Εργασία 1 – Κρεμάλα

Αναπτύξτε ένα πρόγραμμα που υλοποιεί ένα παιχνίδι κρεμάλας ανάμεσα στον υπολογιστή κι ένα παίκτη δίνοντας στον υπολογιστή τη δυνατότητα να "κλέβει" με βάση το γράμμα που δίνει ο παίκτης σε κάθε γύρο. Η επιθυμητή λειτουργικότητα περιγράφεται παρακάτω. Γράψτε το πρόγραμμά σας στο αρχείο με όνομα hw1.c που σας δίνουμε και το οποίο κάνει #include το hw1.h. Μη μεταβάλλετε το αρχείο hw1.h. Επιπλέον, σας δίνουμε μία στατική βιβλιοθήκη libhw1.a που περιέχει τις υλοποιήσεις κάποιων έτοιμων συναρτήσεων που περιγράφονται παρακάτω.

Περιγραφή παιχνιδιού

Στο κλασικό παιχνίδι κρεμάλας, ο υπολογιστής επιλέγει από ένα λεξικό μια λέξη και σε κάθε γύρο την εμφανίζει στην οθόνη έχοντας αντικαταστήσει με παύλες όσα γράμματα δεν έχει μαντέψει ακόμη ο παίκτης. Ο παίκτης, σε κάθε γύρο μαντεύει ένα γράμμα, κι αν αυτό υπάρχει στη λέξη, τότε αντικαθίστανται η αντίστοιχη παύλα. Ο παίκτης κερδίζει αν προλάβει να μαντέψει όλα τα γράμματα πριν τελειώσουν οι γύροι του παιχνιδιού.

Το ζητούμενο είναι να φτιαχτεί μια παραλλαγή όπου ο υπολογιστής κλέβει ως εξής:

Στην αρχή του παιχνιδιού δεν επιλέγει μία συγκεκριμένη λέξη. Όταν ο παίκτης μαντέψει το πρώτο του γράμμα, ο υπολογιστής χωρίζει όλες τις λέξεις που υπάρχουν στο λεξικό σε ομάδες ("κλάσεις ισοδυναμίας") με κριτήριο το αν και πού εμφανίζεται το γράμμα στις λέξεις. Για παράδειγμα, ας υποθέσουμε ότι το λεξικό έχει μόνο τις παρακάτω 10 λέξεις με 4 γράμματα κάθε μια:

arts, dear, deer, feel, flew, pear, pend, perk, poor, rear

Έστω ότι ο παίκτης στον πρώτο γύρο μαντεύει e. Ο υπολογιστής πρέπει να εμφανίσει όλα τα e που βρίσκονται στη λέξη. Πριν δεσμευθεί, χωρίζει τις διαθέσιμες λέξεις στις παρακάτω κλάσεις ισοδυναμίας, με βάση το πού βρίσκεται σε αυτές το γράμμα e:

- λέξεις χωρίς e: arts, poor
- λέξεις όπου το e εμφανίζεται μόνο στη 2η θέση: dear, pear, pend, perk, rear
- λέξεις όπου το e εμφανίζεται στη 2η και στην 3η θέση: deer, feel
- λέξεις όπου το e εμφανίζεται στην 3η θέση: flew

Κάθε κλάση χαρακτηρίζεται από έναν **εκπρόσωπο** που αντικατοπτρίζει την κοινή ιδιότητα των λέξεων της κλάσης. Για παράδειγμα ο εκπρόσωπος της δεύτερης κλάσης είναι "_ e _ _ " γιατί η κλάση περιέχει όλες τις λέξεις όπου το e εμφανίζεται μόνο στη 2η θέση.

Στην συνέχεια, ο υπολογιστής επιλέγει την πολυπληθέστερη κλάση, δηλαδή στο παραπάνω παράδειγμα αυτή με εκπρόσωπο "_ e _ _ " και εμφανίζει το e στη δεύτερη θέση.

