e) D'une fonction `unsigned int pgcd (int entierA, int entierB)`, qui retourne un nombre entier > 0 correspondant au plus grand entier > 0 divisant à la fois entierA et entierB.

Euclide (3eme siècle avant JC) a fourni une méthode pour calculer le PGCD de deux nombres.
Cette méthode est connue sous le nom d'algorithme d'Euclide.
Données : entierA, entierB, entierA >= entierB > 0
Résultat : pgcd, entier > 0

Elle se base sur le constat suivant :
La division entière de entierA par entierB s'écrit :
entierA = quotient * entierB + reste, avec 0 <= reste < entierB

En itérant cette division, et en remplaçant entierA par entierB et entierB par reste, on aboutira finalement à un reste nul.
➔ Le pgcd est alors le dernier reste non nul.

Exemple : PGCG(42, 30)
42 = 1 * 30 + 12
30 = 2 * 12 + 6
12 = 2 * 6 + 0 → le PGCD est 6, le dernier reste non nul

<u>Nous généraliserons l'algorithme</u> : il calculera le pgcd de 2 nombres entiers, positifs ou négatifs,
avec la pré-condition suivante : les 2 entiers sont tels qu'au moins 1 d'eux est différent de 0.
Le PGCD calculé sera > 0.

<u>Comportement attendu :</u>

pgdc(0, 2) = 2
pgcd (2, 0) = 2
pgdc(0, -2) = 2
pgcd (-2, 0) = 2
pgcd (2, -4) = 2
pgcd (-2, 4) = 2
pgcd (3, 5) = 1
pgcd (42, 30) = 6
pgcd (2, 2) = 2
pgcd (1, 1) = 1
pgcd(0, 0) : résultat non garanti car pré-condition non respectée

Dictionnaire des éléments

entierA, entierB	Entiers positifs ou négatifs, mais pas nuls en même temps	Les paramètres Données de la fonction
a	a >= 0	Initialisé à abs(entierA), puis dividende de la division euclidienne
b	b >= 0	Initialisé à abs(entierB), puis diviseur de la division euclidienne
reste	Entier >= 0 et < b	Reste de la division euclidienne de a par b
lePgcd	Entier > 0	Pgcd de a et de b

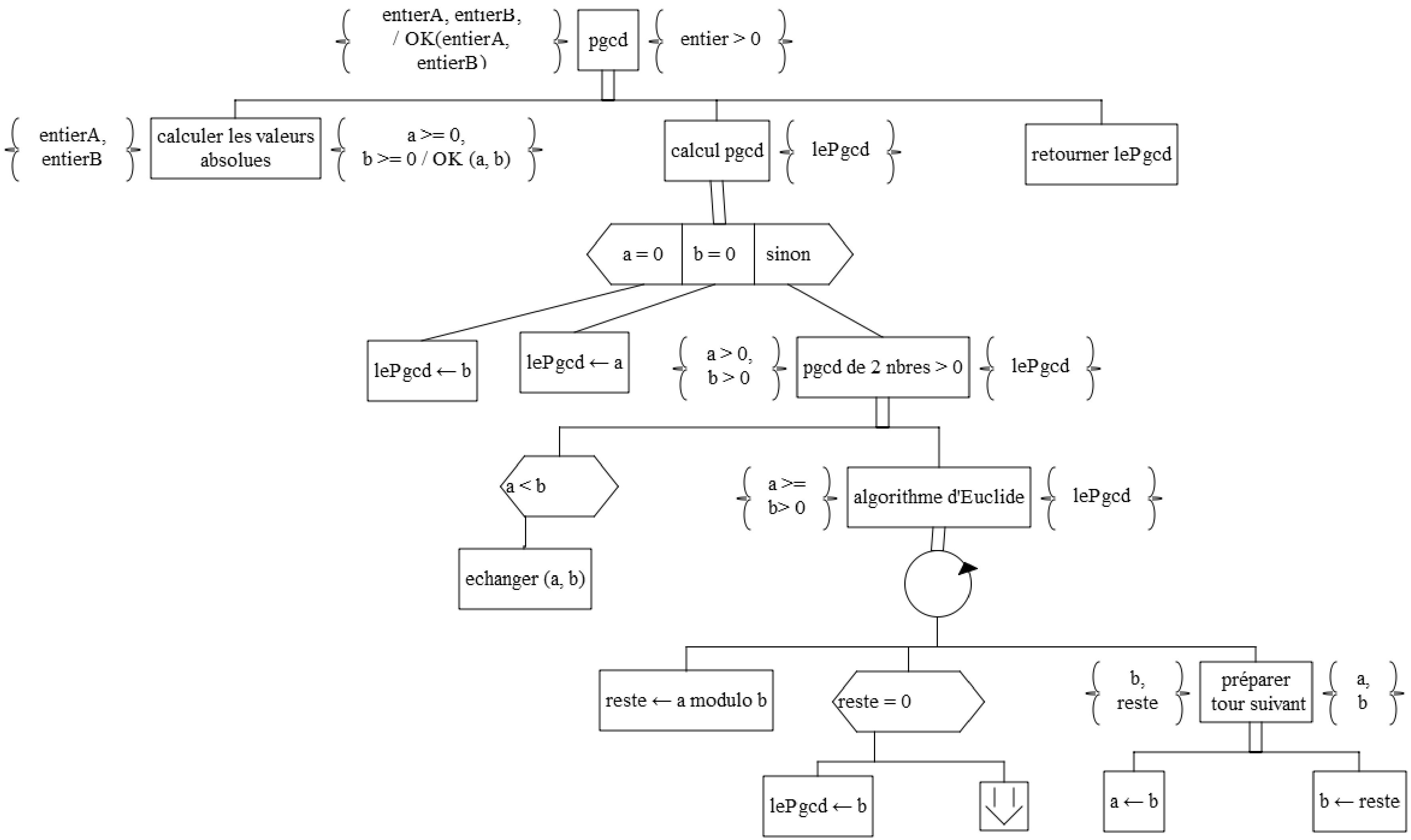

Figure 10 : Algorithme de la fonction pgcd()

Et **pré-condition OK (entierA, entierB)** : au moins un des 2 nombres est différent de 0.
post-condition OK (a b) : au moins un des 2 nombres est différent de 0.