**ESERCIZIO 4**

(a) Si descriva l'algoritmo di Prim, se ne valuti la complessità computazionale e se ne dimostri la correttezza.

(b) Sia $G = (V, E)$ un grafo non orientato connesso con funzione peso $w : E \to \mathbb{N}$ e sia $E_M$ l'insieme degli archi di un suo albero di copertura minimo. Si supponga inoltre che $G$ contenga esattamente tre archi distinti $e_1$, $e_2$ ed $e_3$ aventi peso 0. Ovviamente vale $0 \leq |E_M \cap \{e_1, e_2, e_3\}| \leq 3$.
Per ciascun valore di $k = 0, 1, 2, 3$, si determini una proprietà $\mathcal{P}_k$ di $(G, e_1, e_2, e_3)$ tale che

$$|E_M \cap \{e_1, e_2, e_3\}| = k \quad \longleftrightarrow \quad \mathcal{P}_k(G, e_1, e_2, e_3) = \textbf{true}\,.$$

$|E_M \cap \{e_1, e_2, e_3\}| = 0 \rightarrow P_0(G; e_1, e_2, e_3) := \underline{\text{false}}$

$|E_M \cap \{e_1, e_2, e_3\}| = 1 \rightarrow P_1(G; e_1, e_2, e_3) := \underline{\text{false}}$

$|E_M \cap \{e_1, e_2, e_3\}| = 2 \rightarrow P_2(G; e_1, e_2, e_3) :=$ "$e_1, e_2, e_3$ formano un ciclo"

$|E_M \cap \{e_1, e_2, e_3\}| = 3 \rightarrow P_3(G; e_1, e_2, e_3) :=$ "$e_1, e_2, e_3$ non formano un ciclo"

## ESERCIZIO 2

**Definizione** Dato un grafo non orientato $G = (V, E)$ con funzione peso $w : E \to \mathbb{R}$ e dato un sottoinsieme $U$ di $V$, il sottografo di $(G, w)$ INDOTTO DA $U$ è il grafo pesato ottenuto rimuovendo da $G$ tutti i nodi non appartenenti a $U$ e tutti gli archi che toccano qualche nodo non appartenente a $U$.

Sia quindi $G = (V, E)$ un grafo connesso non orientato con funzione peso $w : E \to \mathbb{R}$.

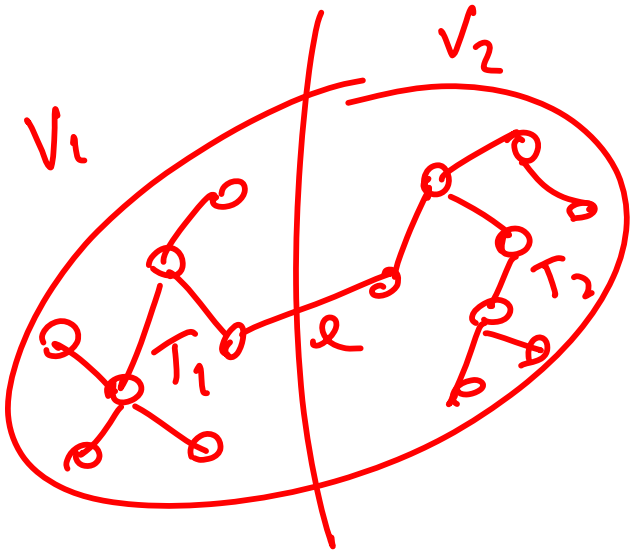
Per ciascuna delle seguenti asserzioni, stabilire se è necessariamente vera oppure no, motivando adeguatamente le risposte.

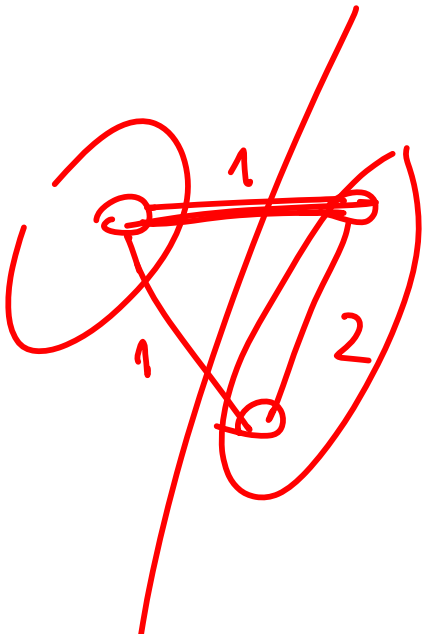
(A) Sia $\mathcal{T} = (V, T)$ un albero ricoprente minimo di $(G, w)$ e sia $e \in T$ un suo arco. Rimuovendo $e$ da $\mathcal{T}$ si formano due alberi $\mathcal{T}_1$ e $\mathcal{T}_2$ insistenti rispettivamente sugli insiemi di nodi $V_1$ e $V_2$. Allora,