Στον επόμενο γύρο, οι λέξεις που παραμένουν στο παιχνίδι είναι μόνο οι λέξεις της κλάσης που επιλέχθηκε προηγουμένως, δηλαδή dear, pear, pend, perk.

Ας υποθέσουμε ότι στο δεύτερο γύρο ο παίκτης μαντεύει r. Οι νέες κλάσεις και οι εκπρόσωποι τους είναι:

- " _ e _ _ " (λέξεις χωρίς r: pend)
- " _ e _ r " (λέξεις όπου το r εμφανίζεται μόνο στην 4η θέση: dear, pear)
- " _ e r _ " (λέξεις όπου το r εμφανίζεται μόνο στην 3η θέση: perk)
- " r e _ r " (λέξεις όπου το r εμφανίζεται στην 1η και στην 4η θέση: rear)

Η πιο πολυπληθής κλάση είναι η δεύτερη με εκπρόσωπο " _ e _ r " , και οι λέξεις που παραμένουν στο παιχνίδι για τον επόμενο γύρο είναι dear και pear.

Ας υποθέσουμε ότι στον τρίτο γύρο ο παίκτης μαντεύει f. Το f δεν υπάρχει σε καμία λέξη, οπότε θα βρεθεί μόνο μια κλάση:

- " _ e _ r " (λέξεις χωρίς f: dear, pear)

Το παιχνίδι συνεχίζει με τον ίδιο τρόπο. Αν τελειώσουν οι γύροι χωρίς να έχει βρει τη λέξη ο παίκτης, τότε ο υπολογιστής εμφανίζει την πρώτη (λεξικογραφικά μικρότερη) λέξη της τρέχουσας κλάσης ισοδυναμίας και ισχυρίζεται πως αυτή ήταν η λέξη που είχε επιλέξει από την αρχή.

Έτοιμος κώδικας

Στο αρχείο hw1.h έχουν οριστεί:

- Ένας τύπος enum που αναπαριστά κωδικό λάθους.
- Η επικεφαλίδα της συνάρτησης getWord η υλοποίηση της οποίας δίνεται στη στατική βιβλιοθήκη. Η συνάρτηση παίρνει ως παράμετρο το όνομα ενός αρχείου και κάθε φορά που καλείται επιστρέφει δείκτη προς την επόμενη λέξη που βρίσκεται στο αρχείο (ως δυναμικά δεσμευμένη συμβολοσειρά) ή NULL αν το αρχείο δεν περιέχει άλλες λέξεις.
- Η επικεφαλίδα της συνάρτησης printErrorMsg η υλοποίηση της οποίας δίνεται στη στατική βιβλιοθήκη. Η συνάρτηση παίρνει ως παράμετρο έναν κωδικό λάθους κι εκτυπώνει αντίστοιχο μήνυμα.

Ορισμοί

Ορίστε ένα struct που αναπαριστά μια κλάση ισοδυναμίας. Το struct περιέχει πεδία για τις εξής ποσότητες: (α) τον εκπρόσωπο (δυναμικά δεσμευμένη συμβολοσειρά), (β) το πλήθος λέξεων της κλάσης, (γ) τις λέξεις που ανήκουν στην κλάση (δυναμικά δεσμευμένος πίνακας από δείκτες σε ήδη υπάρχουσες συμβολοσειρές).

Κατά τη διάρκεια του παιχνιδιού θα χρησιμοποιήσετε ένα δυναμικά δεσμευμένο πίνακα από δείκτες προς κλάσεις ισοδυναμίας.

Υλοποίηση

Το πρόγραμμα δέχεται τρία ορίσματα: (α) το μήκος της λέξης για την κρεμάλα, (β) το πλήθος γύρων, (γ) το όνομα ενός αρχείου-λεξικού.

