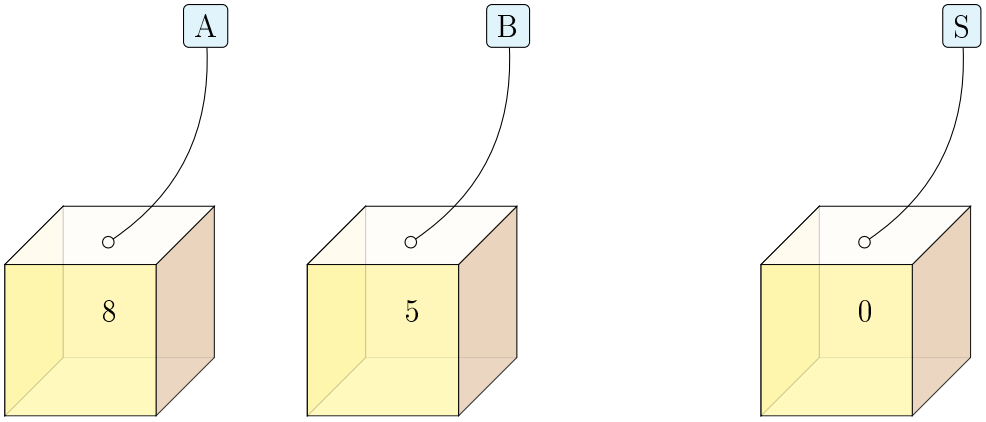
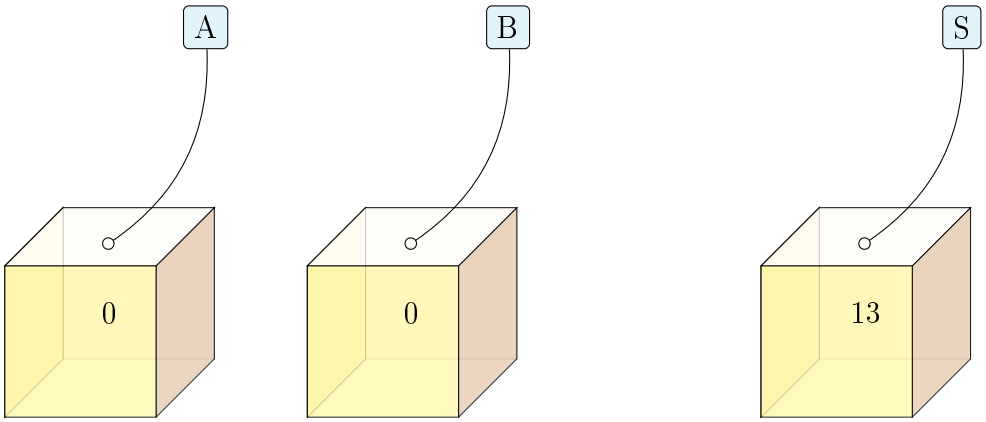
Algorithme d'addition

Voici la situation avant l'addition (état initial) :

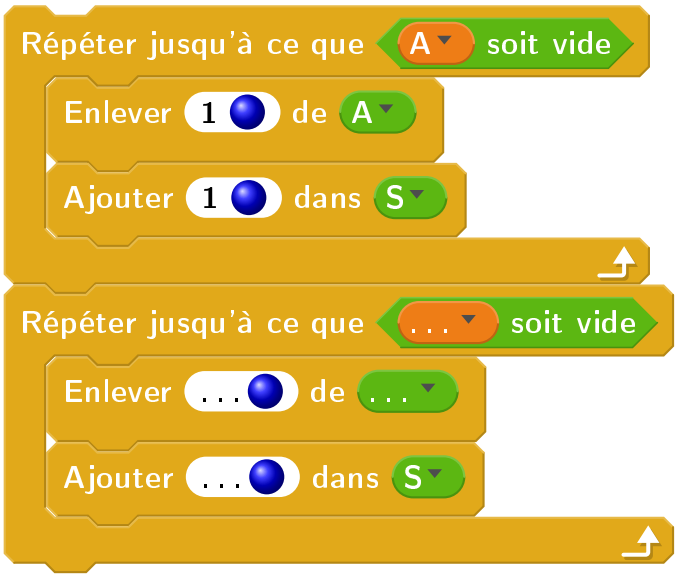


Cela signifie qu'il y a 8 jetons dans la boîte étiquetée A et 5 jetons dans la boîte étiquetée B. La boîte étiquetée S est initialement vide. Un algorithme d'addition transforme cet état en l'état final suivant où le contenu de S est :

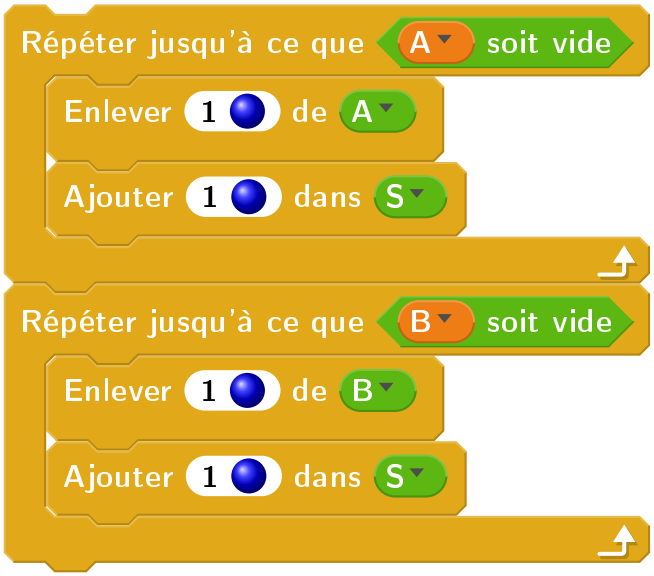


Autrement dit, additionner les contenus de A et B, revient à transférer ces contenus dans S. Le faire par revient à appliquer un *algorithme*.

