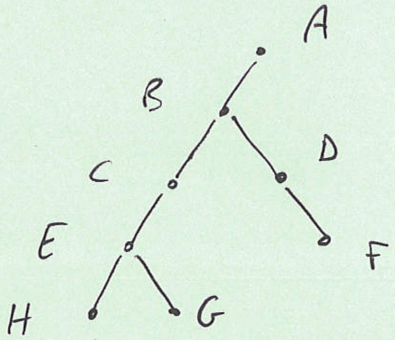
DS - Corrigé

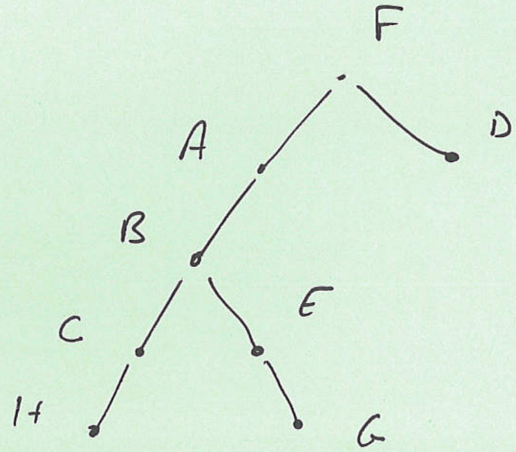
Exercice 1

1.



par exemple

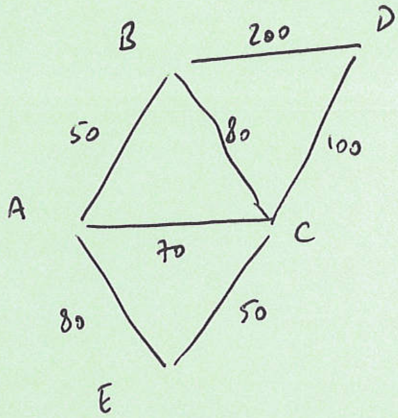
2.



par exemple

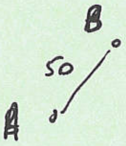
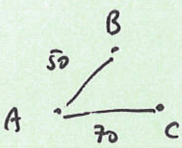
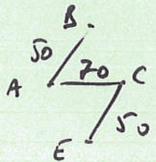
Exercice 2

Le problème revient à trouver un arbre couvrant de poids minimal dans le graphe suivant

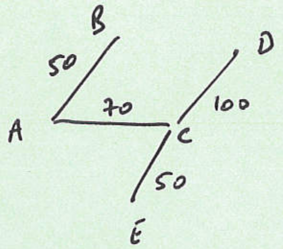


On peut appliquer l'algorithme de Kruskal ou celui de Prim.

Appliquons celui de Prim en débutant en A

Arbre courant	Arêtes candidates	Arête sélectionnée
A.	AB, 50 AC, 70 AE, 80	AB, 50
	AC, 70 AE, 80 BC, 80 BD, 200	AC, 70
	AE, 80 CE, 50 CD, 100 BD, 200	CE, 50
	CD, 100 BD, 100	CD, 100

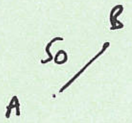
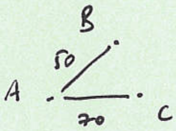
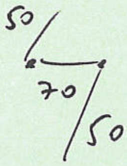
On obtient finalement le réseau



qui est de coût total 270.

2. ~~Lors~~ Lors du déroulement de l'algorithme de Prim, une ville ne représente un coût que si elle passe d'un sommet de degré 1 à un sommet de degré 2 dans le réseau existant.
On peut donc reprendre l'algorithme de Prim en adaptant le poids des arêtes candidates :

- si elles relient un sommet de degré 1 de la structure existante au reste du graphe, on leur ajoute le poids du sommet correspondant
- si elles relient un sommet de degrés ≥2 dans la structure existante au reste du graphe, on considère uniquement le poids de l'arête.