Γράψτε μια συνάρτηση ελέγχου των ορισμάτων που επιστρέφει κωδικό λάθους και παίρνει ως παραμέτρους: (α) το πλήθος ορισμάτων (argc), (β) τον πίνακα ορισμάτων (argv), (γ) δείκτη σε ακέραιο και (δ) δείκτη σε ακέραιο. Η συνάρτηση λειτουργεί ως εξής:

- Αν έχει δοθεί λάθος πλήθος ορισμάτων, επιστρέφει INVALID_ARGS.
- Αν έχει δοθεί μη-θετικό μήκος λέξης, επιστρέφει INVALID_LEN.
- Αν έχει δοθεί μη-θετικό πλήθος γύρων, επιστρέφει INVALID_TURNS.
- Οι παραπάνω έλεγχοι γίνονται με τη σειρά που περιγράφονται. Εάν δεν διαπιστωθεί κάποιο από τα παραπάνω σφάλματα, αποθηκεύει στις διευθύνσεις μνήμης που δείχνουν οι 3η και 4η παράμετροι το μήκος της λέξης και το πλήθος γύρων αντίστοιχα, κι επιστρέφει OK.

Καλέστε την παραπάνω συνάρτηση στη main. Σε περίπτωση που δεν επιστραφεί OK, καλέστε τη συνάρτηση printErrorMsg με τον κατάλληλο κωδικό λάθους και τερματίστε το πρόγραμμα.

Γράψτε μια συνάρτηση δημιουργίας του λεξικού που παίρνει ως παραμέτρους (α) το όνομα του αρχείου, (β) το επιθυμητό μήκος λέξης και (γ) τη διεύθυνση ενός ακεραίου. Αυτή η συνάρτηση χρησιμοποιεί τη συνάρτηση getWord σε επανάληψη για να διαβάσει τις λέξεις του αρχείου και να τις αποθηκεύσει σε ένα δυναμικά δεσμευμένο πίνακα από δείκτες προς δυναμικά δεσμευμένες συμβολοσειρές. Αν δε βρεθεί καμία λέξη με το επιθυμητό μήκος, η συνάρτηση αποδεσμεύει ότι μνήμη τυχόν δέσμευσε κι επιστρέφει NULL. Διαφορετικά, αποθηκεύει το πλήθος λέξεων του πίνακα στη θέση μνήμης που δείχνει η 3η παράμετρος κι επιστρέφει τη διεύθυνση του πίνακα.

Καλέστε την παραπάνω συνάρτηση στη main. Σε περίπτωση αποτυχίας δημιουργίας του λεξικού, εκτυπώστε στη main το μήνυμα "No words of length X.\n" όπου X το επιθυμητό μήκος και τερματίστε το πρόγραμμα.

Σε επανάληψη μέχρι ο παίκτης να επιλέξει τερματισμό υλοποιήστε το παιχνίδι:

Το παιχνίδι σε κάθε γύρο:

- Εκτυπώνει το μήνυμα "\n== ROUND X/Y ==\n\n" όπου X ο αύξων αριθμός του γύρου ξεκινώντας από 1, και Y το συνολικό πλήθος γύρων.
- Εκτυπώνει το μήνυμα "Used letters: " και μετά ένα-ένα τα γράμματα που έχουν ήδη χρησιμοποιηθεί από τον παίκτη, ταξινομημένα σε αύξουσα σειρά, με ένα κενό πριν από κάθε γράμμα και ένα χαρακτήρα '\n' μετά το τελευταίο.
- Εκτυπώνει το μήνυμα "Unused letters: " και μετά τα υπόλοιπα γράμματα, με τον ίδιο τρόπο.
- Εκτυπώνει το μήνυμα "\nWord: " και τη λέξη της κρεμάλας, με χαρακτήρα underscore ('_') στις θέσεις που δεν έχει βρεθεί γράμμα από τον παίκτη, ένα κενό πριν από κάθε γράμμα και '\n' στο τέλος.
- Εκτυπώνει το μήνυμα "\nGuess a letter: " και διαβάζει ένα χαρακτήρα. Αν αυτός δεν είναι γράμμα, τότε εκτυπώνει το μήνυμα "X is not a letter.\n" όπου X το γράμμα και το βήμα επαναλαμβάνεται. Αν είναι γράμμα αλλά έχει ήδη επιλεγεί σε προηγούμενο γύρο, τότε εκτυπώνει το μήνυμα "You have already guessed X.\n" όπου X το γράμμα και το βήμα επαναλαμβάνεται.
- Κατασκευάζει νέες κλάσεις ισοδυναμίας όπως αυτές διαμορφώνονται από την επιλογή γράμματος. Όλες οι κλάσεις ισοδυναμίας πρέπει να είναι αποθηκευμένες σε δυναμικά δεσμευμένο πίνακα από δείκτες προς αυτές ο οποίος ανανεώνεται κατάλληλα από τον ένα γύρο στον επόμενο.
- Εάν είναι ορισμένο το macro DEBUG (δείτε παρακάτω), εκτυπώνει:
    - Το μήνυμα "\n== CLASSES ==\n"
    - Το πλήθος κλάσεων ισοδυναμίας και δύο χαρακτήρες '\n'
    - Για κάθε κλάση ισοδυναμίας, τα στοιχεία της κλάσης όπως περιγράφεται στις απαιτήσεις υλοποίησης. Οι κλάσεις εκτυπώνονται ταξινομημένες λεξικογραφικά ως προς τους εκπροσώπους.
- Βρίσκει την πολυπληθέστερη κλάση. Σε περίπτωση ισοτιμίας, επιλέγει αυτή που ο εκπρόσωπος της έχει τις περισσότερες παύλες. Σε περίπτωση ισοτιμίας και σε αυτό το κριτήριο, επιλέγει την τελευταία που βρέθηκε (κι έχει τις περισσότερες παύλες).
- Εάν είναι ορισμένο το macro DEBUG, το μήνυμα "\n== CHOSEN ==\n" και ακολούθως τα στοιχεία της κλάσης που επιλέχθηκε.
- Ελέγχει αν υπάρχει νικητής. Το παιχνίδι τελειώνει με ήττα για τον παίκτη αν τελειώσουν οι γύροι χωρίς να έχουν βρεθεί όλα τα γράμματα. Αν ο παίκτης κέρδισε, το πρόγραμμα εκτυπώνει το μήνυμα "\nYou won! The word is X.\n" όπου X η λέξη. Αν έχασε, το πρόγραμμα εκτυπώνει το μήνυμα "\nYou lost! The word is X.\n" όπου X η πρώτη λέξη της τρέχουσας κλάσης ισοδυναμίας.

Όταν ένα παιχνίδι τελειώσει, το πρόγραμμα εκτυπώνει το μήνυμα "\n\nYOU: X - ME: Y\n\n" όπου X ο αριθμός φορών που έχει κερδίσει ο παίκτης και Y ο αριθμός φορών που έχει χάσει. Μετά, εκτυπώνει το μήνυμα "Play again? (y/n)\n" και διαβάζει ένα χαρακτήρα. Αν αυτός δεν είναι 'y'/'Y'/'n'/'N', το βήμα επαναλαμβάνεται. Αν δοθεί 'y' ή 'Y' το παιχνίδι ξεκινά από την αρχή (με το ίδιο λεξικό και μήκος λέξης), διαφορετικά τερματίζει αφού ελευθερώσει όλη τη δυναμικά δεσμευμένη μνήμη.