Compléter cet algorithme pour qu'il fasse effectivement l'addition :

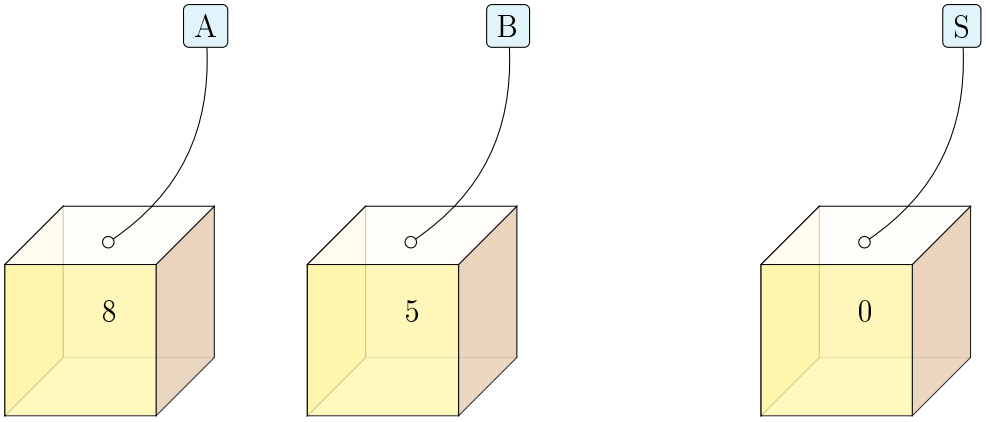


Corrigé

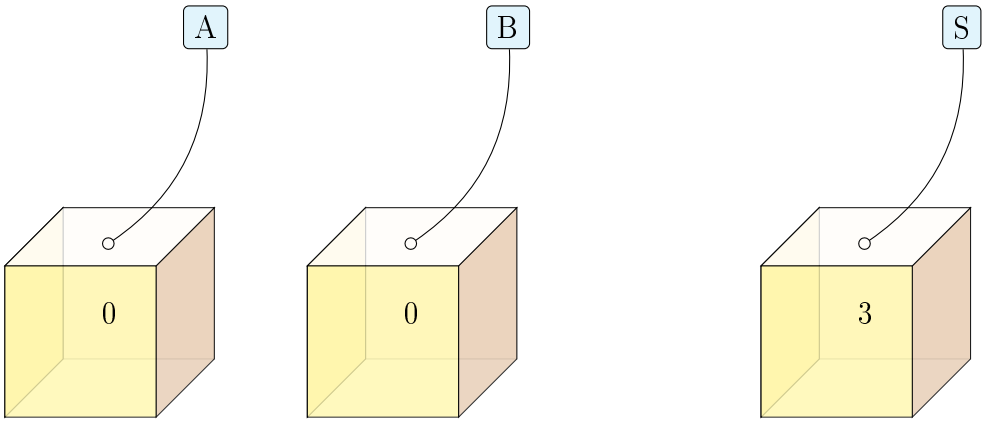

Répéter jusqu'à ce que A soit vide
Enlever 1 de A
Ajouter 1 dans S
Répéter jusqu'à ce que B soit vide
Enlever 1 de B
Ajouter 1 dans S

Algorithme de soustraction

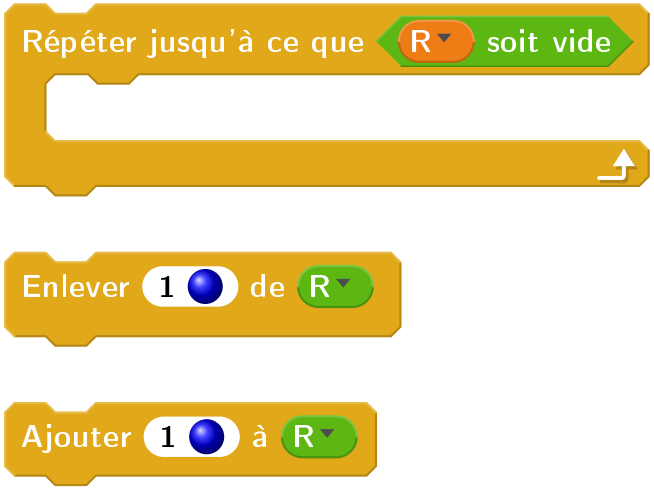
Voici la situation avant la soustraction (état initial) :



Cela signifie qu'il y a 8 jetons dans la boîte étiquetée A et 5 jetons dans la boîte étiquetée B. La boîte étiquetée S est initialement vide. Un algorithme de soustraction transforme cet état en l'état final suivant où le contenu de S est :

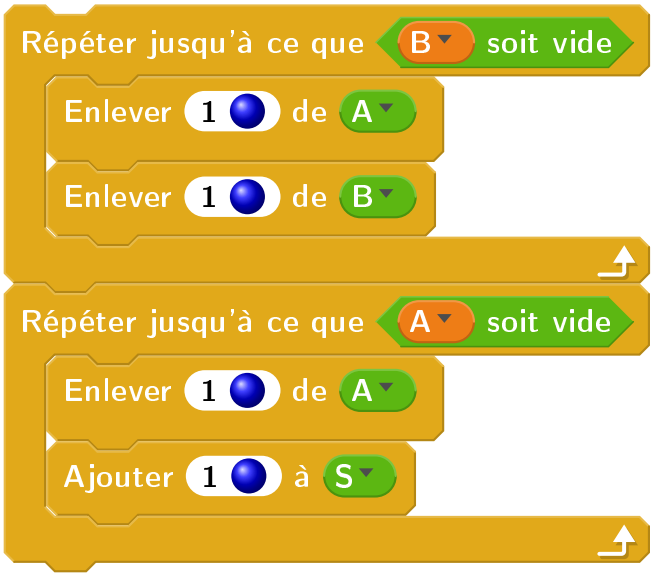


On suppose qu'au départ il y a au moins autant de dans A que dans B. Pour construire l'algorithme de soustraction, on assemble des blocs de ce type (remplacer R par A, B ou S) :



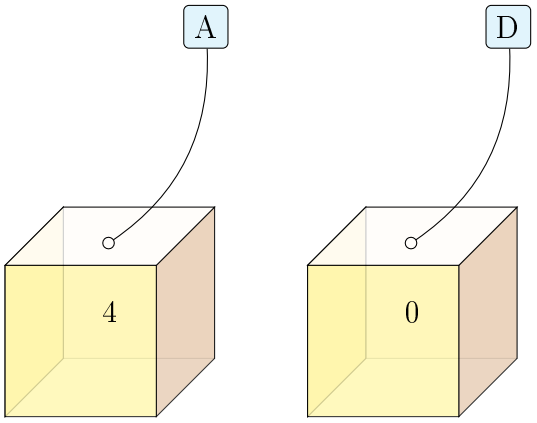
Écrire un tel algorithme. Il faut que pour n'importe quel nombre de dans A et dans B, le résultat de la soustraction soit dans S à la fin.

