Άσκηση 1
Προγραμματισμός Συστήματος

Προθεσμία: 18 Μαΐου 2014

Σ'αυτή την άσκηση θα υλοποιήσετε ένα σύστημα auto-complete κατά τη διάρκεια πληκτρολόγησης. Ο πυρήνας του συστήματος είναι μια δομή trie (απλό δέντρο από προθέματα λέξεων) επαυξημένο με συχνότητες εμφάνισης λέξεων.

Συγκεκριμένα θα χρειαστεί να φορτώσετε ένα εξωτερικό λεξικό σε μια δομή trie. Κάθε μονοπάτι από τη ρίζα μέχρι κάποιο κόμβο της δενδρικής αυτής δομής αντιστοιχεί σε ένα πρόθεμα λέξης, κι έτσι τα κοινά προθέματα λέξεων φυλάσσονται μία φορά εξασφαλίζοντας χώρο μνήμης. Θεωρούμε ότι οι λέξεις μας αποτελούνται μόνο από λατινικούς χαρακτήρες και στη δομή αγνοούνται τα κεφαλαία γράμματα, αν και στην έξοδο διατηρούνται όσα κεφαλαία έχει πληκτρολογήσει ο χρήστης, όπως υποδεικνύουν και τα παραδείγματα εκτέλεσης παρακάτω.

Περιγραφή Δομής. Ένα trie (βλ. `http://en.wikipedia.org/wiki/Trie`) είναι ουσιαστικά μία δενδρική δομή η οποία συσχετίζει κλειδιά τύπου συμβολοσειράς με τιμές (οποιουδήποτε τύπου), κατ'ανάλογο τρόπο με ένα hash-table. Έχει τα εξής χαρακτηριστικά: (i) οι *τιμές* αποθηκεύονται στα φύλλα, (ii) κάθε ακμή επισημαίνεται με ένα σύμβολο, (iii) τα *κλειδιά* δεν αποθηκεύονται ξεχωριστά σε έναν κόμβο το καθένα, αλλά το κλειδί που αφορά μία τιμή σε κάποιο φύλλο σχηματίζεται από την συνένωση όλων των (συμβόλων των) ακμών από τη ρίζα μέχρι και το φύλλο αυτό. Έτσι, για οποιοδήποτε υποδέντρο, τα κλειδιά που αποθηκεύει έχουν κοινό πρόθεμα τη συμβολοσειρά που σχηματίζεται από τη ρίζα του trie μέχρι και τη ρίζα του υποδέντρου αυτού.

Η ακριβής δομή είναι ένα παραλλαγμένο trie με τις εξής τροποποιήσεις/εξειδικεύσεις:

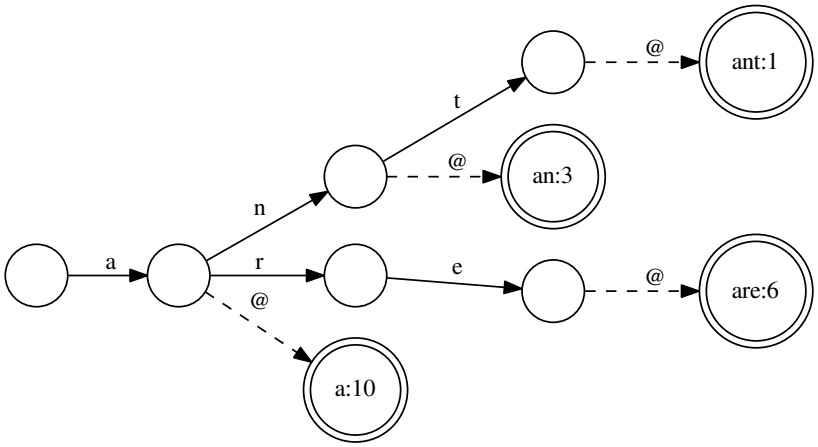
- ολόκληρες λέξεις αναπαρίστανται από φύλλα του δέντρου και όχι από εσωτερικούς κόμβους
- κάθε φύλλο κρατά τη συχνότητα εμφάνισης της λέξης που αντιπροσωπεύει
- κάθε εσωτερικός κόμβος κρατά τα N (όπου N είναι μια compile-time σταθερά[1]) πιο συχνά φύλλα του υποδένδρου του. Η δομή θα πρέπει να γυρνά σε σταθερό χρόνο (δηλαδή χωρίς έρευνα ολόκληρου υποδένδρου) τις N πιο συχνές λέξεις για κάθε πρόθεμα.