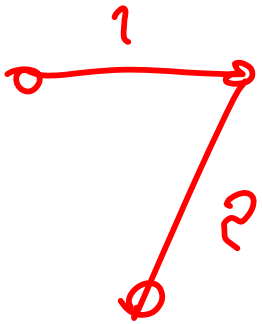
- $e$ è un arco di peso minimo che attraversa il taglio $(V_1, V_2)$ di $G$;
- $\mathcal{T}_i$ è un albero ricoprente minimo del sottografo di $(G, w)$ indotto da $V_i$, per $i = 1, 2$.

(B) Sia $(V_1, V_2)$ un taglio di $G$, sia $e$ un arco di peso minimo che attraversa il taglio $(V_1, V_2)$, e sia $\mathcal{T}_i = (V_i, T_i)$ un albero ricoprente minimo del sottografo di $(G, w)$ indotto da $V_i$, per $i = 1, 2$. Allora, il grafo $(V, T_1 \cup T_2 \cup \{e\})$ è un albero ricoprente minimo di $(G, w)$.

NO



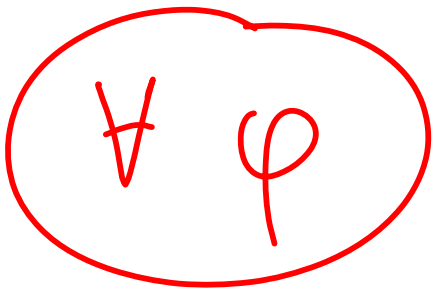



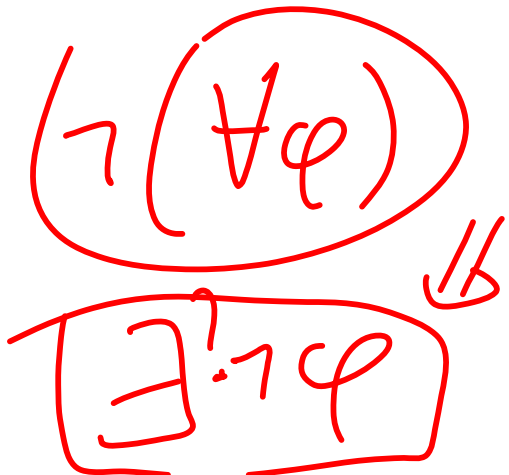
Dato un grafo $G = (V, E)$ non orientato e connesso, con funzione peso $w : E \to \mathbb{R}$, si consideri la seguente procedura ricorsiva:

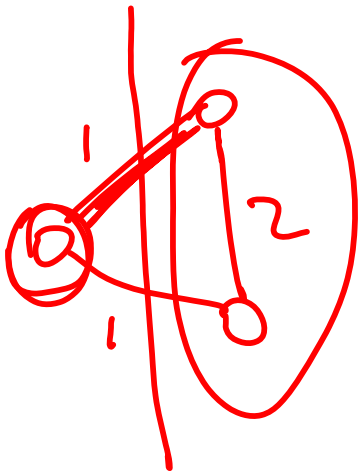
**procedure** Quick_Something$(V, E)$
  **if** $|V| \leq 1$ **then**
    **return** $\emptyset$
  **else**
    - sia $(V_1, V_2)$ un taglio di $(V, E)$ tale che i sottografi $(V_1, E_1)$ e $(V_2, E_2)$ risultino connessi, dove
      · $E_1$ := insieme degli archi in $E$ i cui estremi sono in $V_1$,
      · $E_2$ := insieme degli archi in $E$ i cui estremi sono in $V_2$;
    - sia $e$ un arco di peso minimo (rispetto alla funzione peso $w$) che attraversa il taglio $(V_1, V_2)$;
    $E_1$ := Quick_Something$(V_1, E_1)$;
    $E_2$ := Quick_Something$(V_2, E_2)$;
  **endif**
  **return** $E_1 \cup E_2 \cup \{e\}$;
**end** Quick_Something;

(a) Si dimostri che la procedura Quick_Something$(V, E)$ calcola un albero $\mathcal{T}$ di copertura di $(V, E)$.