Corrigé

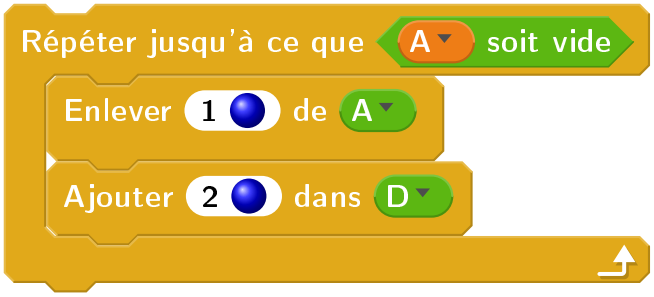

Répéter jusqu'à ce que B soit vide
Enlever 1 de A
Enlever 1 de B
Répéter jusqu'à ce que A soit vide
Enlever 1 de A
Ajouter 1 à S

Algorithme de doublement

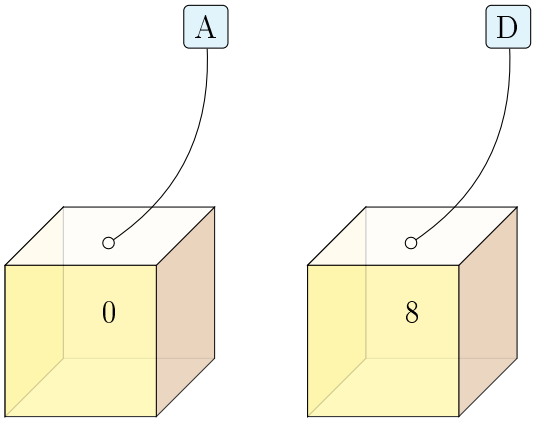
On suppose qu'au début la variable A contient 4 jetons et que le registre D est vide :



Calculer par algorithme le double de A c'est faire en sorte qu'à la fin il y ait 8 jetons dans D. L'algorithme suivant fait en sorte qu'en plus A est vide à la fin :



La situation finale est la suivante :



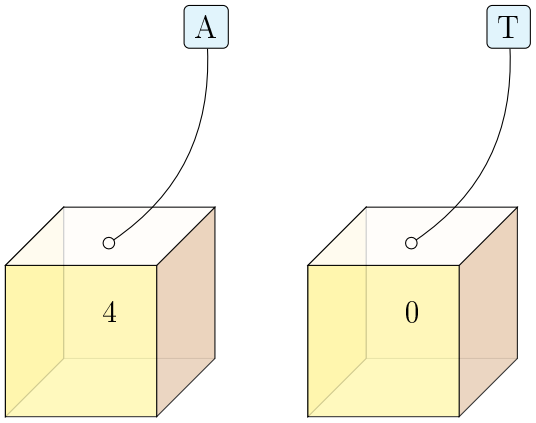
Algorithme de triplement

Écrire un algorithme qui calcule le triple de la variable A.

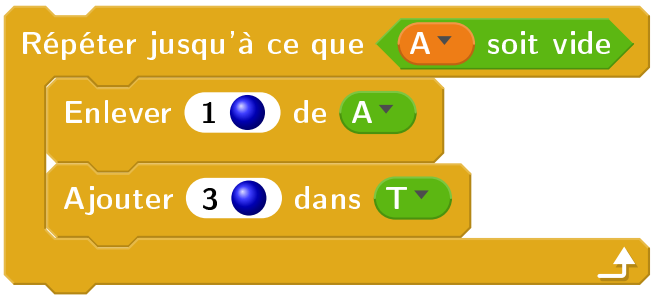
Algorithme de quadruplement

Écrire un algorithme qui calcule le quadruple de la variable A.

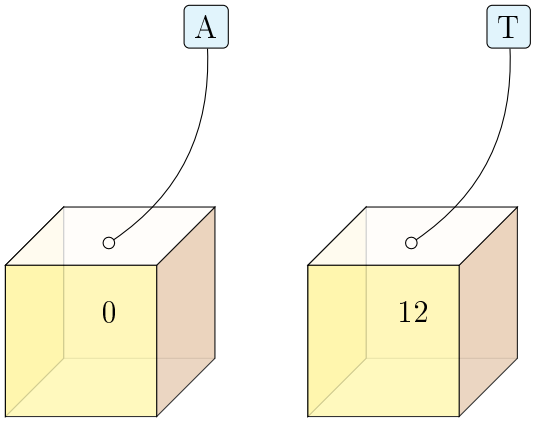
Corrigé
En partant de la situation initiale



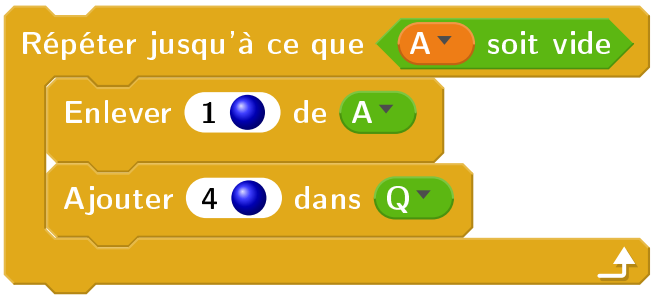
cet algorithme



mène à la situation finale

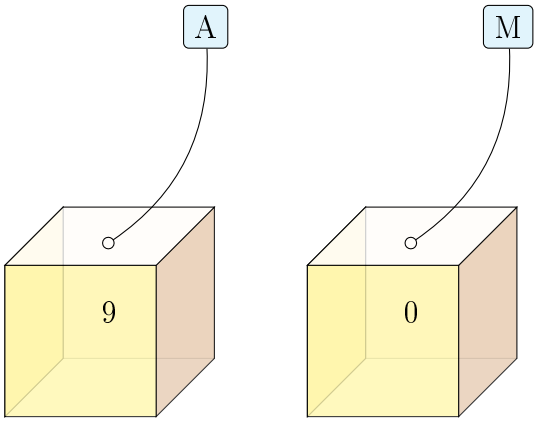


Voici un algorithme de quadruplement :

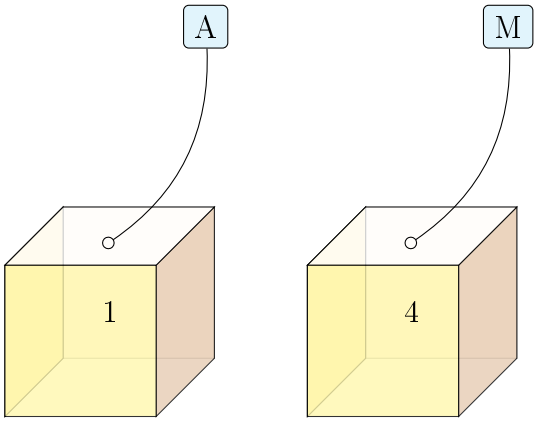


Division par 2

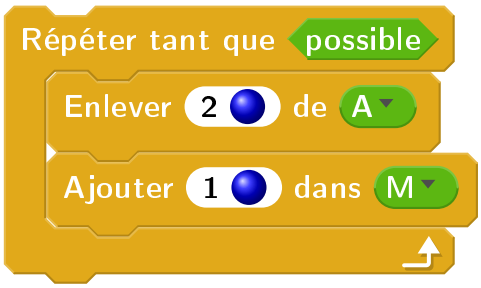
On suppose qu'au début la variable A contient 9 et que le registre M est vide :



Calculer par algorithme la moitié de A c'est faire en sorte qu'à la fin il y ait 4 jetons dans M :



S'il reste un jeton dans A à la fin (comme ci-dessus) ça veut dire que le nombre de départ (ici, 9) est impair. Tester, pour plusieurs valeurs initiales de A, l'algorithme ci-dessous :



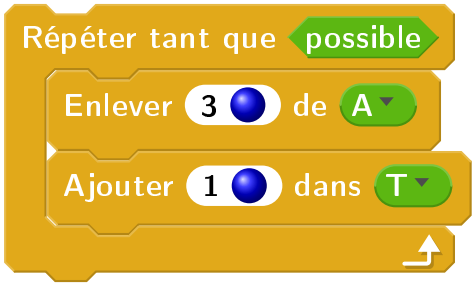
Division par 3

Écrire un algorithme qui divise A par 3.