Συμβάσεις.

1. Κάποια συμβολοσειρά μπορεί να είναι και πρόθεμα και ολόκληρη λέξη, αλλά οι δύο της αυτές ιδιότητες αναπαρίστανται από ξεχωριστούς κόμβους του δέντρου (εσωτερικούς έναντι φύλλων).
2. Για όλα τα φύλλα η ακμή από τον πατέρα τους έχει την ειδική τιμή `"@"` αντί κανονικού συμβόλου.
3. Οι αριθμοί στα φύλλα αναπαριστούν τη συχνότητα εμφάνισης της εκάστοτε λέξης.

[1] Η σταθερά αυτή θα πρέπει να είναι εύκολα παραμετροποιήσιμη κατά τη μεταγλώττιση.

Ακολουθεί ένα παράδειγμα ενός trie με βάση τις παραπάνω συμβάσεις και παραδοχές.



Αυτό το trie αναπαριστά 4 λέξεις (`a`, `an`, `ant`, `are`) που εμφανίζονται στα φύλλα μαζί με τις συχνότητες εμφάνισής τους. Το trie έχει εσωτερικούς κόμβους που αναπαριστούν τα προθέματα `a`, `an`, `ant`, `ar`, `are`. Σημειώστε πως οι λέξεις που εμφανίζονται στα φύλλα εξυπηρετούν απλά την οπτικοποίηση του δέντρου και δεν θα πρέπει στη πράξη να αποθηκεύονται αυτούσιες σε κάποιο κόμβο, καθώς μπορούν να αναπαραχθούν ακολουθώντας το μονοπάτι από τη ρίζα μέχρι και το εκάστοτε φύλλο.

Η δομή που θα υλοποιήσετε θα είναι λίγο πιο πολύπλοκη από το παραπάνω απλοποιημένο σχήμα ως προς τα εξής:

- Κάθε εσωτερικός κόμβος, n, κρατάει τα N φύλλα με τη μεγαλύτερη συχνότητα στο υποδένδρο με ρίζα τον n. Δηλαδή αν $N = 2$, η ρίζα του παραπάνω δένδρου θα πρέπει να δείχνει (μέσω οποιασδήποτε δομής, της επιλογής σας) στα φύλλα `*a:10*` και `*are:6*`, ο εσωτερικός κόμβος `*an*` θα πρέπει να δείχνει στα φύλλα `*an:3*` και `*ant:1*`, κ.ο.κ. Αν φυσικά δεν υπάρχουν N φύλλα στο υποδένδρο, ο κόμβος δείχνει σε όσα υπάρχουν.

- Κάθε κόμβος χρειάζεται δείκτη στο γονιό του, για ανανέωση της πληροφορίας «πιο συχνών φύλλων στο υποδένδρο». Όταν δηλαδή η συχνότητα ενός φύλλου αλλάξει, αυτή η ανανέωση θα διαδοθεί στο γονιό του και πιθανόν να αλλάξει την πληροφορία των N πιο συχνών λέξεων κάτω από τον γονιό αυτό. Παρόμοια αν η πληροφορία αυτή αλλάξει, θα μεταδοθεί στον γονιό του γονιού, κ.ο.κ., έως ίσως και τη ρίζα του δέντρου.

- Οι συχνότητες ανανεώνονται κάθε φορά που ολοκληρώνεται η πληκτρολόγηση μίας νέας λέξης από τον χρήστη με οποιοδήποτε τρόπο, αυξάνοντας το μετρητή εμφανίσεων της λέξης αυτής.