(b) $\mathcal{T}$ è necessariamente un minimo albero di copertura per $(V, E)$ ?



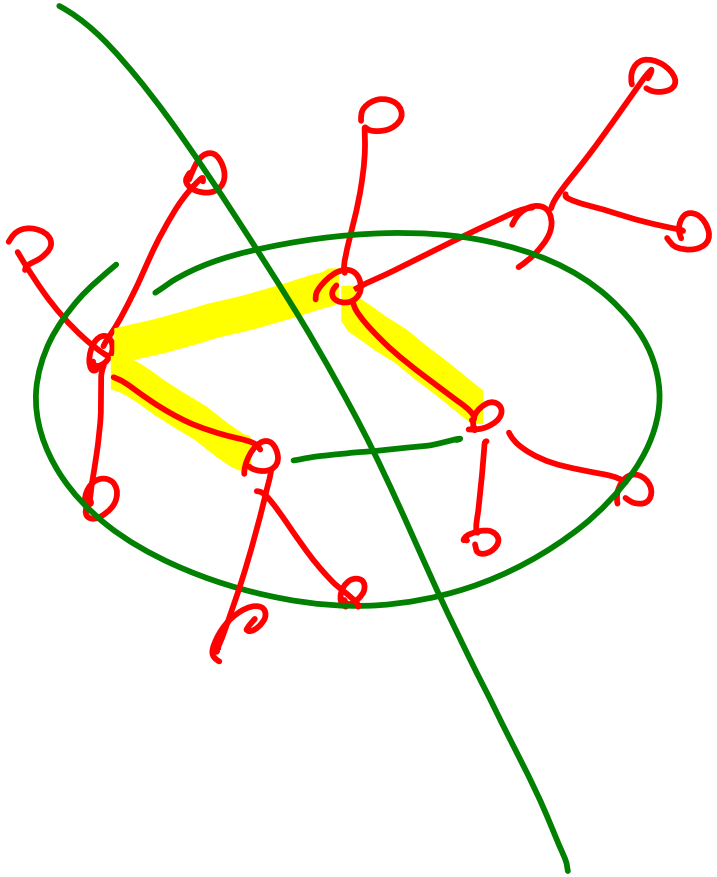



**EXERCISE 3**

Let $G = (V, E)$ be an undirected connected graph and let $w : E \longrightarrow \mathbb{R}$ be a weight function over $G$. Also, let $e \in E$ be an edge of $G$.

Describe an algorithm to establish whether $e$ is contained in some MST of $(G, w)$ and evaluate its computational complexity.



## ESERCIZIO 2

Sia $G = (V, E)$ un grafo non orientato connesso e con funzione peso $w : E \to \mathbb{R}$, e siano

$$(u_1, v_1),\ (u_2, v_2),\ \ldots,\ (u_k, v_k) \qquad (*)$$

tutti e soli gli archi di $G$ di peso minimo.