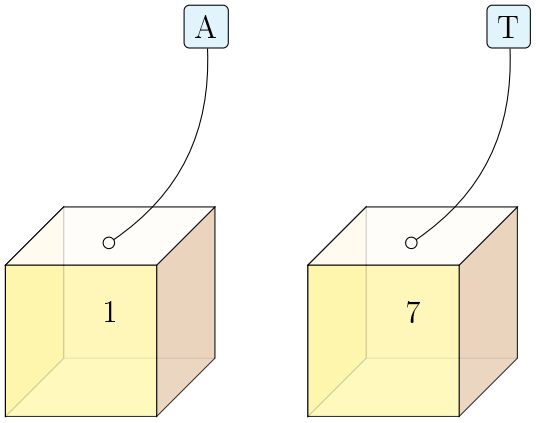
Division par 4

Écrire un algorithme qui divise A par 4. Combien y a-t-il de restes possibles à la fin de l'algorithme ?

Corrigé
Division par 3

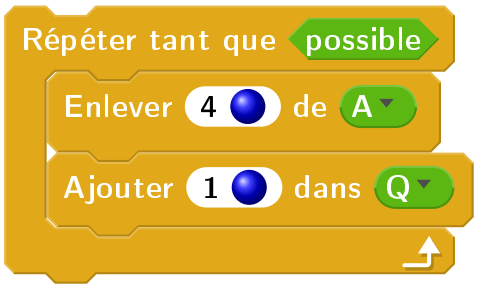


S'il y avait initialement 22 jetons dans A, on finit par

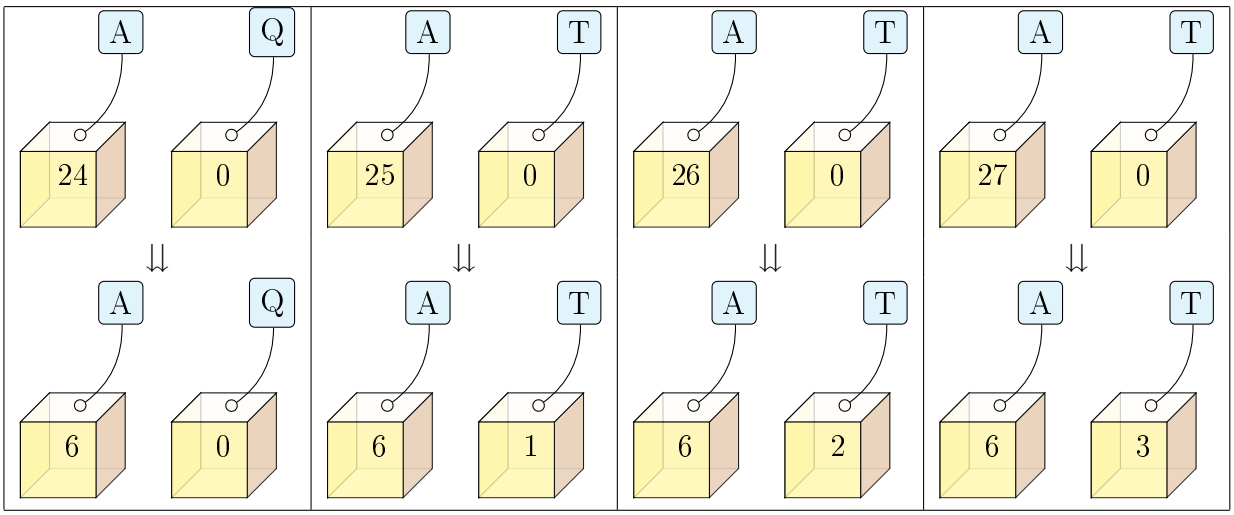


Les 6 dans T signifient que le tiers de 22 est 7 et la 1 dans A signifie que le reste dans la division est 1. Lorsqu'il ne reste que cette 1 il n'est plus possible d'enlever 3 à A.

Division par 4



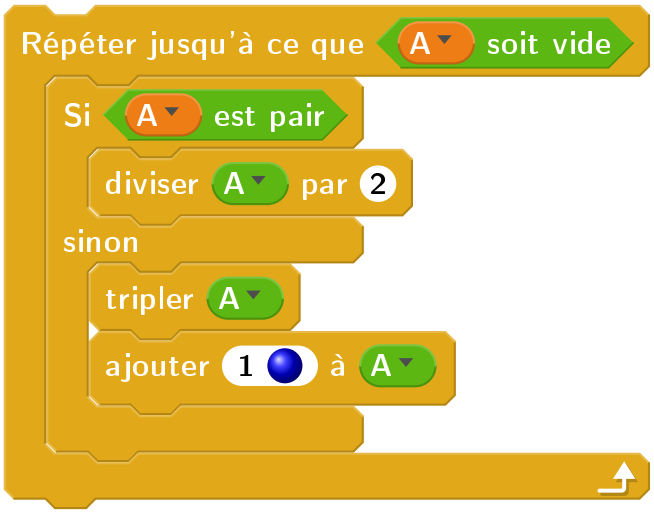
Le tableau ci-dessous (état initial en haut, état final en bas) montre que 5 est le quart de 24, mais aussi le quart de 25, de 26 et de 27 :



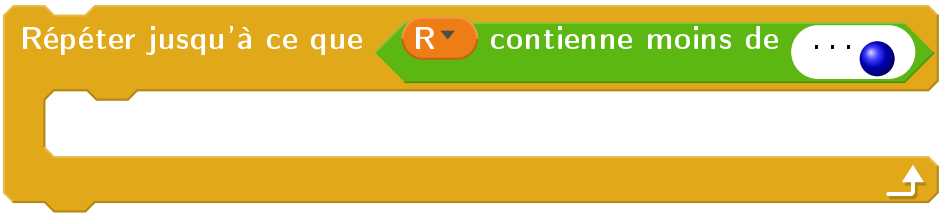
Il y a 4 restes possibles dans la division par 4 : A vide, ou 1, ou 2 ou enfin 3.