Λεπτομέρειες Υλοποίησης. Κατά την εκτέλεση του προγράμματος ο χρήστης θα πληκτρολογεί κανονικά κείμενο (το οποίο θεωρούμε στα πλαίσια αυτής της άσκησης ότι θα αποτελείται αυστηρά από λατινικούς χαρακτήρες) ως input του προγράμματος, ωστόσο κάποια πλήκτρα θα έχουν ιδιαίτερη συμπεριφορά.

Αυτά είναι τα ακόλουθα:

TAB
: Το πλήκτρο αυτό θα ενεργοποιεί την εμφάνιση των πιθανών λέξεων, οι οποίες ταιριάζουν με την τρέχουσα λέξη-πρόθεμα (αγνοώντας διαφορές λόγω πεζών ή κεφαλαίων γραμμάτων, δηλαδή το 'Del' είναι έγκυρο πρόθεμα της λέξης 'delete'), και θα τις εμφανίζει στην επόμενη γραμμή, με κενά ανάμεσά τους. Σε περίπτωση που μόνο μία λέξη ταιριάζει, τότε απλά θα συμπληρώνει τη λέξη και η πληκτρολόγηση θα συνεχίζει στην ίδια γραμμή.

  Το (μέγιστο) πλήθος των λέξεων που εμφανίζονται στη νέα γραμμή θα καθορίζεται από τη σταθερά N και θα πρέπει το πρόγραμμα να είναι εύκολα παραμετροποιήσιμο ως προς αυτή, κατά τη μεταγλώττιση. Η default τιμή αυτής της σταθεράς θα είναι 5. Οι λέξεις που εμφανίζονται θα πρέπει επίσης να είναι αλφαριθμητικά ταξινομημένες.

1 . . . 9
: Τα πλήκτρα αυτά θα είναι ενεργά μόνο αφού ο χρήστης έχει πατήσει το πλήκτρο TAB κι έχει εμφανιστεί η γραμμή με τις πιθανές λέξεις για συμπλήρωση. Τότε, πληκτρολογώντας αυτό το νούμερο θα συμπληρώνεται η λέξη στην οποία αυτό αντιστοιχεί (π.χ., με 3 θα διαλέγεται η 3^{η} λέξη).

  Αν το νούμερο που εισήχθηκε είναι μεγαλύτερο από τη σταθερά N ή από τις διαθέσιμες λέξεις, τότε δεν θα έχει κανένα αποτέλεσμα.

Αν μετά το TAB και την εμφάνιση των πιθανών λέξεων προς συμπλήρωση, ο χρήστης πληκτρολογήσει έναν χαρακτήρα που δεν εμπίπτει στις παραπάνω κατηγορίες, τότε οι λέξεις προς συμπλήρωση αγνοούνται και η πληκτρολόγηση συνεχίζει κανονικά από το σημείο που ήταν (ουσιαστικά αντιγράφοντας την τελευταία γραμμή ως είχε).

Η γραμμή συμπλήρωσης θα πρέπει να εκτυπώνεται στο ρεύμα λαθών (`stderr`), ώστε να μπορεί να διαχωριστεί από την έξοδο του προγράμματος.

Command-line Options. Το πρόγραμμα θα καλείται ως εξής, από τη γραμμή εντολών:

```
./typing-assistant [-d PATH]
```

1. Η επιλογή '`-d PATH`' καθορίζει την τοποθεσία του λεξικού, το οποίο διαβάζεται από τον δίσκο. Σε περίπτωση που το όρισμα αυτό παραλείπεται, η default τοποθεσία του λεξικού είναι η `"$HOME/.dict"`. Τα περιεχόμενα του λεξικού έχουν την ακόλουθη μορφή (μία λέξη ανά γραμμή), και δεν είναι απαραίτητα ταξινομημένα:

```
about
above
abort
aborted
abolish
...
```

Το πρόγραμμά σας θα πρέπει κατά την εκκίνησή του να διαβάζει αυτό το λεξικό και να το φορτώνει στη μνήμη σε μορφή trie. Επίσης, κατά την έξοδο του προγράμματος, θα πρέπει τα περιεχόμενα του λεξικού στον δίσκο να ανανεώνονται ώστε να προστίθενται τυχόν νέες λέξεις που έχει εισάγει ο χρήστης και δεν περιέχονταν στο λεξικό αρχικά. Οι συχνότητες εμφάνισης των λέξεων, οι οποίες αλλάζουν δυναμικά κατά την εκτέλεση, δεν θα αποθηκεύονται.

