Devoir #2

I. Tours de communication

Dans un réseau de communication téléphonique sans fil, les tours de communication doivent être placées dans des endroits stratégiques pour optimiser les transmissions. Chaque tour a une portée circulaire de rayon 5 km. Pour assurer aux utilisateurs une bonne réception de signal et pour éviter des coupures en cas de panne d'une des tours, il est préférable d'être dans une zone couverte par au moins deux tours.
Votre travail consiste à vérifier si des points dont on vous fournit les coordonnées sont effectivement couverts par au moins deux tours ou non. Un point est considéré dans la portée d'une tour si sa distance directe à la tour est inférieure à 5 km. On rappelle la formule pour calculer la distance entre deux points P_1 et P_2 de coordonnées respectives (x_1, y_1) et (x_2, y_2) :

$$d(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

En entrée, on vous donne : sur la 1ère ligne, un entier T indiquant le nombre de tours de communication. Puis sur les T lignes suivantes, les coordonnées (x, y) de chaque tour (deux nombres réels séparés par un espace sur chaque ligne). Puis un entier M indiquant le nombre des endroits à vérifier, puis sur les M lignes suivantes, les coordonnées de ces endroits (ici aussi deux nombres réels séparés par un espace sur chaque ligne).
En sortie, votre programme doit afficher les numéros des endroits (on suppose que les endroits sont numérotés 1, 2, 3, etc.) **qui ne sont pas correctement couverts**.

En commentaire avant le `main()`, indiquez la complexité algorithmique de votre programme en fonction de T et de M.

Exemples d'entrée :

```
3
5 3
2 8
6 9
3
5 6
8 11
30 9
```

Sortie correspondante :

```
2
3
```

En effet, le 2ème endroit $(8, 11)$ est seulement dans la portée de la tour #3 $(6, 9)$.
Le 3ème endroit $(30, 9)$ n'est dans la portée d'aucune des tours.

II. Origami

L'origami est l'art de plier des feuilles de papier pour créer des "objets" ou des "structures". Certaines structures particulièrement sophistiquées requièrent un grand nombre de plis. Le problème est que selon ses dimensions, une feuille de papier ne peut être pliée qu'un certain nombre de fois. Par exemple, une feuille de papier au format lettre standard ne peut pas être pliée plus que 6 ou 7 fois (dépendant de l'épaisseur du papier).

Le problème proposé ici est de calculer le nombre maximal de fois qu'une feuille de papier peut être pliée en deux, selon ses dimensions et son épaisseur initiales. Pour cela, on suppose que l'épaisseur totale d'une feuille pliée ne peut pas être plus qu'un tiers de chacune des dimensions. Par exemple pour une feuille au format lettre avec du papier usuel :

	Dimensions	**Épaisseur**
Dimensions originales :	216 mm × 280 mm	0,1 mm
Après un pli :	216 mm × 140 mm	0,2 mm
Après deux plis :	108 mm × 140 mm	0,4 mm
Après trois plis :	108 mm × 70 mm	0,8 mm
Après quatre plis :	54 mm × 70 mm	1,6 mm
Après six plis :	27 mm × 35 mm	6,4 mm
Après sept plis :	27 mm × 17,5 mm	12,8 mm

On voit que plier 6 fois est possible car si on multiplie l'épaisseur après 6 plis (6,4 mm) par 3, on obtient 19,2 mm, qui est plus petite que chacune des dimensions (27 mm et 35 mm). Par contre après 7 plis, si on multiplie l'épaisseur 12,8 mm par 3 on obtient 38,4 mm, qui est plus grande qu'au moins l'une des deux dimensions (en fait ici même les deux : 27 mm et 17,5 mm).

En entrée, on vous donne sur une ligne trois nombres qui peuvent être des valeurs réelles, indiquant respectivement la largeur, la longueur et l'épaisseur originales de la feuille de papier.

En sortie, votre programme doit afficher le nombre maximal de plis.

En commentaire avant le `main()`, indiquez la complexité algorithmique de votre programme en fonction de la largeur l et de la longueur L originales de la feuille.

Exemples d'entrée :

```
216 280 0.1
```

Sortie correspondante :

```
6
```