RENDU ACT /TP1

NOM: KACI
PRÉNOM:Sofiane

Les réponses:

Q1)

1.1) Seule la 3ième polyligne est bonne, car y a toujours un point en commun pour chaque deux segments

1.2)
Il faut que les coordonnées qui se suivent partagent tour a tour une abscisse ou une ordonnée commune, pour éviter les murs/toits qui ne sont plus parallèle a l'axe des abscisse/ordonnée.

Si possible si on évite les redondances.

1.3)
1.3)(1,0)(1,1)(5,1)(5,13)(9,13)(9,20)(12,20)(12,27)
(16,27)(16,3)(19,3)(19,0)(22,0)(22,3)(25,3)(25,0)

Q2)

La complexité de l'algorithme est en théta de n^2 O(n^2) puisque au début,

on ajoute tous les pixels de la fenêtre de visualisation et puis on balaye tous les pixels pour la ligne de toits.
Cette solution semble avoir le désaventage de ne pas s'intéresser uniquement aux pixels des immeubles.

Q3)

------ajouterImmeuble-----

Entrée:
m meuble;
n la ligne de toits;
si n vide:
n += (m[0],0);
n += (m[0],m[1]);
n += (m[2],m[1]);
n += (m[2],0);

return t;

sinon:
si n[0][0] < m[0]:

return n[0] + ajouterImmeuble(m,n[1:n]);

sinon si n[0][1] < m[1]:
n[0] = (m[0],t[1]);
n[1] = (m[0],m[1]);
i = 2;

sa compléxité ça devrait étre en O(n)
il ya m batiment et et n ligne de toit la compléxité final est O(n*m).

Q4)

la fusion:

Dans cette fonction la premiére approche a faire c'est d'ajouter tous les coordonnées des 2 listes des lignes de toi, j'ai utilisé la class Point pour créer mes point et les utiliser dans la class ligne_de toits, en suite dans la class Ligne_de_toits je stocke les point dans une liste , ensuite je vais faire appel a la fonction Tri() pour pouvoir trier les coordonnées de chaque liste

Une fois cette tache est términé la fonction Determinine_Next_ligne() sert a définir le prochain point qui doit étre utilisé parmi les point des ligne de toit, et si deux point sont egaux le resulat va prendre la valeur 0 .

```
Result_fusion.get(i).x=0;
Result_fusion.get(i).y=0;
```

La fonction Dispalay_resultat() pour afficher les résultat final de la fusion des deux lignes de toits. Et on affiche juste les point ou les résultat de (x,y)!= de 0 .

La fonction Display_SVG() qui prend en paramètre un ficher. A pour but de nous générer un fichier.svg qui contient un schéma qui correspond ou résultat de la fusion des deux ligne des toit.

La compléxité de la fusion est en O(n).

l'exécution de la class main donc va nous donner comme résultat la fusion des lignes des toits comme demandé dans le TP , et il va nous généré un fichier.svg qui contient un schéma correspond au résultat de la fusion.

j’ai utilisé 3 package (main,Toits,shema) dans le package schéma j’ai implémenté la class point que j’ai importé dans la class Ligne_de_toits, pour pouvoir utiliser les point dans ma liste.

NB:
Pour la question 5 je l’ai pas implémenté a cause du problème du temps.