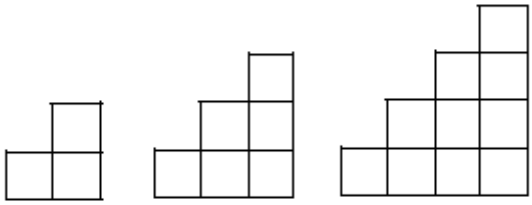
Problème d'escalier avec PluriAlgo

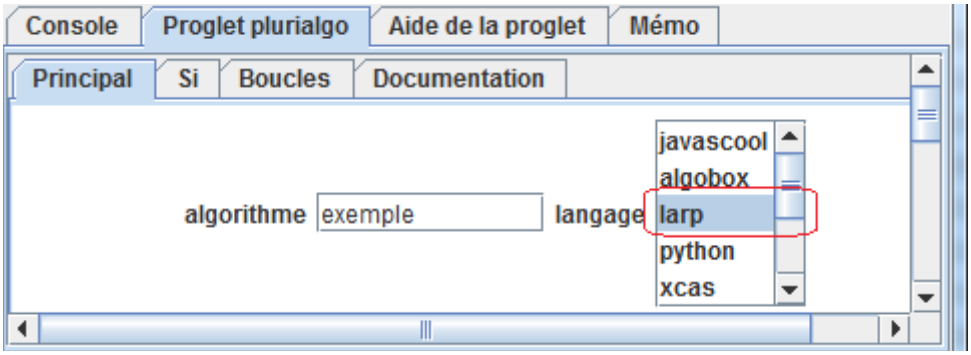


escaliers à 2, 3 et 4 marches

Ce document détaille les deux questions abordées dans l'article principal :

- calculer le nombre de cubes nécessaires pour fabriquer un escalier avec un nombre de marches fixé (n)
- trouver le plus grand escalier pouvant être construit avec un nombre de cubes fixé (nbCubes)

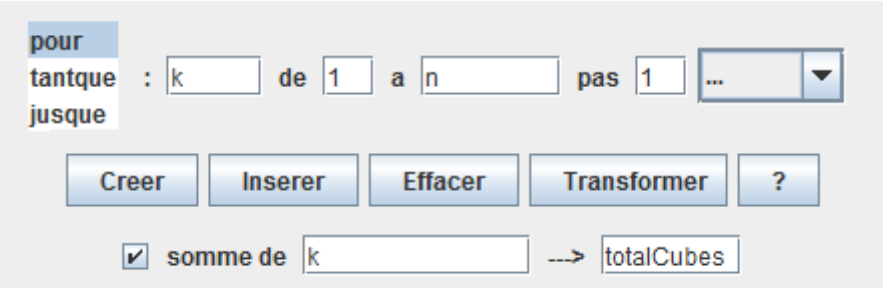
Les programmes obtenus sont exprimés en Larp (langage que j'apprécie particulièrement), mais les indications fournies dans l'article restent valides si l'on choisit un autre langage de développement dans l'onglet Principal.



choix du langage dans l'onglet Principal

Question 1 : nombre de cubes nécessaires (1+2+...+n)

Comme indiqué dans l'article principal, il faut utiliser l'option « somme » de l'onglet Boucles :



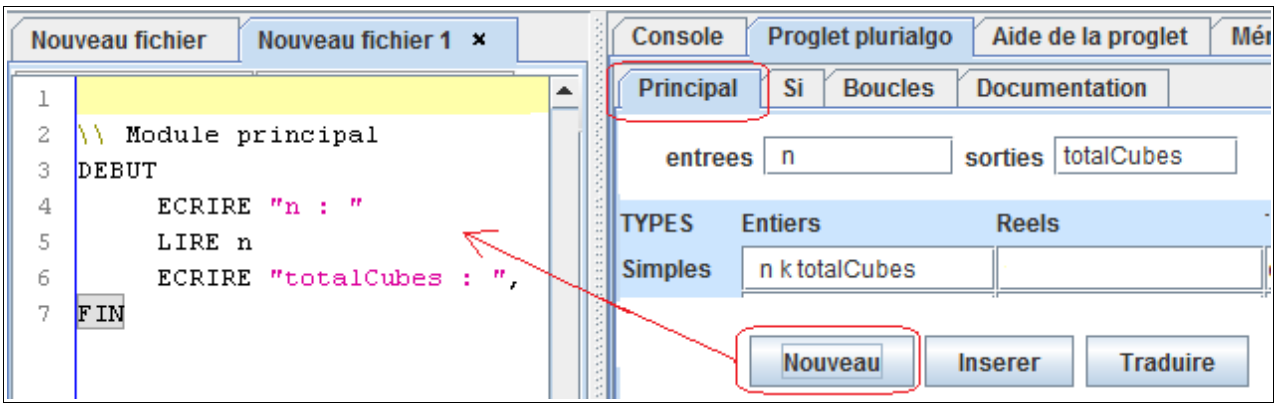
calcul de 1+2+...+n avec l'onglet Boucles

Version 1 : bouton Insérer

Le bouton **Insérer** permet d'insérer dans le programme courant de l'éditeur les instructions entourées en rouge :

```
DEBUT
     ECRIRE "n : "
     LIRE n
     totalCubes = 0
     POUR k=1 JUSQU'A n FAIRE
          totalCubes = totalCubes + k
     FINPOUR
     ECRIRE "totalCubes : ", totalCubes
FIN
```

Le programme initial (avec la saisie de n et l'affichage de totalCubes) peut être obtenu en complétant l'onglet Principal, puis en cliquant sur le bouton **Nouveau** :



Version 2 : avec le bouton Créer

Le bouton **Créer** permet d'obtenir directement le résultat final, sans avoir à gérer les entrées-sorties dans l'onglet Principal.

