# Γλώσσες Προγραμματισμού II

## Άσκηση 7:
## Συστήματα τύπων - μηχανή στοίβας

Δανάη Ευσταθίου, 10ο εξάμηνο

AM : 03115122

# Περιγραφή

Για την άσκηση αυτή ζητείται ο ορισμός ενός συστήματος τύπων για μια γλώσσα μηχανής στοίβας. Για τον σκοπό αυτό θεωρήθηκαν 3 βασικοί τύποι:

1. ο Int που αντιστοιχεί στους ακέραιους,
2. ο Bool που αντιστοιχεί στις λογικές τιμές true, false και
3. ο Unit που αντιστοιχεί στον κενό τύπο.

Αρχικά, θα πρέπει να βρεθεί ένας τρόπος αναπαράστασης των τύπων των στοιχείων της στοίβας. Για τον σκοπό αυτό, επιλέχθηκε ο τύπος λίστας για την αναπαράσταση της στοίβας. Συγκεκριμένα, ο τύπος λίστας έχει τη μορφή $List \equiv Unit + (A, List)$, δηλαδή είναι αναδρομικός τύπος, όπου $A$ μπορεί να είναι είτε $Int$ είτε $Bool$ τύπος. Η λίστα είτε είναι άδεια (περίπτωση $Unit$) είτε έχει τουλάχιστον ένα στοιχείο, όπου ο τύπος $A$ αποτελεί και την κορυφή της λίστας.

Όταν εισάγεται ένα στοιχείο στην στοίβα τύπου $B$, η αρχική μορφή της λίστας $(A, List)$ θα μετατρέπεται σε $(B, (A, List))$. Από την άλλη, όταν αφαιρείται ένα στοιχείο από την στοίβα, η αρχική μορφή της λίστας $(B, (A, List))$ θα μετατρέπεται σε $(A, List)$. Πριν την έναρξη του προγράμματος θεωρούμε ότι η στοίβα έχει τύπο $Unit$.

# Κανόνες Τύπων

Για το σύστημα τύπων αναπτύχθηκαν οι παρακάτω κανόνες τύπων, όσον αφορά την επίδραση των εντολών στον τύπο της στοίβας, όπως αυτή ορίστηκε παραπάνω. Οι κανόνες είναι είτε της μορφής:

$$x: \ (A, ListA) \ \rightarrow \ (B, ListB)$$

που σημαίνει ότι η εντολή $x$ λαμβάνει είσοδο με τύπο $(A, ListA)$ και δίνει έξοδο με τύπο $(B, ListB)$,

είτε της:

$$\frac{x,\ y}{z}$$

που σημαίνει ότι αν ισχύουν τα $x, y$ τότε ισχύει το $z$.

Για την αναπαράσταση οποιουδήποτε εκ των τριών τύπων $Int$, $Bool$, $Unit$ χρησιμοποιούνται οι χαρακτήρες του αγγλικού αλφαβήτου ($a$, $b$, $c$, ...), για την αναπαράσταση των τύπων $Int$, $Bool$ χρησιμοποιούνται οι χαρακτήρες του ελληνικού αλφαβήτου ($\alpha$, $\beta$, $\gamma$, ...), ενώ για την αναπαράσταση του τύπου λίστας χρησιμοποιούνται οι λέξεις κλειδιά $List, ListA, ListB, ...$ ώστε να υποδηλώσουν τους τύπους λιστών στοιχείων διαφορετικού τύπου.

Με αυτόν τον τρόπο, οι κανόνες διαμορφώθηκαν ως εξής:

$$n: \ List \ \rightarrow \ (Int, List)$$
$$true, \ false: \ List \ \rightarrow \ (Bool, List)$$
$$+, \ *: \ (Int, (Int, List)) \ \rightarrow \ (Int, List)$$
$$-: \ (Int, List) \ \rightarrow \ (Int, List)$$
$$/: \ (Int, (Int, List)) \ \rightarrow \ (Int, (Int, List))$$
$$<: \ (Int, (Int, List)) \ \rightarrow \ (Bool, List)$$
$$=: \ (\alpha, (\alpha, List)) \ \rightarrow \ (Bool, List)$$
$$and: \ (Bool, (Bool, List)) \ \rightarrow \ (Bool, List)$$
$$not: \ (Bool, List) \ \rightarrow \ (Bool, List)$$
$$dup: \ (\alpha, List) \ \rightarrow \ (\alpha, (\alpha, List))$$
$$pop: \ (\alpha, List) \ \rightarrow \ List$$
$$swap: \ (\alpha, (\beta, List)) \ \rightarrow \ (\beta, (\alpha, List))$$
$$swap2: \ (\alpha, (\beta, (\gamma, List))) \ \rightarrow \ (\gamma, (\beta, (\alpha, List)))$$
$$nop: \ List \ \rightarrow \ List$$

$$\frac{p_1: \ ListA \ \rightarrow \ ListB, \ p_2: \ ListA \ \rightarrow \ ListB}{cond[p_1|p_2]: \ (Bool, ListA) \ \rightarrow \ ListB}$$

$$\frac{p: \ List \ \rightarrow \ (Bool, List)}{loop[p]: \ (Bool, List) \ \rightarrow \ List}$$

$$\frac{p_1: \ ListA \ \rightarrow \ ListB, \ p_2: \ ListB \ \rightarrow \ ListC}{p_1 \ p_2: \ ListA \ \rightarrow \ ListC}$$

Καθώς για τους υπόλοιπους κανόνες είναι σχετικά προφανής η ιδέα για την διαμόρφωσή τους, θα ήταν σκόπιμο να γίνει διευκρίνιση του γιατί έγιναν οι συγκεκριμένες σχεδιαστικές επιλογές για τους κανόνες των *cond* και *loop*:

- Στον κανόνα για την *cond* τα προγράμματα $p_1$, $p_2$ παίρνουν τον ίδιο τύπο στην είσοδο και δίνουν τον ίδιο τύπο στην έξοδο, προκειμένου και ο τύπος εισόδου και εξόδου της *cond* να είναι απλά ο ίδιος με αυτόν των προγραμμάτων $p_1$, $p_2$, εξαιρουμένου του έξτρα τύπου $Bool$ στην είσοδο για τον έλεγχο.
- Η εντολή *loop* παίρνει ως όρισμα ένα πρόγραμμα $p$, το οποίο δίνει έξοδο του ίδιου τύπου με την είσοδο του *loop*. Η επιλογή αυτή έγινε προκειμένου οι πιθανές συνεχόμενες εκτελέσεις του $p$, που ισοδυναμούν με πιθανή παράθεση του $p$ με τον εαυτό του πολλαπλές φορές, να μην δημιουργούν σύγκρουση με τον κανόνα τύπων για την παράθεση. Και σε αυτή την περίπτωση ισχύει το ίδιο με την περίπτωση του *cond* για την τιμή $Bool$ στην είσοδο.

Είναι κατανοητό πως αυτός ο τρόπος αναπαράστασης του τύπου της στοίβας θα δώσει σφάλμα τύπου σε προγράμματα που θα μπορούσαν να τρέξουν χωρίς να κολλήσουν, όπως για παράδειγμα η εντολή *cond* με προγράμματα διαφορετικού τύπου στον ορισμό της, αλλά όσα προγράμματα περάσουν τον έλεγχο τύπων είναι σίγουρο ότι δεν θα κολλήσουν.