```
#exo 3
def S1(a:bool,b:bool,c:bool)->bool:
    sortie= not(a or b) and (not c or b)
    return sortie

def S1b(a:bool,b:bool,c:bool)->bool:
    assert (type(b)==bool and type(c)==bool and
type(a)==bool),"arguments booléens attendus"
    sortie= not(a or b) and (not (c) or b)
    return sortie
#appel de la fonction : >>>S1b(False,False,
False)

#exo 4
def S2(a:bool,b:bool,c:bool)->bool:
    sortie= a and (b or not(c)) or not(a+c)
    return sortie

#exo5
# Si a > b  alors X = 1
# Si a+b <= 10 alors Y = 0
# Si a+b > 10 alors Y = 1

# X Y   A   B
# 0 0   1   1
# 0 1   1   0
# 1 0   1   1
# 1 1   0   1

# A = not(X and Y)
# B = not(not(X) and Y)

def ouvertureBarriere(a:int,b:int)->str:
    # On teste s'il n'y a qu'un seul camion
    if a == 0:
        return 'B'
    elif b == 0:
        return 'A'

    # Transformation des valeurs a et b en
booleens
    X , Y = False , False
    if a > b:
        X = True
    if a + b > 10:
        Y = True

    # Fonctions boolennes
    A = not(X and Y)
    B = not(not(X) and Y)

    # Decision de la barriere a ouvrir
    if A == True and B == True:
        return 'AB'
    elif A == True and B == False:
        return 'A'
    else:
        return 'B'
```