόν τον έλεγχο.
    - `test.out` με την αναμενόμενη έξοδο του προγράμματος γι αυτόν τον έλεγχο.

Προσαρμόστε αναλόγως τον τρόπο υπολογισμού του τελικού βαθμού θεωρώντας ότι όλα τα τεστ είναι ισότιμα και το συνολικό άθροισμα πόντων (για την κατηγορία output) είναι 100.

- Αλλάξτε το κυρίως πρόγραμμα ώστε να παίρνει ως επιπλέον όρισμα το μέγιστο χρόνο σε δευτερόλεπτα (real time) που επιτρέπεται να τρέχει η P2. Αν η P2 δεν έχει τερματίσει μετά από τόσα δευτερόλεπτα εκτέλεσης, το κυρίως πρόγραμμα της στέλνει σήμα SIGKILL και προσθέτει στα μηνύματα εξόδου του το "`Timeout: t sec`" όπου t ο χρόνος.