L'objectif de cette colle est de proposer une implémentation de l´algorithme CHA dont le principe est rappelé ci-dessous :

Entrée : ensemble de données $x_1, \ldots, x_N$.
Début algorithme
  Créer N classes $\{x_1\}, \ldots, \{x_n\}$.
  Tant que nécessaire **Faire**
    Fusionner les deux classes les plus proches.

On représente une classe de points dans le plan par une liste des points, chaque point étant représenté par un couple de flottants correspondant aux coordonnées.

1. Écrire une fonction

   ```
   delta_simple : (float * float) list -> (float * float) list -> float
   ```

   qui calcule la distance $\Delta(a, b)$ de liaison simple entre deux classes a et b.

2. Écrire une fonction

   ```
   proximite : (float * float) list array -> float array array
   ```

   qui prend en argument un tableau de m classes et renvoie une matrice de proximité `mp` de dimensions $m \times m$ telle que `mp.(a).(b)` est égal à $\Delta(a, b)$.

3. En déduire une fonction

   ```
   classes_plus_proches : (float * float) list array -> int * int * float
   ```

   qui prend en argument un tableau de classes et renvoie un triplet $(a, b, \Delta(a, b))$ où a et b sont les indices des classes les plus proches.

4. Écrire une fonction

   ```
   cha : (float * float) list -> float -> (float * float) list list
   ```

   qui prend en argument une liste de points et un seuil et renvoie une partition en classes selon l'algorithme de clustering hiérarchique ascendant, en arrêtant les fusions dès que la distance minimale entre deux classes dépasse le seuil.