ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (2013-14)

Άσκηση 2

Πολύ συχνά, στον προγραμματισμό χρειάζεται ένα πρόγραμμα να ελέγξει αν η είσοδός του είναι σύμφωνη με κάποιους συντακτικούς κανόνες γνωστούς εκ των προτέρων. Ο έλεγχος αυτός ονομάζεται *συντακτική ανάλυση* (parsing). Ένα κλασικό παράδειγμα είναι οι μεταγλωττιστές, οι οποίοι για να μεταφράσουν ένα πρόγραμμα σε γλώσσα assembly, πρέπει να ελέγξουν αν το πρόγραμμα είναι σωστό, με βάση τους συντακτικούς κανόνες της πηγαίας γλώσσας, και μόνο τότε θα γίνει η μεταγλώττιση. Σε αντίθετη περίπτωση, δίνουν στην έξοδο τα συντακτικά λάθη του προγράμματος, όπως ίσως γνωρίζετε πολύ καλά.

Στην άσκηση αυτή θα πρέπει να γράψετε ένα πρόγραμμα C (έστω ότι το πηγαίο αρχείο του ονομάζεται `parse.c`), το οποίο να φέρει σε πέρας μία απλή συντακτική ανάλυση της εισόδου, σύμφωνα με κάποιους κανόνες που θα περιγραφούν στη συνέχεια. Συγκεκριμένα, το πρόγραμμά σας να διαβάζει γραμμές από την πρότυπη είσοδο (χαρακτήρα-χαρακτήρα), μέχρι το τέλος της εισόδου, και να εκτυπώνει στην έξοδο, για κάθε γραμμή, αν είναι συντακτικά σωστή ή όχι. Αποδεκτές συντακτικά γραμμές είναι εκείνες που περιέχουν τα εξής:

- Τίποτα (κενή γραμμή).
- Ένας ακέραιος αριθμός (χωρίς πρόσημο), π.χ. `327`.
- Ακέραιοι αριθμοί (χωρίς πρόσημο), χωρισμένοι με κόμματα, π.χ. `12,0,857,6641,3`.
- Το κενό σύνολο, δηλαδή `{}`.
- Ένα σύνολο από ακεραίους, όπως οι παραπάνω, π.χ. `{115,89,3478,89}`.[1]
- Ένα σύνολο από ακεραίους και σύνολα (ακεραίων και συνόλων ή κενά), π.χ. `{14,{},327,0,{22,{14,8},839}}`
- Μία ακολουθία από τα προηγούμενα, χωρισμένα με κόμματα, π.χ. `115,{{77},{},215},{},10,{{{{{427}}}}},187,925678926,{}`

Μία ενδεικτική εκτέλεση του προγράμματος είναι η εξής:[2]

```
$ ./parse
327
OK
12,0,857,6641,3
OK
24.2
Error!
{}
OK
{115,89,3478,89}
OK
{125,32
```

[1] Θεωρούμε ότι για τα σύνολά μας δεν ισχύει η απαίτηση να μην περιέχουν κάποιο στοιχείο περισσότερες από μία φορά. Ουσιαστικά, πρόκειται για *πολυ-σύνολα* (multisets).

[2] Τα `OK` και `Error!` είναι οι απαντήσεις του προγράμματος στις εισόδους του χρήστη που έχουν δοθεί αμέσως πριν.

```
Error!
247,0}
Error!
{14,{},327,0,{22,{14,8},839}}
OK
253,{},
Error!

OK
115,{{77},{},215},{},10,{{{{{427}}}}},187,925678926,{}
OK
115,{{77},{},215},{},10,{{{{{427}}}}},187,925678926,{}
Error!
^D
$
```

Ή, εναλλακτικά, για να μπορείτε εύκολα να ελέγξετε την ορθότητα των αποτελεσμάτων του προγράμματός σας:[3]

```
$ cat parseinp1.txt
327
12,0,857,6641,3
24.2
{}
{115,89,3478,89}
{125,32
247,0}
{14,{},327,0,{22,{14,8},839}}
253,{},

115,{{77},{},215},{},10,{{{{{427}}}}},187,925678926,{}
115,{{77},{},215},{},10,{{{{{427}}}}},187,925678926,{}
$
$ ./parse < parseinp1.txt
OK
OK
Error!
OK
OK
Error!
Error!
OK
Error!
OK
OK
Error!
$
```

[3] Το αρχείο `parseinp1.txt` βρίσκεται στο `http://www.di.uoa.gr/~ip/hwfiles/parse/parseinp1.txt`

Για το άριστα της βαθμολογίας της άσκησης αρκεί να έχει καλυφθεί, με σωστό τρόπο, στην υλοποίηση που θα παραδοθεί, ό,τι έχει περιγραφεί σαν προδιαγραφές μέχρι στιγμής. Όμως, προαιρετικά, μπορεί να παραδοθεί η άσκηση και με κάποιες από τις επεκτάσεις που περιγράφονται στη συνέχεια, που μπορούν να οδηγήσουν σε αύξηση της βαθμολογίας της πέρα από το άριστα (bonus).

Επέκταση Α (bonus 5%):

Οι ακέραιοι αριθμοί να είναι δυνατόν να έχουν και πρόσημο (`+` ή `-`).