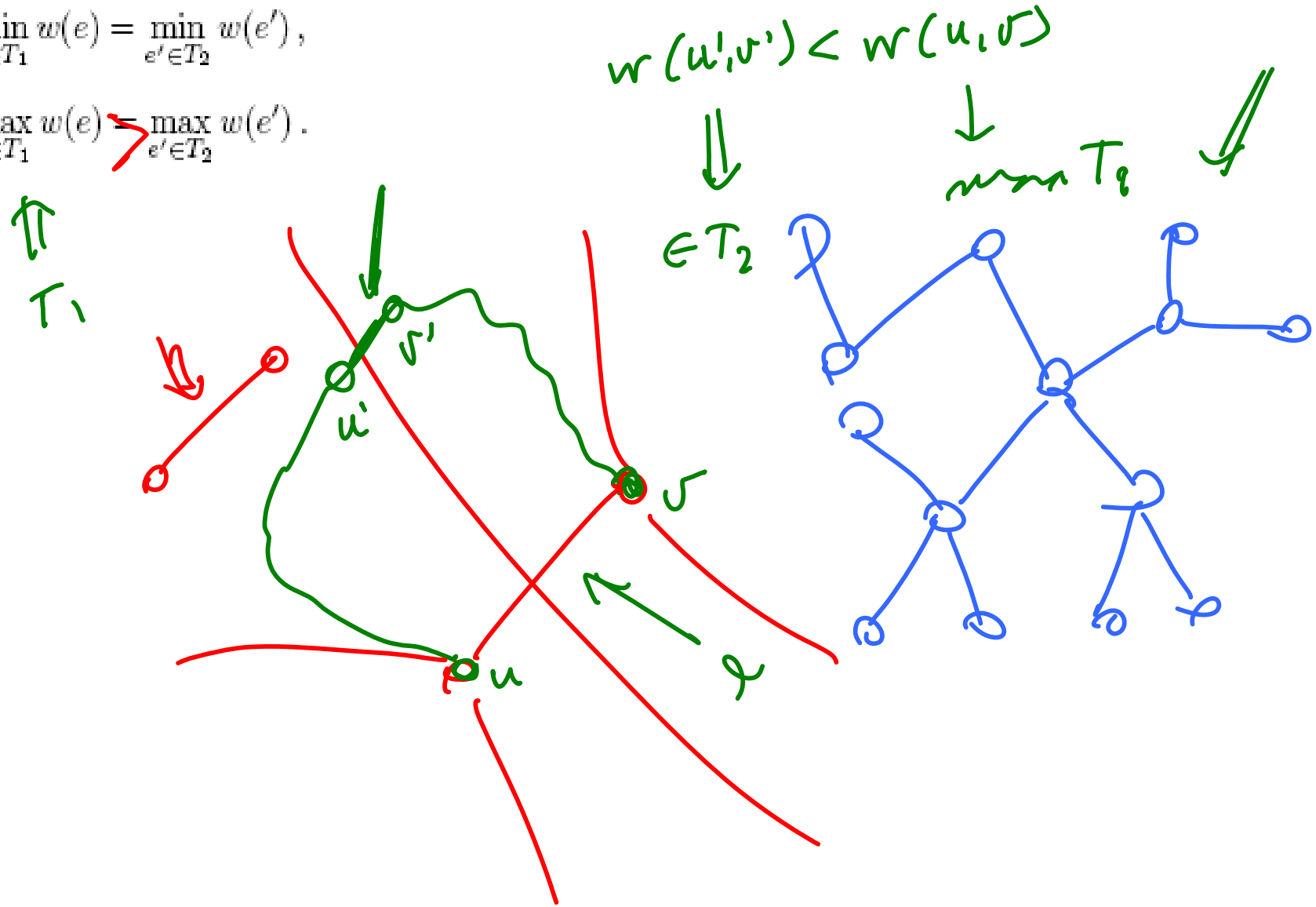
Si enunci e si dimostri una proprietà necessaria e sufficiente affinché esista un *minimum spanning tree* di $G$ che contenga tutti gli archi $(*)$. Quindi si proponga un algoritmo per verificare tale proprietà e se ne valuti la complessità computazionale.

## ESERCIZIO 3

Siano $T_1$ e $T_2$ due MST distinti di un dato grafo non-orientato connesso e pesato $(G, w)$. Si verifichi che:

(a) $\min_{e \in T_1} w(e) = \min_{e' \in T_2} w(e')$,

(b) $\max_{e \in T_1} w(e) = \max_{e' \in T_2} w(e')$.



**ESERCIZIO 3**

Si etichettino i seguenti punti del piano

A B C D E F G H I

$(1,3),\ (2,1),\ (2,6),\ (1,6),\ (6,6),\ (3,4),\ (3,6),\ (5,2),\ (5,7)$

9 nodi

13 archi

con le lettere da $A$ ad $I$, rispettivamente.

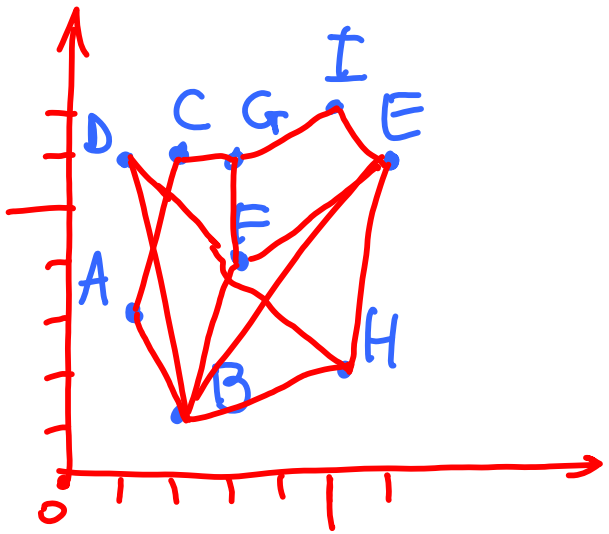
Prendendo come pesi le distanze tra gli estremi degli archi (distanza euclidea), si consideri il grafo pesato $(\mathcal{G}, w)$ avente i seguenti archi