Arbre courant	Arêtes candidates	Arête sélectionnée
A	AB, 50 AC, 70 AE, 80	AB, 50
	AC, 70 BC, 80 + 20 AE, 80 BD, 200 + 20	AC, 70
	AE, 80 CE, 50 + 20 BD, 200 + 20 CD, 100 + 20	CE, 70
	BD, 200 + 20 CD, 100	CD, 100

le réseau est finalement le même, pour un coût de

290 .

Exercice 3

1. <u>G1</u> $V_1 = \{d\}$ est l'ensemble des sources de G1

$V_2 = \{a\}$ est l'ensemble des sources de $G1 \setminus V_1$

$V_3 = \{b, c, f\}$ ——— $G1 \setminus \{V_1, V_2\}$

$V_4 = \{e\}$ ——— $G1 \setminus \{V_1, V_2, V_3\}$

V_1, V_2, V_3, V_4 est bien une partition de G1 donc G1 est un graphe de niveau

<u>G2</u> $V_1 = \{r\}$ est l'ensemble des sources de G2

$G2 \setminus V_1$ n'a pas de sources. G2 n'est donc pas un graphe de niveau.

2. <u>G1</u> $L = d$

puis $L = d, a$

puis $L = d, a, b$ (par exemple, non-unicité ici)

puis $L = d, a, b, c$ "

puis $L = d, a, b, c, e$ "

puis $L = d, a, b, c, e, f$ "

<u>G2</u> $L = r$

et l'algorithme s'arrête.

3. <u>Supposons que l'algorithme renvoie une liste contenant tous les sommets</u>.

Alors V_1 est non vide puisque le premier sommet de L appartient à V_1.

Soit u le premier sommet de L qui n'appartient pas à V_1. Comme u est une source au moment où il est selectionné, il est une source dans $G \setminus V_1$. V_2 n'est donc pas vide.

On montre de même que si $G \setminus \{V_1 .. V_i\}$ est non vide, V_{i+1} est non vide puisque le premier sommet de L qui n'est pas dans $V_1 \cup \ldots \cup V_i$ est une source dans $G \setminus \{V_1 \ldots V_i\}$.

G est donc un graphe de niveau.

<u>Supposons que G est un graphe de niveau</u>

A tout moment de l'algorithme, il existe une source dans le graphe résiduel. En effet, soit i le plus petit entier tel que il reste au moins un sommet u de V_i dans le graphe résiduel.

~~[illegible]~~ le graphe résiduel est alors un sous-graphe de $G \setminus \{V_1, \ldots V_{i-1}\}$. Par conséquent, u est une source du graphe résiduel.

L'algorithme ne s'arrête donc pas tant qu'il n'a pas sélectionné tous les sommets.

4. Soit (u,v) une arête dans un graphe de niveau avec $u \in V_i$ et $v \in V_j$. Alors $j > i$ puisque j n'est pas une source tant que u n'a pas été supprimé.

L'existence d'un cycle orienté impliquerait l'existence d'au moins une arête allant d'un niveau grand vers un niveau plus petit. C'est donc impossible.

5. Supposons que tout sommet a un prédécesseur. On peut alors construire une suite infinie de sommet en prenant pour u_{i+1} un prédécesseur de u_i.

Or, le graphe a un nombre fini de sommet, donc au moins un sommet est répété dans cette suite. Ceci implique l'existence d'un cycle orienté.

6. Appliquons l'algorithme de la question 2 à un graphe sans cycle orienté. Tous ses sous-graphes sont également sans cycle orienté. Par conséquent, d'après la question 5, l'algorithme trouve une source tant que le graphe est non-vide, et renvoie donc tous les sommets.

D'après la question 3, G est donc un graphe de niveau.

7. D'après les questions 4 et 6, G est acyclique si et seulement si G est de niveau.

Or, d'après la question 3, G est de niveau si et seulement si l'algorithme 2 renvoie tous les sommets.

Faire tourner cet algorithme et tester si la longueur de L vaut n permet donc de répondre à la question.

L'algorithme de la question 2 est en $\mathcal{O}(n^2 m)$ (qu'on peut améliorer en ne le codant pas de manière naïve).