# Compilateur ALGO ➜ RAM

## Introduction

Compilateur d'un langage de description d'algorithmes vers un code executable en machine RAM.
La machine utilisee pour tester les programme est la suivante: https://zanotti.univ-tln.fr/RAM/newram.php

## Specificites du programme

Les declarations globales (var. globales et fonctions) se font avant l'execution du MAIN( ), il y a donc un STOP juste apres la derniere instruction du MAIN( ).

### Registre:

| | |
|---|---|
| TEMP | valeur temporaire |
| PILE | adresse de la pile |
| GLOBAL | adresse pour acceder aux variables globales |
| LOCAL | adresse pour acceder aux variables locales |
| ADR_AFFECT | adresse pour faire une affectation |
| RETOUR_FCT | Ligne a laquelle JUMP pour sortir d'une fonction |
| VALEUR_RETOUR | valeur que la fonction retourne (-999) par defaut |
| | |

### GLOBAL / LOCAL:

Les donnees globales et locales sont stockees dans la PILE a partir de l'adresse 9.
Les donnees locales demarent juste apres la pile globale donc si global est vide elles commencent a 9.
A chaque recherche d'ID (pour une fonction variable ou liste) on cherche d'abord si il y en a un local, cela permet d'avoir 2 ID de meme nom dans des contextes differents avec une priorite pour l'ID local en cas d'acces.

### Expression:

Une Expression va toujours placer son resultat dans l'accumulateur.
Elle utilisera au plus TEMP et la PILE sans restaurer TEMP.

### Liste:

Une Liste va empiler chacune des valeurs la contenant.
Declarer une liste va allouer en memoire sa taille.

### Pointeurs

Un pointeur (IDL pour id liste) ne se declare pas, il sert principalement a passer une adresse de liste.
On peut acceder et affecter a l'endroit avec ID[Expression]
Il n'y a aucune securite pour les pointeurs, ils peuvent acceder a n'importe quelle case de memoire.

### Affectation:

Une Affectation stocke dans ADR_AFFECT puis evalue l'expression a placer dans ADR_AFFECT
Il faut donc s'assurer que ADR_AFFECT ne change pas en evaluant l'expression (voir Fonction & appel).

### Appel de fonction

Un appel de fonction empile la ligne a laquelle retourner, le RETOUR_FCT, la VALEUR_RETOUR, l'ADR_AFFECT et le LOCAL actuel
Il empile ensuite la liste de parametres puis JUMP a l'initialisation de la fonction.

### Fonction:

Une declaration de fonction place a son adresse la ligne du debut de l'initialisation puis JUMP apres toutes les instructions de la fonction.

#### Initialisation

Change LOCAL pour corespondre au contexte de la fonction.
Change RETOUR_FCT pour qu'un RETURN JUMP bien a la fin de la fonction en cours.

#### Declarations si il y en a

#### Instructions

#### Restauration

Depile plusieurs fois pour remettre les valeurs du registre dans le meme etat qu'avant l'appel de fonction.
Charge VALEUR_RETOUR puis JUMP a la ligne apres l'appel de fonction.

# Exemples

### exemple1.algo:

- declaration & affectation
- expressions arithmetique avec des variables
- lecture & ecriture

### exemple2.algo:

- variables globales
- priorite sur variable locale en cas de meme identificateur

### exemple3.algo:

- Liste et affectation de toute la liste
- priorite sur la liste locale en cas de meme identificateur
- Element d'une liste
- boucle tant que

### exemple4.algo

- Fonction & recursion (besoin d'un prototype)
- condition SI
- Appel de fonction
- pointeur & modification a l'adresse pointee

### fibonacci.algo

- Retour de fonction
- condition SI SINON

### tri_rapide & triselect.algo

Ces 2 fichiers se compilent, l'AST et la TS sont generes cependant le code RAM genere est trop lourd pour etre lu par la machine RAM utilisee pour executer les programmes.

# Arbre de syntaxe abstrait

L'AST possede une valeur qui est une union de types en fonction du noeud ou de la feuille de l'AST.

# Semantique & Table de symboles

Determine la longueur du code, genere la table de symboles et filtre des erreurs semantiques
La table de symboles contiendra toujours au moins GLOBAL et MAIN comme contexte.
Lors de la creation d'une fonction, le 1er symbole du contexte sert a verifier le nombre de parametres de la fonction.