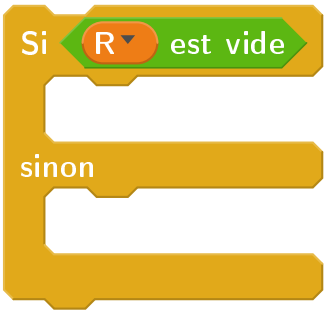
Suite de nombres de Collatz

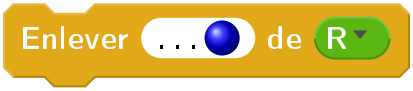
On suppose qu'au début la variable A contient ●●●. On applique le programme de calcul suivant :



Écrire un algorithme permettant de simuler ce programme de calcul avec les contenus successifs de la variable A. On pourra utiliser des variables auxiliaires B, C etc. On peut avoir besoin de ces blocs (R peut être remplacé par A etc) :



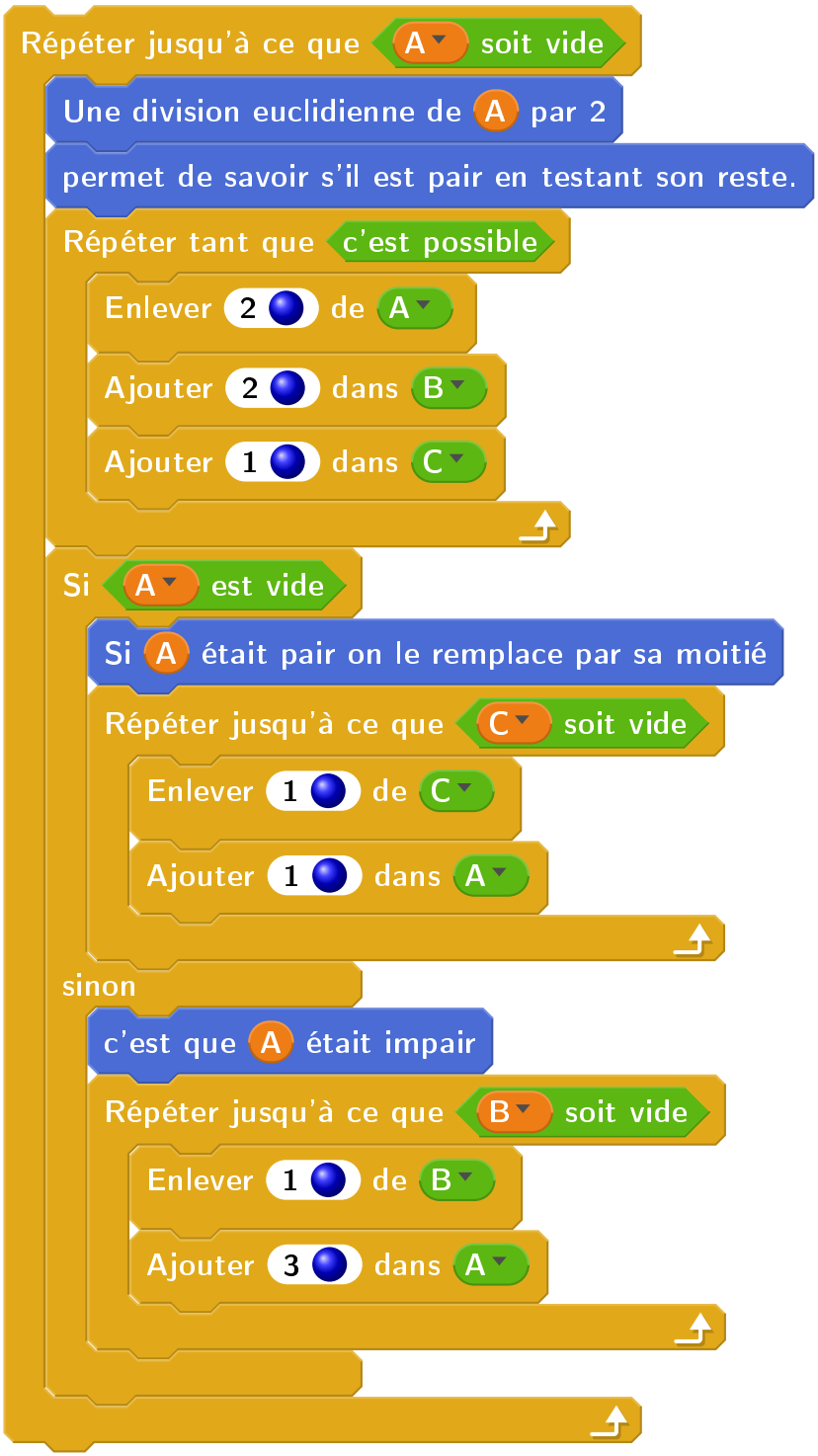




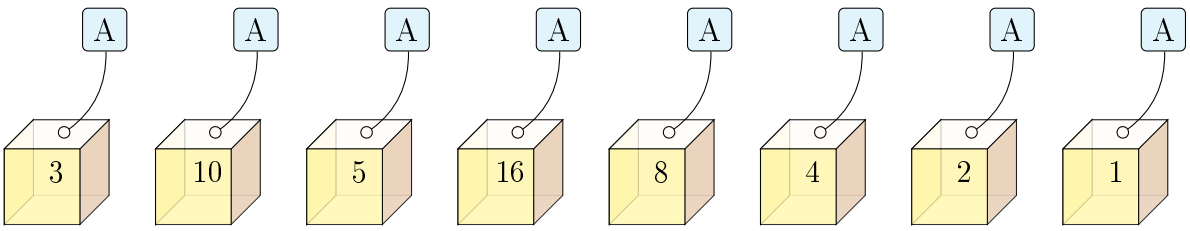
Si on n'arrête que lorsque A est vide, le programme ne s'arrête jamais. Par quoi faut-il remplacer cette condition pour que le programme s'arrête ?

