

Syntaxgerichtete Übersetzung einer einfachen Programmiersprache in Drei-Adress-Code als Zwischencode

Hinweise:

Die Hilfsfunktionen **newL** und **newT** dienen der Erzeugung neuer symbolischer Marken bzw. neuer Namen (temporary names).

Das Attribut **next** einer Anweisung S erhält als Wert eine Marke L, die unmittelbar hinter die Befehle von S gesetzt wird.

Das Attribut **addr** eines Ausdrucks E erhält als Wert denjenigen Namen, unter dem der Wert von E gespeichert wird.

Produktion	Semantische Regeln
$P \rightarrow S$	$S.next = newL(); P.code = S.code \parallel \text{Label } S.next : \text{Stop}$
$S \rightarrow id = E;$	$S.code = E.code \parallel id.lexeme = E.addr$
$E \rightarrow E_1 + E_2$	$E.addr = newT(); E.code = E_1.code \parallel E_2.code \parallel$ $E.addr = E_1.addr + E_2.addr$
$E \rightarrow - E_1$	$E.addr = newT(); E.code = E_1.code \parallel$ $E.addr = \text{minus } E_1.addr$
$E \rightarrow (E_1)$	$E.addr = E_1.addr; E.code = E_1.code$
$E \rightarrow id$	$E.addr = id.lexeme; E.code = ' '$
$S \rightarrow S_1 S_2$	$S_1.next = newL(); S_2.next = S.next;$ $S.code = S_1.code \parallel \text{Label } S_1.next : S_2.code$
$S \rightarrow \text{if } (B) S_1$	$S_1.next = S.next;$ $S.code = B.code \parallel \text{ifFalse } B.addr \text{ goto } S.next \parallel S_1.code$
$S \rightarrow \text{if } (B) S_1 \text{ else } S_2$	$S_1.next = S_2.next = S.next; Lf = newL();$ $S.code = B.code \parallel \text{ifFalse } B.addr \text{ goto } Lf \parallel S_1.code \parallel$ $\text{goto } S.next \parallel \text{Label } Lf : S_2.code$
$S \rightarrow \text{while } (B) S_1$	$Lb = newL(); S_1.next = Lb;$ $S.code = \text{Label } Lb : B.code \parallel \text{ifFalse } B.addr \text{ goto } S.next$ $\parallel S_1.code \parallel \text{goto } Lb$