Επέκταση Β (bonus 5%):

Στις περιπτώσεις γραμμών στην είσοδο που είναι συντακτικά σωστές, το πρόγραμμα να μην εκτυπώνει απλώς το `OK`, αλλά κάποιες πληροφορίες για τη γραμμή εισόδου. Συγκεκριμένα, να εκτυπώνει:

- Το μέγιστο βάθος συνόλων σε κάθε γραμμή εισόδου. Αν δεν υπάρχουν σύνολα στην είσοδο, το βάθος αυτό ισούται με 0. Αν υπάρχουν μόνο σύνολα ακεραίων, το βάθος είναι 1. Αν υπάρχει έστω και ένα σύνολο που περιέχει ως στοιχείο κάποιο σύνολο ακεραίων, το βάθος είναι τουλάχιστον 2, κοκ.
- Το πλήθος των συνόλων που εμφανίζονται σε κάθε γραμμή εισόδου, ανεξαρτήτως βάθους, εμφωλευμένων ή όχι.
- Το πλήθος των ακεραίων που εμφανίζονται σε κάθε γραμμή εισόδου.

Επέκταση Γ (bonus 10%):

Σε περίπτωση λάθους εισόδου, να εκτυπώνεται, αντί για το `Error!`, ακριβές διαγνωστικό μήνυμα για το είδος του λάθους, όπως:

```
Invalid character
Extra closing brace(s)
Missing closing brace(s)
Digit expected
Unexpected end of line
```

Τα προηγούμενα είδη λαθών πρέπει να καλυφθούν οπωσδήποτε για να θεωρηθεί η υλοποίηση της επέκτασης πλήρης. Μπορείτε όμως, αν επιθυμείτε, να τα εξειδικεύσετε περισσότερο, προσθέτοντας και περισσότερα είδη λαθών. Η ακριβής σημασία των μηνυμάτων θα γίνει περισσότερο σαφής από την ενδεικτική εκτέλεση που θα ακολουθήσει.

Επέκταση Δ (bonus 10%):

Το πρόγραμμά σας να δέχεται την ύπαρξη "λευκών διαστημάτων" (white spaces), δηλαδή κενών χαρακτήρων ή χαρακτήρων στηλογνώμονα (πλήκτρο tab), όχι όμως στο εσωτερικό των αριθμών (τα ψηφία τους πρέπει να είναι συνεχόμενα), ή μεταξύ του προσήμου ενός αριθμού και του πρώτου ψηφίου του.

Μία ενδεικτική εκτέλεση του προγράμματος, με υλοποιημένες όλες τις προηγούμενες επεκτάσεις, φαίνεται στη συνέχεια.[4]

```
$ ./parse < parseinp1.txt
depth = 0  sets = 0  integers = 1
depth = 0  sets = 0  integers = 5
Invalid character
```

[4] Το αρχείο `parseinp2.txt` βρίσκεται στο `http://www.di.uoa.gr/~ip/hwfiles/parse/parseinp2.txt`

```
depth = 1  sets = 1  integers = 0
depth = 1  sets = 1  integers = 4
Missing closing brace(s)
Extra closing brace(s)
depth = 3  sets = 4  integers = 7
Unexpected end of line
depth = 0  sets = 0  integers = 0
depth = 5  sets = 10  integers = 7
Extra closing brace(s)
$
$ cat parseinp2.txt
347, 1806
{       }
{{23,-56} , {734} , {  11  }}
  +82   ,               {   }
{- 77}
{34 56,    88}
$
$ ./parse < parseinp2.txt
depth = 0  sets = 0  integers = 2
depth = 1  sets = 1  integers = 0
depth = 2  sets = 4  integers = 4
depth = 1  sets = 1  integers = 1
Digit expected
Invalid character
$
```

Η παράδοση της άσκησης αυτής συνίσταται στην υποβολή του πηγαίου αρχείου `parse.c` μέσω του συστήματος υποβολής στη διεύθυνση `http://hwadm.di.uoa.gr`. Σε περίπτωση που στην άσκηση που παραδίδεται έχουν υλοποιηθεί κάποιες από τις επεκτάσεις Α, Β, Γ ή Δ, αυτό θα πρέπει να αναφέρεται ρητά στην αρχή του πηγαίου αρχείου, σε σχόλιο, ώστε να τύχει η παραδοθείσα άσκηση ενδεχόμενου βαθμολογικού bonus (μέγιστο 30%).

Υπόδειξη: Ασχοληθείτε με τις προτεινόμενες επεκτάσεις μόνο αφού εξασφαλίσετε ότι έχετε υλοποιήσει σωστά τη βασική λειτουργικότητα που ζητείται για το πρόγραμμα (χωρίς τις επεκτάσεις). Επεκτάσεις σε πρόγραμμα που δίνει λάθος αποτελέσματα στο κύριο ζητούμενο της άσκησης δεν θα βαθμολογηθούν. Ενδεχομένως θα έχουν και αρνητικές επιπτώσεις. Ως προς τη βασική λειτουργικότητα, προτείνεται αρχικά να υλοποιήσετε έτσι το πρόγραμμα ώστε να αναγνωρίζει μόνο έναν ακέραιο. Στη συνέχεια, προσθέστε και την αναγνώριση ακολουθίας από ακεραίους. Στο τέλος, να συμπεριλάβετε και την αναγνώριση συνόλων.

Σημείωση: Στην άσκηση αυτή δεν χρειάζεται να χρησιμοποιηθούν πίνακες, ούτε συμβολοσειρές. Για την ακρίβεια, η χρήση πινάκων ή συμβολοσειρών θα έχει σαν αποτέλεσμα τον μηδενισμό της άσκησης. Η χρήση συναρτήσεων επιτρέπεται, αλλά δεν είναι υποχρεωτική.