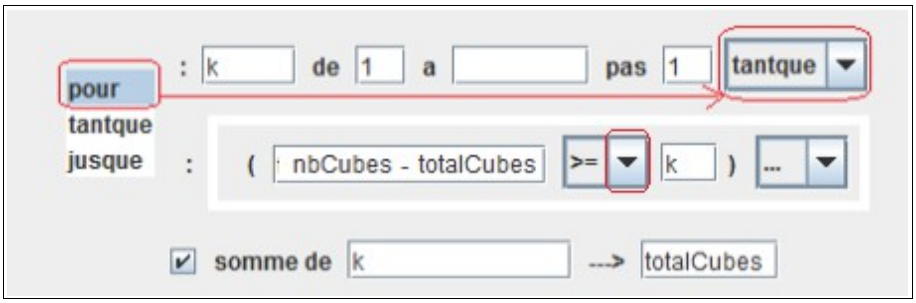
Question 2 : recherche du plus grand escalier

Comment aider les étudiants … sans trop leur en dire !

```
DEBUT
      ECRIRE "nombre de cubes disponibles"
      LIRE nbCubes
      totalCubes = 0
      k = 1
      TANTQUE (totalCubes - nbCubes>=k) FAIRE
            totalCubes = totalCubes + k
            k = k+1
      FINTANTQUE
      ECRIRE "nombre de marches maximal :", k-1
FIN
```

algorithme final

Les instructions entourées en rouge peuvent être obtenues en complétant l'onglet Boucles, puis en cliquant sur le bouton **Insérer** :

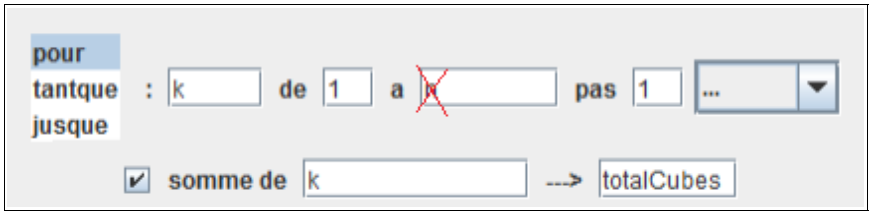


transformation d'un Pour en Tantque

Dans le cadre d'une initiation à l'algorithme, cette approche ne peut être proposée directement aux étudiants : en effet, elle leur fournirait immédiatement la solution, les privant ainsi de tâtonnements et d'expérimentations nécessaires pour progresser. Plusieurs alternatives pédagogiques sont donc examinées, pour aider les étudiants … sans trop leur en dire !

Approche 1

On reprend les indications de la question 1, en supprimant la variable n puisque le nombre d'itérations est inconnu :



Comme la valeur d'arrivée de la boucle Pour n'est pas spécifiée, PluriAlgo transforme le Pour en Tantque quand on clique sur le bouton **Insérer** :

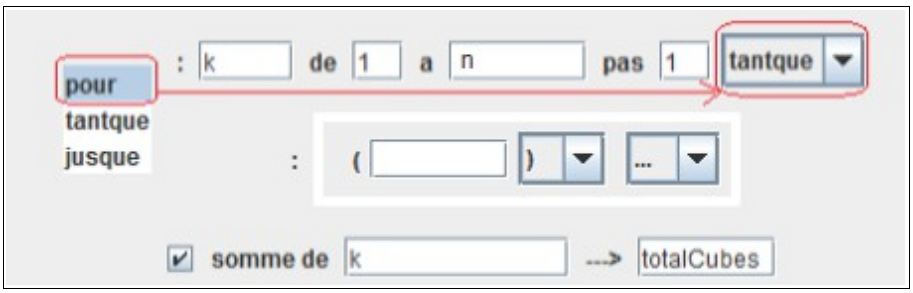
```
totalCubes = 0
k = 1
TANTQUE () FAIRE
     totalCubes = totalCubes + k
     k = k+1
FINTANTQUE
```

à compléter

Il reste à compléter la condition de la boucle, puis à tester le programme dans un environnement adapté au langage choisi.

Approche 2

On reprend les indications de la question 1, en demandant explicitement à ce que le Pour soit transformé en Tantque :



Il reste à adapter le code obtenu en cliquant sur le bouton **Insérer** :

```
totalCubes = 0
k = 1
TANTQUE (k<=n) FAIRE
     totalCubes = totalCubes + k
     k = k+1
FINTANTQUE
```

à modifier

Approche 3

Cette alternative peut être proposée aux étudiants peinant à conclure avec l'approche 1 ou de l'approche 2. Elle consiste à partir d'un algorithme à trous où les instructions du Tantque sont inversées (k=k+1 devient la première instruction du tantque) :

```
totalCubes = 0
k =
TANTQUE (   ) FAIRE
      k = k+1
      totalCubes = totalCubes + k
FINTANTQUE
```

à compléter

L'avantage de cette solution est, me semble-t-il, d'avoir une condition de boucle plus facile à trouver :

```
DEBUT
      ECRIRE "nombre de cubes disponibles"
      LIRE nbCubes
      totalCubes = 0
      k = 0
      TANTQUE ( totalCubes <= nbCubes ) FAIRE
            k = k+1
            totalCubes = totalCubes + k
      FINTANTQUE
      ECRIRE "nombre de marches maximal :", k-1
FIN
```