ΠΛΥ410 – ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΥΣΤΗΜΑΤΩΝ

Εργαστήριο #4
24/3/2015

I. ΠΑΡΑΜΕΤΡΟΙ ΣΤΗ MAIN()

Υλοποιήστε ένα πρόγραμμα-κομπιουτεράκι, το οποίο χρησιμοποιεί παραμέτρους στην `main()` και λειτουργεί ως εξής:

- Διαβάζει το πρώτο όρισμα που του δίνεται και το οποίο δηλώνει την πράξη που πρέπει να κάνει. Υποστηρίζονται: `+` (πρόσθεση), `-` (αφαίρεση), `*` (πολλαπλασιασμός), `/` (διαίρεση).
- Διαβάζει όλα τα υπόλοιπα ορίσματά του και εκτελεί την δεδομένη πράξη με αυτά.

Για παράδειγμα, η εκτέλεση:

```
$ ./a.out / 10 2
```

πρέπει να τυπώσει 5, ενώ το:

```
$ ./a.out + 2 4 6 8
```

πρέπει να τυπώσει 20.

II. ΔΗΜΙΟΥΡΓΙΑ ΑΝΤΙΓΡΑΦΟΥ ΣΥΜΒΟΛΟΣΕΙΡΑΣ

Υλοποιήστε μία συνάστηση `newstr()` με το παρακάτω πρωτότυπο:

```
char *newstr(char *str);
```

Η συνάρτηση αυτή θα πρέπει να δημιουργεί μία νέα συμβολοσειρά η οποία θα είναι πανομοιότυπο αντίτυπο της συμβολοσειράς που δίνεται ως παράμετρος. Συγκεκριμένα, θα πρέπει:

1. Να βρίσκει το μήκος της συμβολοσειράς `str`.
2. Να δεσμεύει, με χρήστη της `malloc()`, χώρο ίσο με το χώρο που καταλαμβάνει η `str`.
3. Να αντιγράφει στο νέο χώρο, τη συμβολοσειρά `str`.

Για τα 1 και 3 μπορείτε να χρησιμοποιήσετε τις συναρτήσεις για συμβολοσειρές που παρέχει το `<string.h>`.

III. ΚΡΥΠΤΟΓΡΑΦΗΣΗ ΚΕΙΜΕΝΟΥ

Η "κωδικοποίηση του Καίσαρα" είναι μία από τις πιο γνωστές τεχνικές κωδικοποίησης, όπου κάθε γράμμα του κειμένου αντικαθίσταται από κάποιο άλλο γράμμα με σταθερή απόσταση κάθε φορά στο αλφάβητο. Για παράδειγμα, με μετατόπιση 3, το A αντικαθίσταται από το D, το B από το E, κλπ, το X από το A, το Y από το B και το Z από το C. Θα πρέπει να σχεδιάσετε μία συνάρτηση η οποία δέχεται ως παραμέτρους μία συμβολοσειρά και μια σταθερά και κωδικοποιεί τη συμβολοσειρά με βάση τον κώδικα του Καίσαρα. Η συνάρτηση θα είναι η

```
char *encrypt(char *str, int dist);
```

η οποία δημιουργεί (χρησιμοποιώντας `malloc()`) μία *νέα συμβολοσειρά* όπου αποθηκεύεται η κωδικοποιημένη μορφή της παραμέτρου `str`. ΠΡΟΣΟΧΗ: Πρέπει να τροποποιούνται μόνο όσοι χαρακτήρες είναι γράμματα, κεφαλαία ή μικρά (μπορείτε να τα βρείτε κάνοντας χρήση των συναρτήσεων του `ctype.h`). Αντίστοιχα, υλοποιείστε μία συνάρτηση

```
void decrypt(char *str, int dist);
```

η οποία κάνει αποκωδικοποίηση επί τόπου της συμβολοσειράς `str` που δίνεται ως παράμετρο (δηλ. κατά την επιστροφή η `str` περιέχει το αποτέλεσμα της αποκωδικοποίησης).

Το πρόγραμμά σας θα πρέπει όταν εκτελείται να δέχεται 3 παραμέτρους: το είδος της λειτουργίας που θα κάνει (encrypt/decrypt), την κωδική απόσταση (μεταξύ 1 και 25) και τη συμβολοσειρά, π.χ.

```
./a.out encrypt 4 [This.String.is.not.encrypted]
```

το οποίο θα εκτυπώνει

```
[Xlmw.Wxvmrk.mw.rsx.irgvctxih]
```

Επομένως, θα πρέπει να γίνει χρήση ορισμάτων στην `main()` (δηλ. τα argc, argv).

IV. DEBUGGING ME GDB

Το σημερινό εργαστήριο περιλαμβάνει και εξάσκηση με τον gdb, ένα απαραίτητο εργαλείο για την αποσφαλμάτωση των προγραμμάτων σας.

V. ΕΠΙΠΛΕΟΝ ΕΞΑΣΚΗΣΗ

Το γνωστό *τρίγωνο του Pascal* είναι μια τριγωνική διάταξη των δυωνυμικών συντελεστών. Για παράδειγμα, για $n = 5$ έχει ως εξής:

```
1
1   1
1   2   1
1   3   3   1
1   4   6   4   1
```

Πρόκειται για έναν κάτω τριγωνικό πίνακα n γραμμών όπου τα στοιχεία της πρώτης στήλης (στήλη 0) και της διαγωνίου είναι ίσα με 1, ενώ οποιοδήποτε άλλο στοιχείο στη γραμμή i και στη στήλη j ισούται με το άθροισμα του ακριβώς από πάνω στοιχείου (δηλ. στη γραμμή $i-1$, στήλη j) και του από πάνω και αριστερά (δηλ. στη γραμμή $i-1$, στήλη $j-1$).

Ζητείται να φτιάξετε συνάρτηση `int **pascal(int n);` η οποία θα κατασκευάζει *δυναμικά* το τρίγωνο του Pascal με n γραμμές και θα το επιστρέφει. Κάθε γραμμή του πίνακα θα πρέπει να έχει χώρο για ακριβώς όσα στοιχεία χρειάζονται (δηλ. η γραμμή i θα πρέπει να έχει $i+1$ στοιχεία).

Η `main()` θα περιμένει ως μοναδικό όρισμα (... argc, argv) το n, θα καλεί την `pascal()` για να φτιαχτεί το τρίγωνο, θα το τυπώνει στην οθόνη και τέλος θα απελευθερώνει την μνήμη που δεσμεύτηκε.