1. Convergence des k-moyenne vers une solution non optimale

1. Le barycentre de $\{1,2,9,12\}$ est $b_1 = 6$, celui de $\{19\}$ est 19 donc la partition est stable et son inertie est $I_1 = (5+4+3+6)+(0) = 18$

2. Le barycentre de $\{1\}$ est $b_1 = 1$, celui de $\{2,9,12,19\}$ est $b_2 = \dfrac{21}{2}$ donc on réorganise en $\{1,2\}$ et $\{9,12,19\}$ dont les barycentres sont $b_1 = \dfrac{3}{2}$ et $b_2 = \dfrac{40}{3}$ qui est donc stable. L'inertie finale est donc $I_2 = \left(\dfrac{1}{2}+\dfrac{1}{2}\right)+\left(\dfrac{13+4+17}{3}\right) = 12$

3. Le deuxième partitionnement correspond à un autre minimum local, plus bas que le premier, donc la partition fournie par le premier exemple n'est pas optimale

2. Éviter les sous ensembles vides

1. `part[i]` est une liste qui représente un des sous-ensemble de la partition

```
def vide(part) :
    L = []
    for i in range(len(part)) :
        if len(part[i]) == 0 :
            L.append(i)
    return L
```

2. On commence par chercher un premier indice correspondant à un sous-ensemble de longueur $\geqslant 2$ puis on continue le parcours de `part` jusqu'à trouver le point recherché

```
def distMax(part) :
    indPart = 0
    while len(part[indPart]) < 1 :
        indPart += 1
    indPoint = 0
    barPart = barycentre(part[indPart])      # on mémorise ce barycentre
       pour ne pas avoir à le recalculer à chaque fois ; il est utile
       pour faire les comparaisons suivantes
    for i in range(indPart,len(part)) :
        if len(part[i]) > 1 :
            Bi = barycentre(part[i])
            for j in range(len(part[i])) :
                if d(part[i][j],Bi) > d(part[indPart][indPoint],barPart)
                   :
                    indPart = i
                    indPoint = j
                    barPart = Bi
    return indPart,indPoint
```

3. Après constitution d'une nouvelle partition, pour tout sous-ensemble vide, on le remplit avec le point de plus grande inertie (et on l'enlève du sous-ensemble auquel il appartenait pour conserver une partition) :

```
def kMoy(X,k) :
    n = len(X)
    part = initialisation(X,k)
    evolue = True
    while evolue :
        B = [barycentre(E) for E in part]
        part2 = [[] for _ in range(k)]
        for i in range(n) :
            ind = plusProche(X[i],B)
            part2[ind].append(X[i])
        L = vide(part2)
        while len(L) > 0 :
            i = L.pop()          # l'indice d'un sous-ensemble vide
            j,k = distMax(part2)    # la position du point de plus grande
                inertie
```

```
            part2[i].append(part2[j][k])     # on le rajoute dans le sous-
                ensemble vide
            part2[j] = part2[j][:k]+part2[j][k:]     # on le supprime de
                son sous-ensemble initial
        if part == part2 :
            evolue = False
        part = part2
    return part
```