Μεταγλώττιση

Για να χρησιμοποιήσετε τις συναρτήσεις που ορίζονται στη βιβλιοθήκη libhw1.a πρέπει να διασυνδέσετε τον κώδικα σας με αυτή. Αυτό γίνεται προσθέτοντας στην εντολή μεταγλώττισης τις επιλογές: `-lhw1 -L.`

Επιπλέον, υπάρχει η δυνατότητα να ορίσετε ένα όνομα (στην προκειμένη περίπτωση το DEBUG) ως macro στο στάδιο της μεταγλώττισης προσθέτοντας την επιλογή -D DEBUG. Αυτό είναι ισοδύναμο με το να είχατε την εντολή #define DEBUG 1 στο πρόγραμμα σας. Ο έλεγχος στον κώδικα για το αν έχει οριστεί το macro DEBUG γίνεται ως εξής:

```
#ifdef DEBUG
   /* εδώ μπαίνει ο κώδικας που πρέπει να εκτελεστεί αν το DEBUG έχει οριστεί */
#endif
```

Τελικά, η εντολή μεταγλώττισης σε περίπτωση που επιθυμείτε να έχει οριστεί το DEBUG είναι:

```
gcc -Wall -g -D DEBUG hw1.c -o hw1 -lhw1 -L.
```

Απαιτήσεις υλοποίησης

- Απαγορεύεται η χρήση global ή static μεταβλητών καθώς και η χρήση goto, gets ή fflush.
- Όλες οι κλάσεις ισοδυναμίας που κατασκευάζονται κατά τη διάρκεια του παιχνιδιού πρέπει να περιέχουν δείκτες προς τις ήδη υπάρχουσες συμβολοσειρές του αρχικού πίνακα-λεξικού και όχι αντίγραφα αυτών.
- Το πρόγραμμα όταν δέχεται είσοδο δεν πρέπει να κάνει διάκριση σε κεφαλαία/μικρά. Όταν εκτυπώνει γράμματα, εκπροσώπους ή λέξεις λεξικού, χρησιμοποιεί πάντα κεφαλαία.
- Όλα τα μηνύματα εξόδου που περιέχουν ':' έχουν ένα χαρακτήρα space μετά το ':'.
- Τα στοιχεία μιας κλάσης εκτυπώνονται ως εξής: ο εκπρόσωπος της κλάσης, ένα κενό, το πλήθος λέξεων στην κλάση με πλάτος 3, και ακολούθως οι λέξεις της κλάσης με κενό πριν από κάθε μία, ταξινομημένες λεξικογραφικά σε αύξουσα σειρά, με κεφαλαία γράμματα. Στο τέλος εκτυπώνεται ένας χαρακτήρας αλλαγής γραμμής.
- Οι δυναμικά δεσμευμένοι πίνακες του προγράμματος πρέπει να έχουν ακριβώς όσο μέγεθος χρειάζεται.
- Σε περίπτωση αποτυχίας δέσμευσης μνήμης εκτυπώνετε το μήνυμα "Memory error.\n" και το πρόγραμμα τερματίζει άμεσα με χρήση εντολής exit.
- Φροντίστε να αποδεσμεύετε πάντα σωστά τη δυναμικά δεσμευμένη μνήμη.
- Ορίστε συναρτήσεις για συγκεκριμένες λειτουργίες (π.χ. παίξιμο ενός γύρου, αναζήτηση εκπροσώπου) καθώς και για λειτουργίες που επαναλαμβάνονται σε διάφορα σημεία (π.χ. εκτύπωση των στοιχείων μιας κλάσης.)
- Το πρόγραμμα πρέπει να ακολουθεί τους κανόνες μορφοποίησης, σχολιασμού, κτλ. που έχουν αναρτηθεί στο eclass, καθ' όλη τη διάρκεια της ανάπτυξης του.

Υποβολή κώδικα

Υποβάλετε το hw1.c στο autolab του μαθήματος. Ο ελάχιστος βαθμός που πρέπει να πετύχετε είναι 70.

Το μέγιστο πλήθος υποβολών χωρίς βαθμολογική ποινή είναι 30. Κάθε υποβολή πλέον των 30 έχει ποινή ίση με 1% του τελικού βαθμού.

Παράδοση

Κυριακή 31/3/2024, 21:00

Δεν υπάρχει δυνατότητα καθυστερημένης υποβολής.