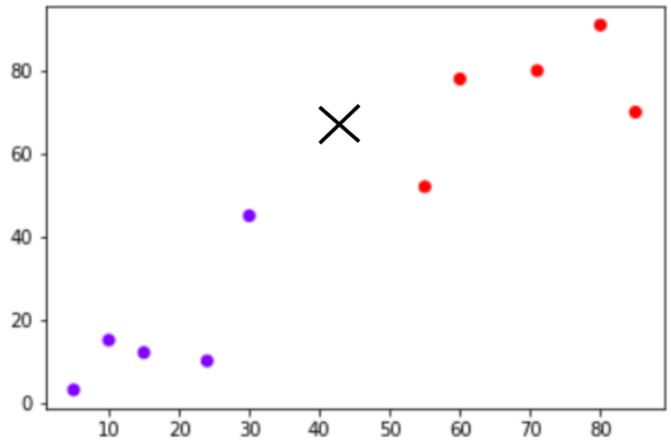
EXO1

Nous souhaitons savoir si la croix ci-dessous appartient au groupe rouge et au groupe bleu, en fonction de la couleur du voisin le plus prochain de cette croix.

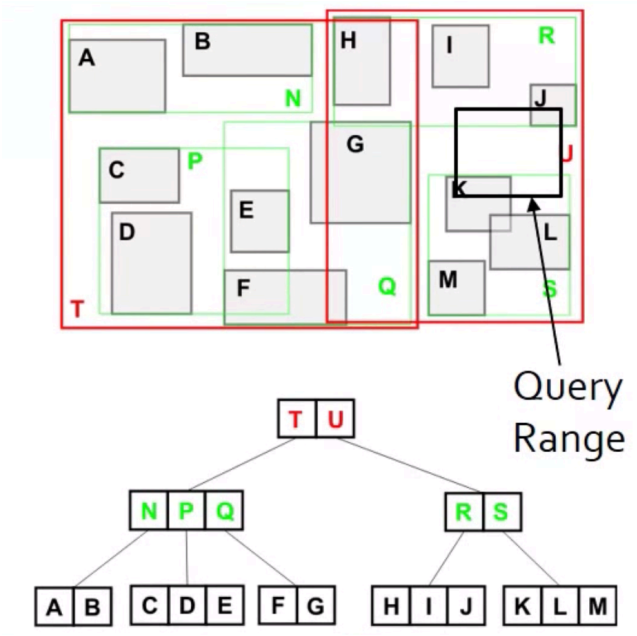
- Donner la formule permettant de calculer la distance entre la croix, notée (x,y), et un autre point i, noté (x_i,y_i).
- Ecrire la solution sous forme d'un algorithme utilisant la technique du voisin le plus proche (nearest-neighbor).



EXO2

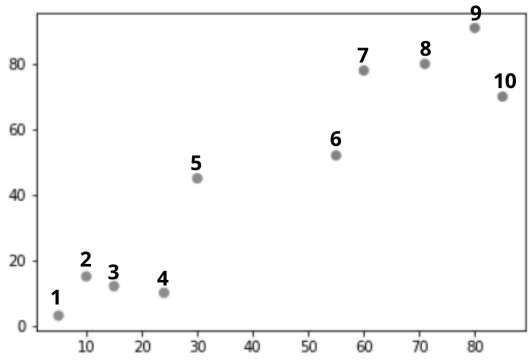
Nous possédons une carte avec des bâtiments (énuméré de a à m) et nous souhaitons connaître ceux qui touchent le rectangle noir (query range). Comme vous pouvez le constater, la carte est structurée hiérarchiquement.

- Comment s'appelle cette structure ?
- Comment fonctionne une recherche ? Enumérez les étapes.
- Quelle est l'avantage d'utiliser une telle structure ?



EXO3

Subdivisez le plan ci-dessous en un ***k-d-tree*** en suivant les numéros des points, c.-à-d. en commençant par le point 1, puis le point 2, etc.



Imaginons maintenant que nous souhaitions trouver le point le plus proche, par exemple, du point aux coordonnées (90,60). Quel est l'intérêt d'avoir subdivisé le plan en un k-d-tree ?

EXO4

On vous donne une liste « List_Coordinates » de coordonnées sur un plan cartésien (deux dimensions) et une coordonnée P (également deux dimensions).

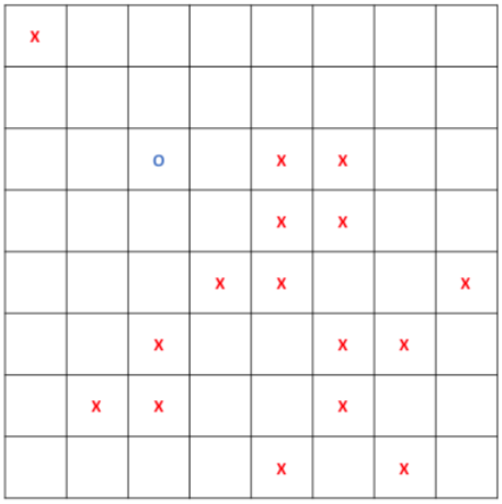
- Ecrivez un programme Python qui détermine le point le plus proche de P dans List_Coordinates en termes de distance euclidienne. Le programme doit renvoyer le point le plus proche sous la forme d'un tuple (deux dimensions).

Exemple :

Avec la liste List_Coordinates = [[8, 9], [10, -3], [2, 4], [-12, -13], [9, 9], [3,3]] et le point P = [1,1], le programme doit retourner (3, 3).

EXO5

Soir le plan donné ci-dessous :



- Dessinez un ***quadtree*** contenant uniquement les croix rouges.
- Modifiez cet arbre afin d'inclure une nouvelle valeur (symbolisé par un rond bleu) .