Γλώσσες Προγραμματισμού ΙΙ
Δηλωτική Σημασιολογία

Μιλτιάδης Στούρας
ΑΜ: 03116022

Ακ. Έτος 2019-2020 - Ροή Λ

Στην εργασία αυτή θα ορίσουμε μια δηλωτική σημασιολογία για την γλώσσα που ορίστηκε στην Άσκηση 7. Θα χρησιμοποιήσουμε τον συμβολισμό $[\![\text{P}]\!](s)$ για να αναφερόμαστε στην δηλωτική σημασιολογία του προγράμματος P όταν αυτό εκκινεί την εκτέλεσή του από μια στοίβα s. Η δηλωτική σημασιολογία του προγράμματος θα είναι η στοίβα με την οποία καταλήγουμε μετά την εκτέλεσή του. Ακόμα, θα ορίσουμε την σημασιολογία μόνο των προγραμμάτων που κάνουν typecheck στο σύστημα τύπων της Άσκησης 7. Αν μια εντολή δεν είναι σωστή ή δεν εκτελείται με τις σωστές προϋποθέσεις στην κορυφή της στοίβας, τότε η σημασιολογία της θα είναι το $Unit$, το οποίο υποτύπος του $Stack$ όπως αυτό ορίστηκε στην προηγούμενη εργασία.

Δηλωτική Σημασιολογία μιας stack machine γλώσσας

Προσθήκη τιμών στην στοίβα

- $[\![n]\!](\sigma) = \sigma \cdot n, \quad n \in \mathbb{N}$
- $[\![\textbf{true}]\!](\sigma) = \sigma \cdot \textbf{true}$
- $[\![\textbf{false}]\!](\sigma) = \sigma \cdot \textbf{false}$

Αριθμητικές πράξεις

- $[\![+]\!](\sigma \cdot n_1 \cdot n_2) = \sigma \cdot (n_1 + n_2), \quad n_1, n_2 \in \mathbb{N}$
- $[\![*]\!](\sigma \cdot n_1 \cdot n_2) = \sigma \cdot (n_1 * n_2), \quad n_1, n_2 \in \mathbb{N}$
- $[\![/]\!](\sigma \cdot n_1 \cdot n_2) = q \cdot r \cdot \sigma, \quad n_1, n_2 \in \mathbb{N},\ n_2 \neq 0,\ (q, r) = \text{quotRem}(n_1, n_2)$
- $[\![\text{-}]\!](\sigma \cdot n) = \sigma \cdot (-n), \quad n \in \mathbb{N}$

Λογικές πράξεις

- $[\![<]\!](\sigma \cdot n_1 \cdot n_2) = \sigma \cdot (n_1 < n_2), \quad n_1, n_2 \in \mathbb{N}$
- $[\![=]\!](\sigma \cdot n_1 \cdot n_2) = \sigma \cdot (n_1 = n_2), \quad n_1, n_2 \in \mathbb{N}$
- $[\![\textbf{and}]\!](\sigma \cdot n_1 \cdot n_2) = \sigma \cdot (n_1 \wedge n_2), \quad n_1, n_2 \in \mathbb{N}$
- $[\![\textbf{not}]\!](\sigma \cdot n) = \sigma \cdot (\neg n), \quad n \in \mathbb{N}$

Εντολές τροποποίησης της στοίβας

- $[\![\ \textbf{dup}\ ]\!]\,(\sigma \cdot u) = \sigma \cdot u \cdot u, \quad u \in \{Int, Bool\}$
- $[\![\ \textbf{pop}\ ]\!]\,(\sigma \cdot u) = \sigma, \quad u \in \{Int, Bool\}$
- $[\![\ \textbf{swap}\ ]\!]\,(\sigma \cdot u_1 \cdot u_2) = \sigma \cdot u_2 \cdot u_1, \quad (u_1, u_2) \in \{Int, Bool\}^2$
- $[\![\ \textbf{swap2}\ ]\!]\,(\sigma \cdot u_1 \cdot u_2 \cdot u_3) = \sigma \cdot u_3 \cdot u_1 \cdot u_2, \quad (u_1, u_2, u_3) \in \{Int, Bool\}^3$

Εντολές stall και ελέγχου ροής

- $[\![\ \textbf{nop}\ ]\!]\,(\sigma) = \sigma$
- $[\![\ \textbf{cond}[P_1|P_2]\ ]\!]\,(\sigma \cdot \textbf{true}) = [\![\ P_1\ ]\!]\,(\sigma)$
- $[\![\ \textbf{cond}[P_1|P_2]\ ]\!]\,(\sigma \cdot \textbf{false}) = [\![\ P_2\ ]\!]\,(\sigma)$

Για να ορίσουμε την δηλωτική σημασιολογία της εντολής **loop** θα χρησιμοποιήσουμε την παρακάτω βοηθητική συνάρτηση:

$$\text{funcLoop}(f,\ \sigma \cdot v\ ;\ \text{P}) = \begin{cases} f([\![\ \text{P}\ ]\!]\,(\sigma)), & \text{if } v = \textbf{true}, \\ \sigma, & \text{otherwise} \end{cases}$$

Χρησιμοποιώντας την παραπάνω συνάρτηση το **loop** μπορεί να οριστεί ως εξής:

$$[\![\ \textbf{loop}[\text{P}]\ ]\!]\,(\sigma \cdot v) = (\text{fix funcLoop})(\sigma \cdot v; \text{P})$$

Τέλος, οι εντολές εκτελούνται από τα αριστερά προς τα δεξιά, δηλαδή:

$$[\![\ P_1\ P_2\ ]\!]\,(\sigma) = [\![\ P_2\ ]\!]\,([\![\ P_1\ ]\!]\,(\sigma))$$