13/1/2017

Δεύτερος Έλεγχος Εργαστηρίου

1. Κώδικας του Καίσαρα

Ο Κώδικας του Καίσαρα είναι μία από τις απλούστερες και πιο γνωστές τεχνικές κωδικοποίησης στην κρυπτογραφία. Είναι κώδικας αντικατάστασης στον οποίο κάθε γράμμα του κειμένου αντικαθίσταται από κάποιο άλλο γράμμα με **σταθερή απόσταση (κλειδί)** κάθε φορά στο αλφάβητο.

Για παράδειγμα, με μετατόπιση 3, το Α θα αντικαθιστούνταν από το Δ, το Β από το Ε, και ούτω καθεξής. Η μέθοδος πήρε το όνομά της από τον Ιούλιο Καίσαρα, ο οποίος την χρησιμοποιούσε στην προσωπική του αλληλογραφία.

Σας δίνεται το εξής κρυπτογραφημένο κειμένο: ”SX]]OH XLI UQNXKPI JSV EQPITCVU JSV SX]]OH XLI EQPITCVU JSV SX]]OH XLI JSV XLI UQNXKPI”, καθώς και ο κώδικας στο αρχείο https://jimouris.github.io/ip_lab/occurCeasar.c

2. Κρυπτογράφηση/Αποκρυπτογράφηση

Το παραπάνω κείμενο έχει κρυπτογραφηθεί με λίγο πιο σύνθετο τρόπο. Κάθε λέξη έχει δικό της κλειδί, το οποίο προκύπτει από τον αριθμό εμφανίσεων της λέξης σε ολόκληρο το κείμενο.

Για παράδειγμα:
Αν έχουμε το κείμενο ” ΖΗΧΦΗΤΡ ΧΘΦΧ ΧΘΦΧ ΖΗΧΦΗΤΡ ΧΘΦΧ ”, παρατηρούμε ότι η λέξη ”ΖΗΧΦΗΤΡ” εμφανίζεται 2 φορές, ενώ η λέξη ”ΧΘΦΧ” εμφανίζεται 3 φορές.

Εφαρμόζοντας τον κώδικα του Καίσαρα με κλειδί τις εκάστοτε εμφανίσεις έχουμε:
key=2) **Ζ**ΗΧΦΗΤΡ ⟶ **Δ**ΕΥΤΕΡΟ, (... **Δ** Ε **Ζ** ...)
key=3) **Χ**ΘΦΧ ⟶ **Τ**ΕΣΤ, (... **Τ** Υ Φ **Χ**...)

Άρα τελικά το κρυπτοκείμενο ”ΖΗΧΦΗΤΡ ΧΘΦΧ ΧΘΦΧ ΖΗΧΦΗΤΡ ΧΘΦΧ” αντιστοιχεί στη φράση ”ΔΕΥΤΕΡΟ ΤΕΣΤ”.
Προφανώς στο παραπάνω παράδειγμα από τη στιγμή που πρόκειται για αποκρυπτογράφηση, τα γράμματα πάνε **πίσω** key θέσεις.

3. Υλοποίηση

Για να μπορέσουμε να κρατήσουμε τον αριθμό εμφανίσεων της κάθε κρυπτολέξης και ταυτόχρονα να μπορούμε να υποστηρίξουμε μεταβλητό αριθμό μοναδικών κρυπτολέξεων, θα χρησιμοποιήσουμε μια λίστα. Για αυτό το σκοπό, έχει οριστεί ένας καινούριος τύπος δεδομένων `node_t` με τρία πεδία, μια συμβολοσειρά (`str`) (η κρυπτολέξη), ένας ακέραιος (`occur`) (ο αριθμός εμφανίσεων της κρυπτολέξης σε όλο το κείμενο), καθώς και ένας δείκτης σε node_t (`next`) για τον επόμενο κόμβο.

Σας δίνεται έτοιμη η `main` και οι εξής συναρτήσεις:

I. `int main(int argc, char **argv)`
   Δέχεται σαν ορίσματα το κρυπτογραφημένο κείμενο. *πχ: ./a.out SX]]OH XLI UQNXKPI*

JSV EQPITCVU JSV SX]]OH XLI EQPITCVU JSV SX]]OH XLI JSV XLI UQNXKPI

II. `void insert_start(node_t **list, char *str)`
Εισάγει στην αρχή της λίστας ένα καινούριο κόμβο με το string `str`, και αρχικοποιεί τη μεταβλητή `occur` σε 1.

III. `void print_list(node_t *list)`
Εκτυπώνει τα περιεχόμενα κάθε κόμβου στη λίστα.

Καλείστε να υλοποιήσετε τις συναρτήσεις:

I. `int in_list(node_t *list, char *str)`
Αν βρει την συμβολοσειρά `str` στη λίστα `list`, αυξάνει τον μετρητή occur σε εκείνο τον κόμβο της λίστας και επιστρέφει 1. Σε αντίθετη περίπτωση επιστρέφει 0.

II. `void decrypt_list(node_t *list)`
Για κάθε κόμβο στη λίστα, αποκρυπτογραφεί την λέξη `str` του κόμβου ανάλογα με το εκάστοτε κλειδί (`occur`), σύμφωνα με τον αλγόριθμο του Καίσαρα, και την εκτυπώνει.

III. `void free_list(node_t *list)`
Η συνάρτηση αυτή είναι υπεύθυνη να αποδεσμεύσει όλη τη μνήμη που δέσμευσε η `insert_start`.

4. Βοηθητικές Συναρτήσεις

Μπορείτε να κάνετε χρήση συναρτήσεων από το string.h, όπως οι `strlen`, `strcmp`, κλπ. Για οδηγίες χρήσης δείτε τα man pages.

5. Παράδοση

Η παράδοση της άσκησης γίνεται μέσω email στο jimouris@di.uoa.gr