Corrigé

On utilise B pour garder une copie de A et C pour garder la moitié de A au cas où on en a besoin :



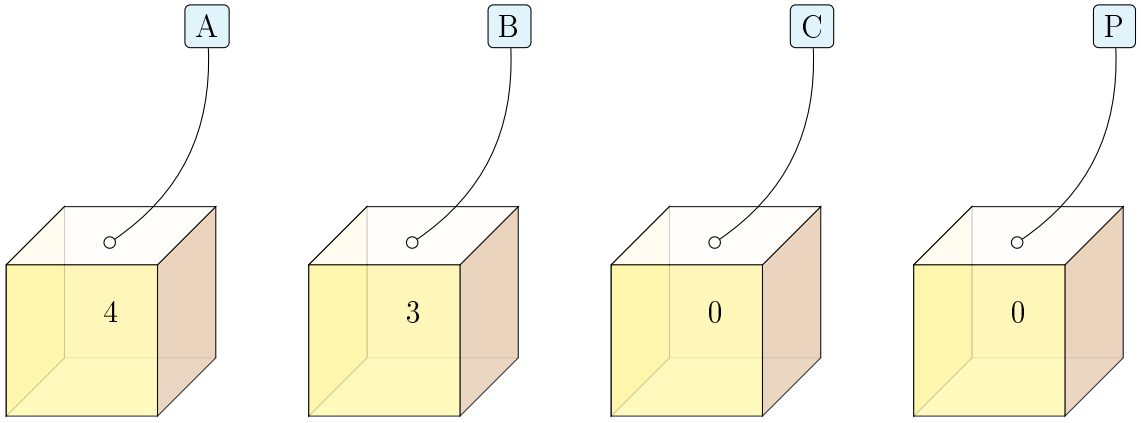
Voici les états successifs de la variable A :



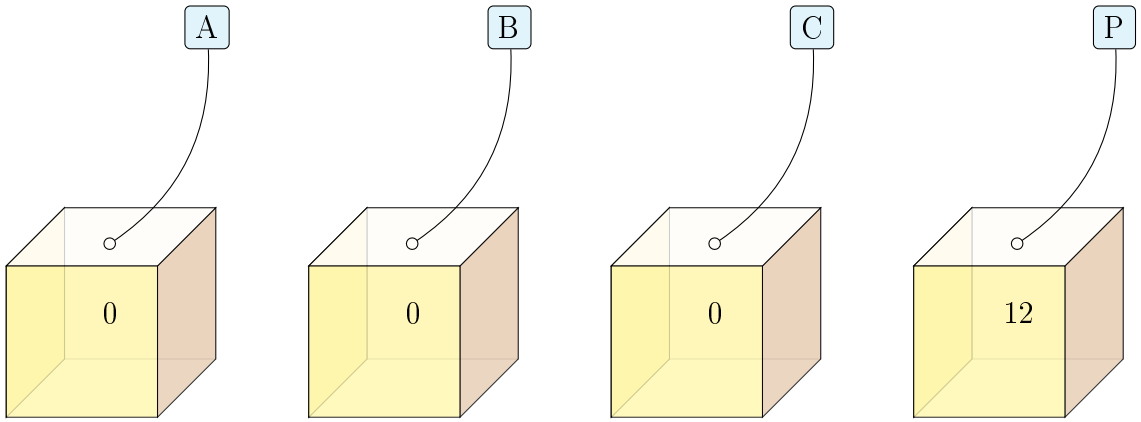
Comme après ça on a de nouveau 4, le programme reste dans le cycle 4-2-1 et A ne se vide jamais. Mais en arrêtant le programme lorsqu'il reste une ● dans A, il s'arrête à l'étape ci-dessus.

Multiplication

On suppose qu'au début la variable A contient 4 jetons et que la variable B contient 3 jetons. On suppose que les variables C et P sont initialement vides :

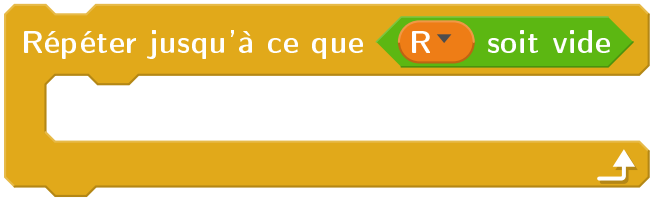


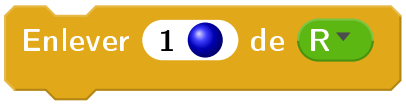
On veut qu'à la fin de l'algorithme il y ait 12 jetons dans P, c'est-à-dire une situation finale comme ceci :



Écrire un tel algorithme, qui, pour n'importe quel choix des nombres de jetons placés au début dans A et dans B, finisse avec le produit de ces deux nombres dans P.

On aura besoin des blocs suivants (R est à remplacer par A ou B ou C ou P) :