Διαδικαστικά:
- Το πρόγραμμά σας θα πρέπει να τρέχει στα μηχανήματα Linux/Unix της σχολής.
- Υπεύθυνοι για την άσκηση αυτή (ερωτήσεις, αξιολόγηση, βαθμολόγηση, κτλ) είναι όλοι οι βοηθοί. Ονόματα και email θα βρείτε στην ιστοσελίδα του μαθήματος.
- Για επιπρόσθετες ανακοινώσεις, παρακολουθείτε το forum του μαθήματος στο piazza.com για να δείτε ερωτήσεις/απαντήσεις/διευκρινήσεις που δίνονται σχετικά με την άσκηση. Η παρακολούθηση του φόρουμ στο piazza.com είναι *υποχρεωτική*.

Τι πρέπει να παραδοθεί:
1. Όλη η δουλειά σας σε ένα tar-file που να περιέχει όλα τα source files, header files, Makefile. **Προσοχη:** φροντίστε να τροποποιήσετε τα δικαιώματα αυτού του αρχείου πριν την υποβολή, π.χ. με την εντολή `chmod 755 OnomaEponymoProject1.tar` (περισσότερα στη σελίδα του μαθήματος).
2. Μια σύντομη περιγραφή (1–2 σελίδες) για τις επιλογές που κάνατε στο σχεδιασμό της άσκησης, σε μορφή PDF.
3. Οποιαδήποτε πηγή πληροφορίας, συμπεριλαμβανομένου και κώδικα που μπορεί να βρήκατε στο Διαδίκτυο θα πρέπει να αναφερθεί και στον πηγαίο κώδικά σας αλλά και στην παραπάνω αναφορά.

Τι θα βαθμολογηθεί:
1. Η συμμόρφωση του κώδικά σας με τις προδιαγραφές της άσκησης.
2. Η οργάνωση και η αναγνωσιμότητα (μαζί με την ύπαρξη σχολίων) του κώδικα.
3. Η χρήση Makefile και η κομματιαστή σύμβολο-μετάφραση (separate compilation).
4. Η αναφορά που θα γράψετε και θα υποβάλετε μαζί με τον πηγαίο κώδικα σε μορφή PDF.

Άλλες σημαντικές παρατηρήσεις:
1. Οι εργασίες είναι **ατομικές**.
2. Όποιος υποβάλλει / δείχνει κώδικα που δεν έχει γραφτεί από την ίδια/τον ίδιο **μηδενίζεται** στο μάθημα.
3. Αν και αναμένεται να συζητήσετε με φίλους και συνεργάτες το πως θα επιχειρήσετε να δώσετε λύση στο πρόβλημα, *αντιγραφή κώδικα* (οποιασδήποτε μορφής) είναι κάτι που **δεν επιτρέπεται** και δεν πρέπει να γίνει. Οποιοσδήποτε βρεθεί αναμεμειγμένος σε αντιγραφή κώδικα *απλά παίρνει μηδέν* στο μάθημα. Αυτό ισχύει για *όλους όσους εμπλέκονται*, ανεξάρτητα από το ποιος έδωσε/πήρε κλπ.
4. Το πρόγραμμά σας θα πρέπει να γραφτεί σε C (ή C++ χωρίς όμως STL extensions).

Παραδείγματα εκτέλεσης:

```
The towˆTAB
> towards towel tower town          -- έστω ότι ο χρήστης πληκτρολογεί 3
The tower
         ˆ


...
The towers of Zenith aspired aboˆTAB
> abolish abort aborted about above
The towers of Zenith aspired abovˆTAB
The towers of Zenith aspired above
                                  ˆ

The towers of Zenith aspired ABOVˆTAB
The towers of Zenith aspired ABOVE
                                  ˆ

The towers of Zenith aspired AbovˆTAB
The towers of Zenith aspired Above
                                  ˆ
```