

Prevodioci i Interpretatori - Septembar 2013.

praktični deo

1. Konstruisati MDKA za prepoznavanje jezika nad azbukom $\Sigma = \{a, b\}$ čija svaka reč ne sme da sadrži faktor *abab*, i zatim napisati *C* program koji proverava da li uneta niska pripada ovom jeziku.
2. Implementirati interpretator koji sračunava vrednosti izraza u kojima se javljaju prirodni brojevi i liste prirodnih brojeva. Svaka naredba iterpretatora se završava znakom ;.

- (a) Nad brojevima je moguće vršiti sve četiri aritmetičke operacije, pri čemu je deljenje celobrojno, a oduzimanje većeg broja od manjeg daje 0. Funkcija *Suc* računa sledbenik broja. Npr.

<code>Suc(5);</code>	6
<code>1 + 2*3;</code>	7
<code>3 - 1;</code>	2
<code>3 - 2*2;</code>	0

- (b) Liste se zapisuju kao $[a_1, \dots, a_n]$. Operator *#* dopisuje element na početak liste, a operator *@* nadovezuje dve liste. Npr.

<code>[1, 2, 3];</code>	<code>[1, 2, 3]</code>
<code>1 # [];</code>	<code>[1]</code>
<code>[1, 2] @ [1+2, 4];</code>	<code>[1, 2, 3, 4]</code>

- (c) Dopusšteno je i korišćenje *let* vezivanja. Promenljive su tada strogo lokalne, tj. važe samo u *let* izrazu. Npr.

<code>let x = 3 in 2*x;</code>	6
<code>let l1 = [1, 2]; l2 = [3, 4] in l1 @ l2;</code>	<code>[1, 2, 3, 4]</code>

- (d) Nad listama su definisane funkcije *hd* koja izdvaja glavu (prvi element) liste, *tl* koja izdvaja rep liste, operator *!* koji izdvaja n-ti element (brojanje počinje od 1), *rev* koja obrće listu i *sort* koja sortira listu rastuće. Npr.

<code>hd([1, 2, 3]);</code>	1
<code>tl([1, 2, 3]);</code>	<code>[2, 3]</code>
<code>[6, 7, 8, 9] ! 2;</code>	7
<code>rev([1, 2, 3]);</code>	<code>[3, 2 1]</code>
<code>sort([7, 9, 8, 6]);</code>	<code>[6, 7, 8, 9]</code>

- (e) Moguće je definisanje funkcija primitivnom rekurzijom (nad brojevima i nad listama) i njihovo korišćenje u izrazima. Npr.

<code>primrec sum where</code>	
<code>sum(0) = 0;</code>	
<code> sum(Suc(n)) = (Suc n) + sum n;</code>	
<code>sum(5);</code>	15
<code>primrec power where</code>	
<code>power(x, 0) = 1;</code>	
<code> power(x, (Suc n)) = x * power(x, n);</code>	
<code>power(2, 3);</code>	8
<code>primrec length where</code>	
<code>length([]) = 0;</code>	
<code> length(h#t) = 1 + length(t);</code>	
<code>length([3, 7, 2]);</code>	3