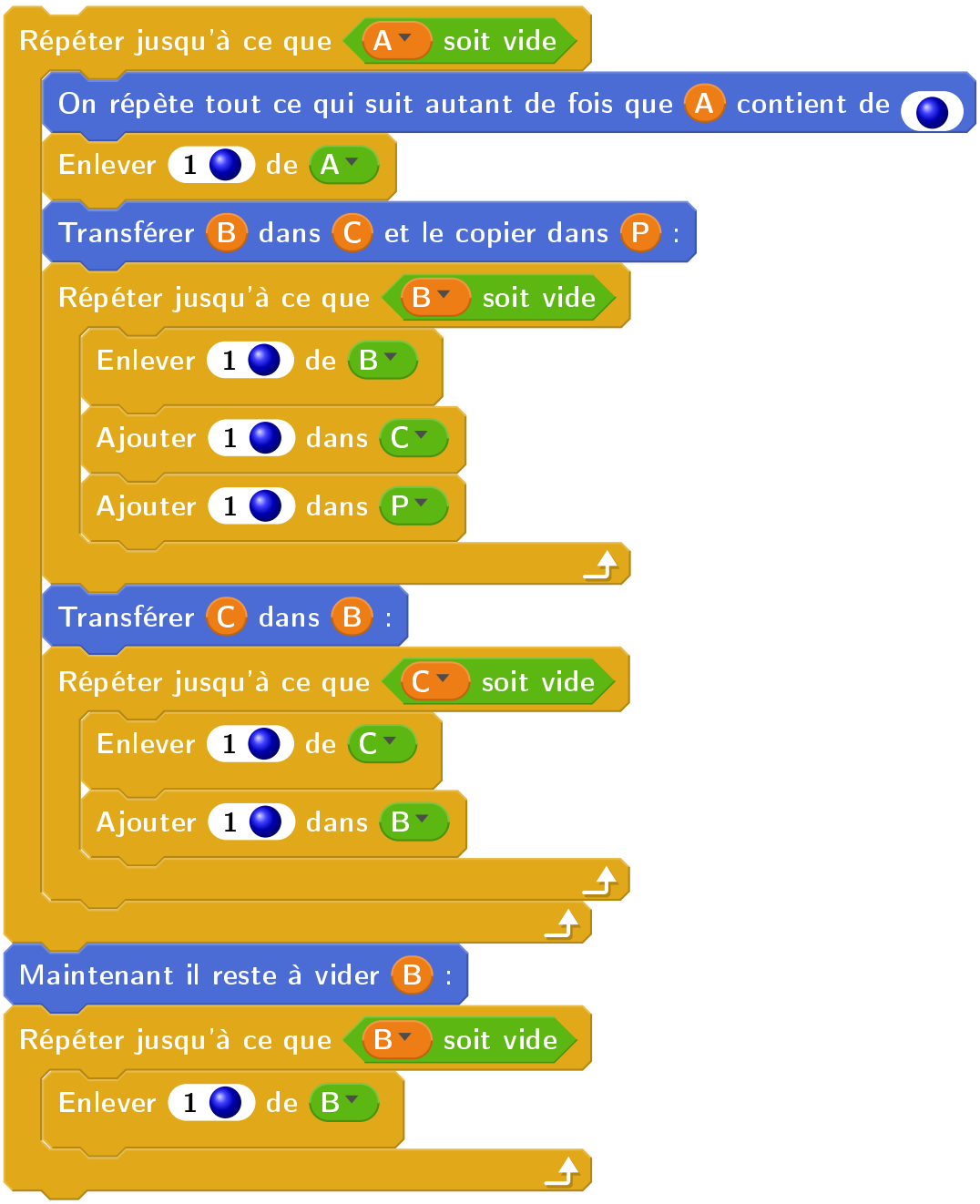




Corrigé

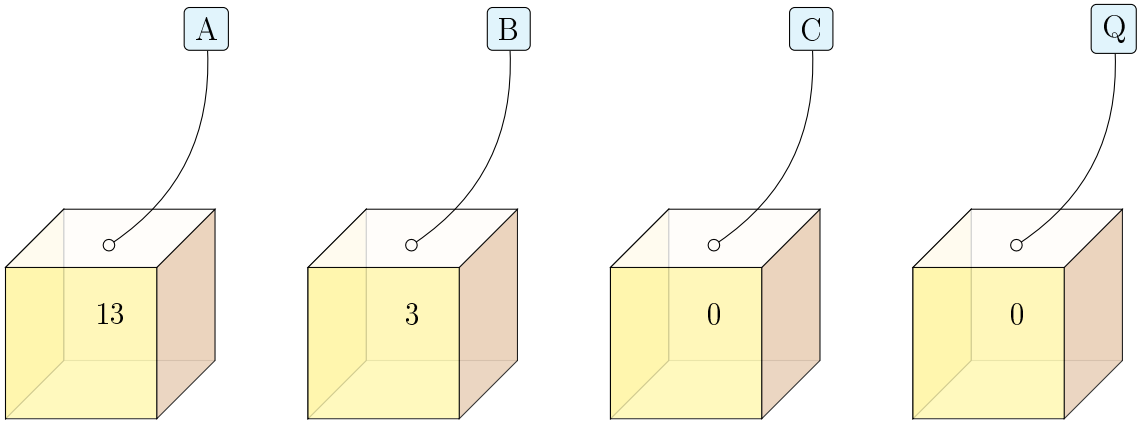
Une multiplication est une addition itérée : Si A contient initialement 4 ●, on doit effectuer 4 additions pour avoir le produit. Pour additionner B, on le copie 4 fois dans P.

Mais cela a pour effet de vider B alors on le copie aussi dans C pour pouvoir ensuite recharger B avec son ancien contenu.

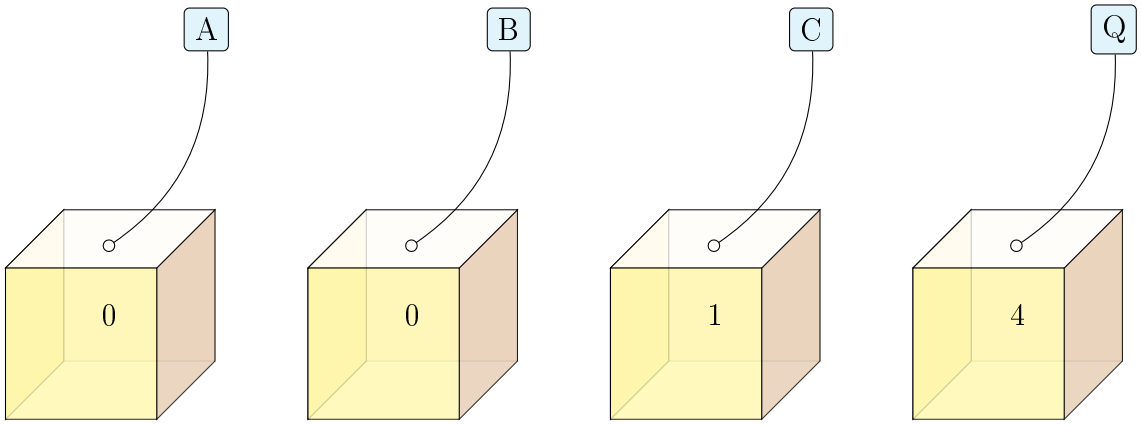


Division euclidienne

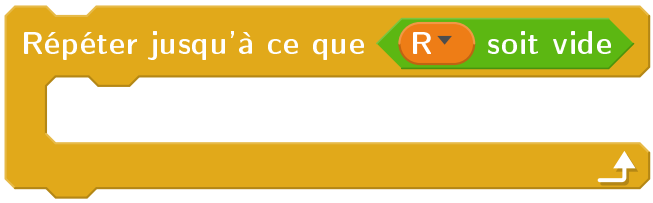
On suppose qu'au début la variable A contient 13 et que la variable B contient 3. Les variables C et Q sont initialement vides :

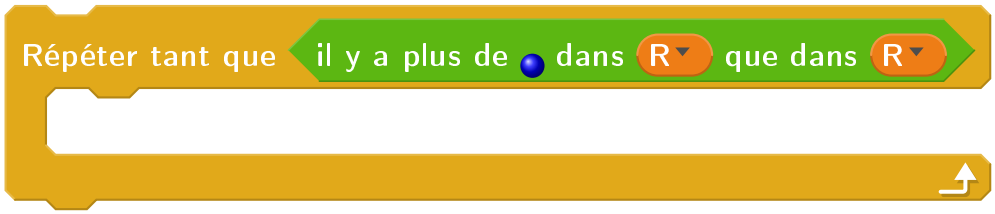


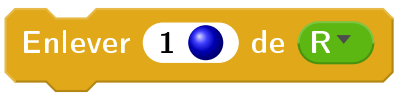
On veut un algorithme qui, dans l'état final, laisse le quotient de 13 par 3 dans Q et le reste dans C :



On aura besoin des blocs suivants (R peut être remplacé par A ou B ou C ou Q) :



