

# Génération de code intermédiaire (code 3 adresses)

On s'intéresse au langage **While**. Nous donnons ci-dessous une grammaire abstraite du langage qui décrit des arbres abstraits. Le problème auquel on s'intéresse est le suivant :

*A partir d'une arbre abstrait, générer du code "3 adresses"*

## 1 Syntaxe abstraite

### 1.1 Catégories syntaxiques

<i>Métavariabes</i>	<i>Catégories</i>	<i>Commentaires</i>
a	: AExp	expressions arithmétiques
b	: BExp	expressions booléennes
S	: Inst	instructions

**Instructions**  $S \rightarrow name := a \mid skip \mid S; S \mid \text{If } b \text{ then } S \text{ else } S \mid \text{while } b \text{ do } S \text{ od}$   
 $name \rightarrow x \mid tab[i] \mid tab[i, j]$

**Expressions arithmétiques**  $a \rightarrow n \mid x \mid a + a \mid tab[i] \mid tab[i, j]$

**Expressions booléennes**  $b \rightarrow \text{True} \mid \text{False} \mid \neg b \mid b \wedge b \mid a = a \mid a < a$

## 2 Code 3 adresses

La syntaxe du code trois adresses est la suivante.

- Soit **Nom** l'espace des noms qui peuvent soit apparaître dans le programme soit être créés par la fonction `nouveauTemp` :  $\rightarrow \text{Nom}$
- Soit  $\mathbb{N}$  l'ensemble des entiers naturels.  
Il y a une fonction partielle `val` :  $\text{AExp} \rightarrow (\text{Nom} \cup \mathbb{N})$ .
- Soit **Label** l'ensemble des étiquettes. Celles-ci sont créées par la fonction `nouvelleEtiqu` :  $\rightarrow \text{Label}$

Dans ce qui suit  $x \in \text{Nom}$  et  $y, z \in (\text{Nom} \cup \mathbb{N})$ ,  $l \in \text{Label}$ .

## 2.1 Catégorie Syntaxique

<i>Métavariabiles</i>	<i>Catégories</i>	<i>Commentaires</i>
C	: Code	code 3 adresses
op	: Op={+, -, *}	
oprel	: Oprel={<, >, =, ≤, ≥}	

**Grammaire**  $C \rightarrow x := y \text{ op } z \mid x := y \mid \text{if } y \text{ oprel } x \text{ goto } l \mid \text{goto } l \mid x := y[z] \mid y[z] := x$

## 2.2 Principe de génération de code

On va générer du code 3 adresses. Nous définissons trois fonctions de génération de code

$\text{GenCodeAExp} : \text{AExp} \rightarrow \text{Code}^* \times (\text{Nom} \cup \mathbb{N})$   
 $\text{GenCodeBExp} : \text{BExp} \times \mathcal{L}\text{abel} \times \mathcal{L}\text{abel} \rightarrow \text{Code}^*$   
 $\text{GenCodeInst} : \text{Inst} \rightarrow \text{Code}^*$

où  $\text{Code}^*$  est l'ensemble des séquences de code 3 adresses, le séparateur de séquence étant  $\parallel$ .

## 3 Génération de code pour les expressions arithmétiques

On considère des tableaux à une dimension et N éléments allant de 0 à N-1 et des tableaux à deux dimensions à NxM éléments. La taille d'un élément est T. On peut généraliser sans problème. Si le tableau à deux dimensions est rangé en colonne (resp. en ligne), l'accès à un élément est

