|  |
| --- |
| ForkInterpreter – Commande « let » |

# Enoncé

Let Ide in Com end declaration de variable

Let Ide be Ide in Com end aliasing

# Aide à la comprehension

1. **Ide** représente le nom d’une variable. Il ne peut pas y avoir d’assignation ici, c’est juste le nom qui est passé ici. Par conséquent, la variable doit impérativement être initialisée avant.
2. En fait, pour faire simple, la valeur de cette variable est temporaire dans la commande qui suit. C’est-à-dire que si je fais une assignation (par exemple, a=6) ici, elle perdra sa valeur à la fin de la commande let.

# Exemple simple

|  |  |
| --- | --- |
| A := 3  Return A  Let A in A := 4 ;return A end  Return A | <- Retourne 3  <- Retourne 4  <- Retourne 3 |

# Comment implémenter cette commande ?

La création de l’arbre ne pose absolument aucun souci. Le problème se pose au niveau de l’interprétation et la zone mémoire.

Par conséquent, on créer une zone de mémoire intermédiaire qui contient le nom de la variable ainsi que sa valeur. De ce fait, lorsqu’on veut faire une assignation, on va d’abord regarder dans la zone temporaire, puis seulement après si on n’a pas trouvé on accède à la mémoire globale. Du coup, avec ce principe, la variable n’est modifiée seulement dans la commande let, et en sortant, elle reprend sa valeur initiale.