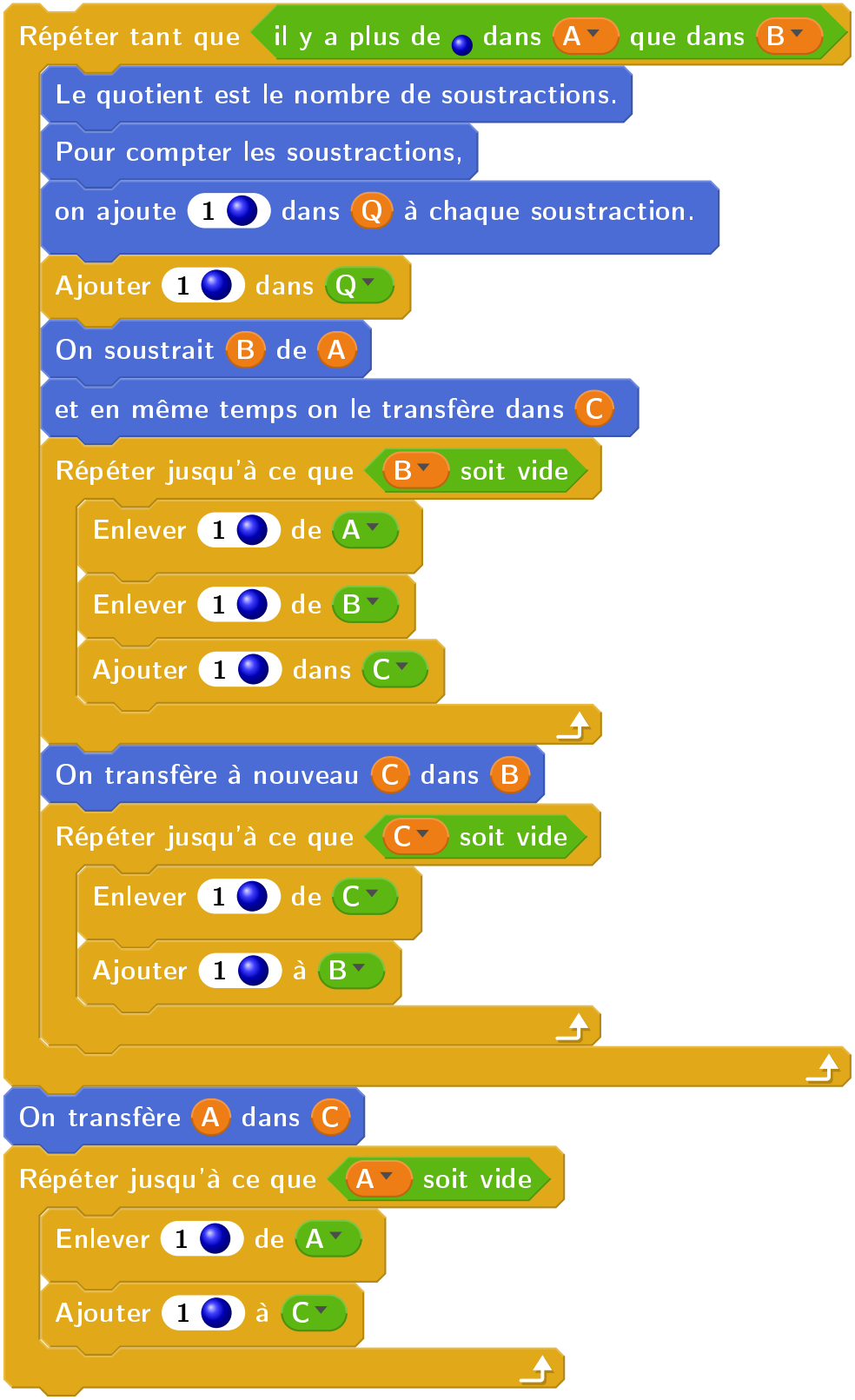




Corrigé

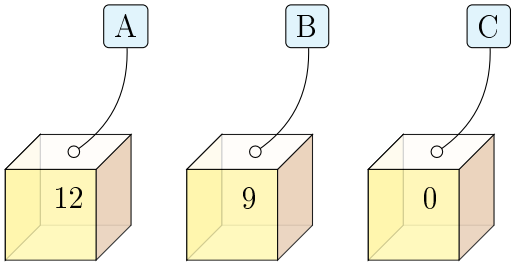
La propriété d'Archimède dit que pour n'importe quel choix des nombres initialement placés dans A et B, on peut soustraire B à A jusqu'à ce qu'il reste moins de ● dans B que dans A.

C'est donc en soustrayant répétitivement B à A que l'on calcule le reste (contenu final de A). Le nombre de soustractions effectuées est le quotient euclidien de A par B.

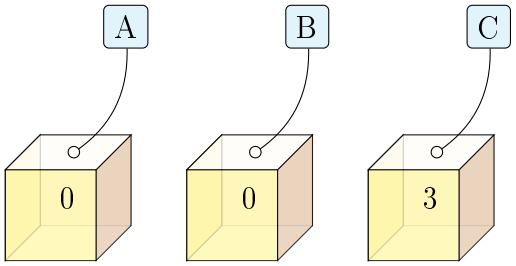


Algorithme d'Euclide

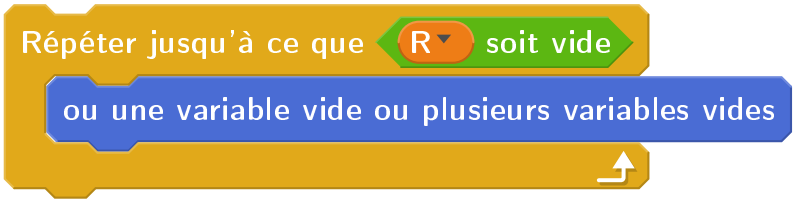
On suppose qu'au début la variable A contient ●●●●●●●●●●●● et que la variable B contient ●●●●●●●●●. La variable C est initialement vide :

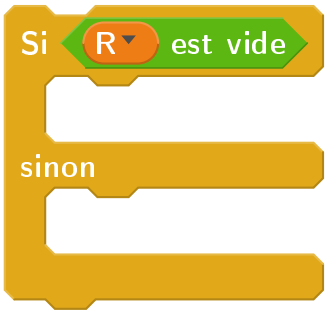


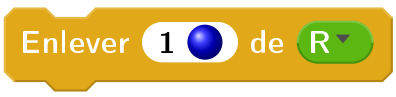
On veut un algorithme qui, dans l'état final, laisse le pgcd de 12 et 9 dans C :



On aura besoin des blocs suivants (R peut être remplacé par A ou B ou C) :







Corrigé


Répéter jusqu'à ce que A et B soient vides tous les deux
On soustrait le plus petit au plus grand
Répéter jusqu'à ce que A ou B soit vide
Enlever 1 de A
Enlever 1 de B
Ajouter 1 dans C
si A est vide
On transfère C dans A
Répéter jusqu'à ce que C soit vide
Enlever 1 de C
Ajouter 1 dans A
sinon
On transfère C dans B
Répéter jusqu'à ce que C soit vide
Enlever 1 de C
Ajouter 1 dans B