$$\text{Tab}[i,j] := N * T * j + i * T \quad (\text{resp } M * T * i + j * T)$$

$\text{GenCodeAExp}(x)$	=	$(\varepsilon, x)$
$\text{GenCodeAExp}(n)$	=	$(\varepsilon, n)$
$\text{GenCodeAExp}(\text{tab}[i])$	= Soit	t1=nouveauTemp, t2=nouveauTemp dans $(t_1 := T * i \parallel$ $t_2 := \text{tab}[t_1, t_2])$
$\text{GenCodeAExp}(\text{tab}[i, j])$	= Soit	t1=nouveauTemp, t2=nouveauTemp t3=nouveauTemp, t4=nouveauTemp t5=nouveauTemp dans $(t_1 := T * i \parallel$ $t_2 := N \times T \parallel$ $t_3 := t_2 \times j \parallel$ $t_4 := t_1 + t_3 \parallel$ $t_5 := \text{tab}[t_4, t_5])$
$\text{GenCodeAExp}(a_1 + a_2)$	= Soit	$(C_1, t_1) = \text{GenCodeAExp}(a_1),$ $(C_2, t_2) = \text{GenCodeAExp}(a_2),$ t=nouveauTemp dans $(C_1 \parallel C_2 \parallel t := t_1 + t_2, t)$

## 4 Génération de code pour les expressions booléennes

$\text{GenCodeBExp}(a_1 < a_2, \text{lvrai}, \text{lfaux})$	=	Soit $(C_1, t_1) = \text{GenCodeAExp}(a_1),$ $(C_2, t_2) = \text{GenCodeAExp}(a_2),$ dans $C_1 \parallel C_2 \parallel \text{if } t_1 < t_2 \text{ goto lvrai} \parallel \text{goto lfaux}$
$\text{GenCodeBExp}(b_1 \wedge b_2, \text{lvrai}, \text{lfaux})$	=	Soit $l = \text{nouvelleEtiq}()$ dans $\text{GenCodeBExp}(b_1, l, \text{lfaux}) \parallel$ $l: \parallel$ $\text{GenCodeBExp}(b_2, \text{lvrai}, \text{lfaux})$
$\text{GenCodeBExp}(\neg b, \text{lvrai}, \text{lfaux})$	=	$\text{GenCodeBExp}(b, \text{lfaux}, \text{lvrai})$

## 5 Génération de code pour les instructions de contrôle

A chaque nœud de l'arbre abstrait, on associe du code de la manière suivante :

$\text{GenCodeInst}(x := a)$	=	Soit $(C, t) = \text{GenCodeAExp}(a)$ dans $C \parallel x := t$
$\text{GenCodeInst}(\text{tab}[i] := a)$	=	Soit $t1 = \text{nouveauTemp},$ $(C, t) = \text{GenCodeAExp}(a)$ dans $(t_1 := N * i \parallel$ $C \parallel \text{tab}[t_1] := t)$
$\text{GenCodeInst}(S_1 ; S_2)$	=	Soit $C_1 = \text{GenCodeInst}(S_1),$ $C_2 = \text{GenCodeInst}(S_2)$ dans $C_1 \parallel C_2$
$\text{GenCodeInst}(\text{while } b \text{ do } S \text{ od})$	=	Soit $l\text{debut} = \text{nouvelleEtiq}(),$ $lvrai = \text{nouvelleEtiq}(),$ $lfaux = \text{nouvelleEtiq}()$ dans $l\text{debut}: \parallel$ $\text{GenCodeBExp}(b, lvrai, lfaux) \parallel$ $lvrai: \parallel$ $\text{GenCodeInst}(S) \parallel$ $\text{goto } l\text{debut} \parallel$ $lfaux:$
$\text{GenCodeInst}(\text{if } b \text{ then } S_1 \text{ else } S_2)$	=	Soit $lsuivant = \text{nouvelleEtiq}(),$ $lvrai = \text{nouvelleEtiq}(),$ $lfaux = \text{nouvelleEtiq}()$ dans $\text{GenCodeBExp}(b, lvrai, lfaux) \parallel$ $lvrai:$ $\text{GenCodeInst}(S_1) \parallel$ $\text{goto } lsuivant \parallel$ $lfaux: \parallel$ $\text{GenCodeInst}(S_2) \parallel$ $lsuivant:$