$$(A,B),\ (A,C),\ (C,G),\ (F,G),\ (B,F),\ (B,E),\ (B,H),\ (B,D),\ (D,H),\ (E,I),\ (G,I),\ (E,F),\ (E,H)\,.$$

(a) Si illustri l'esecuzione degli algoritmi di Kruskal e di Prim sul grafo pesato $(\mathcal{G}, w)$, imponendo tra gli archi l'ordinamento lessicografico nei casi di parità tra i pesi (quindi, ad esempio, l'arco $(A,C)$ precede $(B,F)$ che, a sua volta, precede $(B,H)$).
In particolare, per quanto riguarda l'algoritmo di Prim, lo si esegua a partire dal nodo $A$.

*Nota:* per semplificare i calcoli, si utilizzino i quadrati delle distanze al posto delle distanze stesse.

(b) Si spieghi brevemente, ma in maniera rigorosa, perchè pur utilizzando i quadrati delle distanze si ottengono MST validi per il grafo originale $(\mathcal{G}, w)$, con le distanze euclidee.




AB → 5

AC → 10

CG → 1

FG → 4

EI → 2

BF → 10

BE → 41

BH → 10

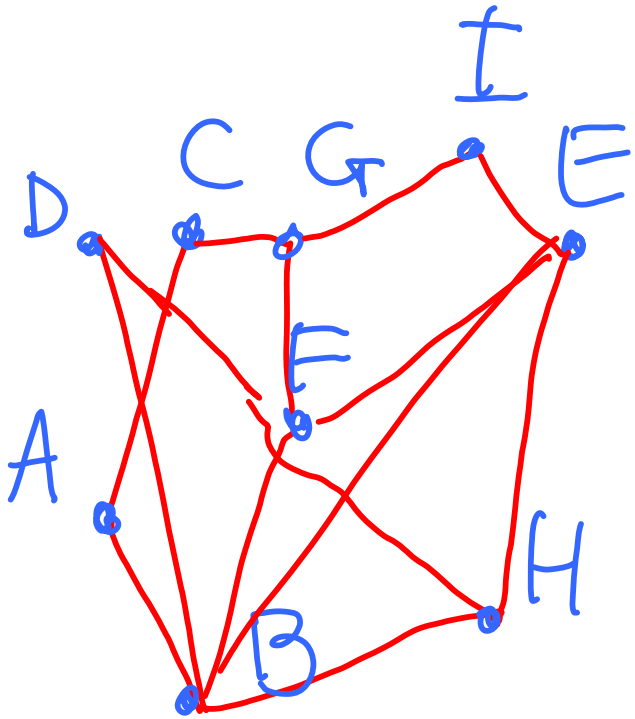
BD → 26

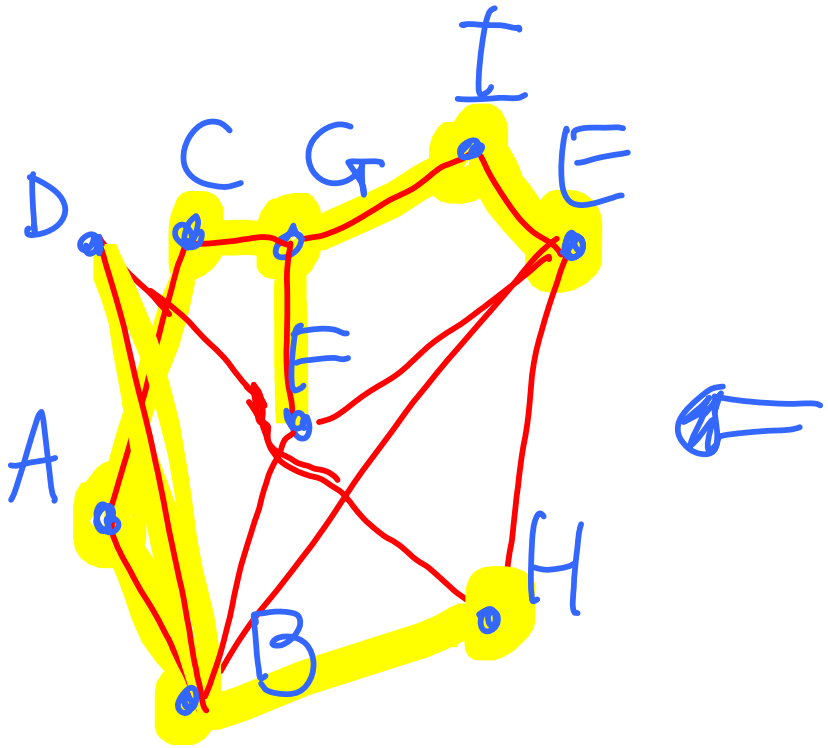
DH → 32

GI → 5

EF → 13

EH → 17



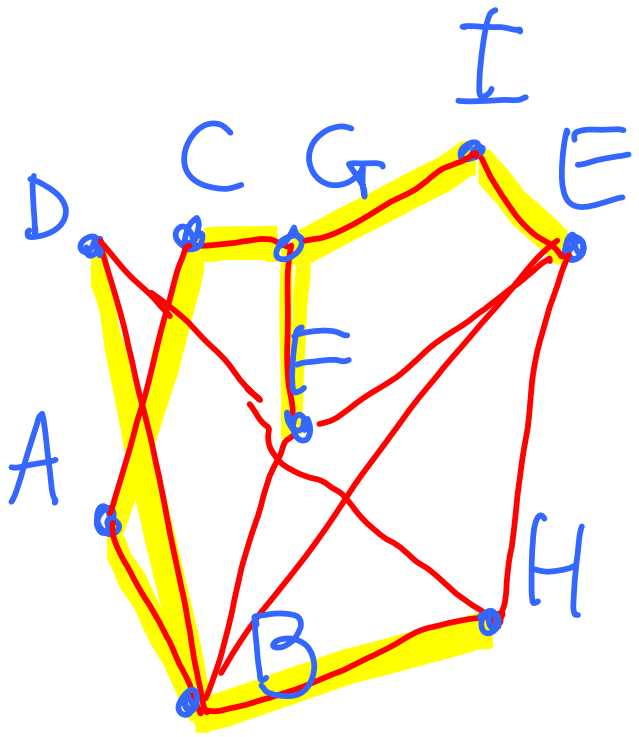

CG EI FG AB GI AC BF BH EF EH

BD DH BE

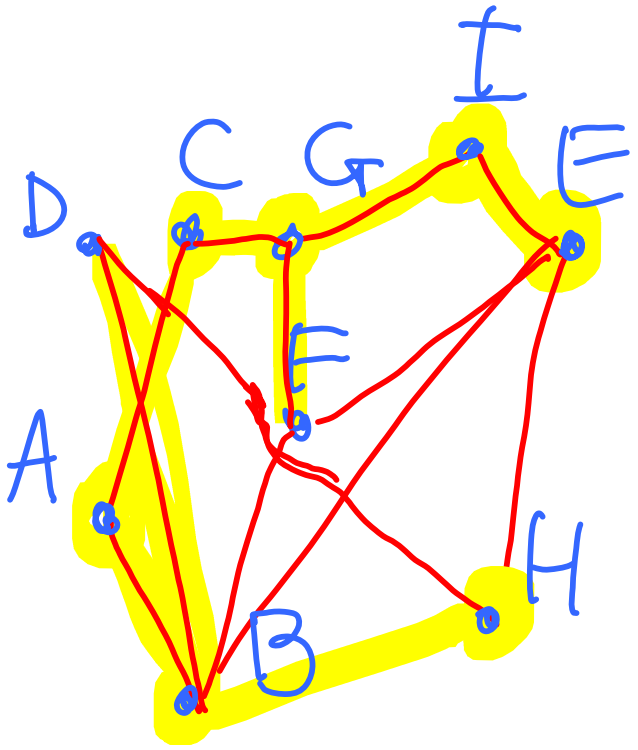
⟹

| A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|
| 0 | 5 | 10 | 26 | 2 | 4 | 1 | 10 | 5 |

| Pred | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| | - | A | A | B | I | G | C | B | G |




- KRUSKAL




PRIM

**ESERCIZIO 4**

Sia dato un grafo connesso non orientato $G = (V, E)$ con funzione peso $w : E \longrightarrow \mathbb{R}$. Sia inoltre $F$ un sottoinsieme aciclico di $E$. Un *$F$-spanning tree* di $G$ è uno spanning tree di $G$ il cui insieme di archi contiene $F$.

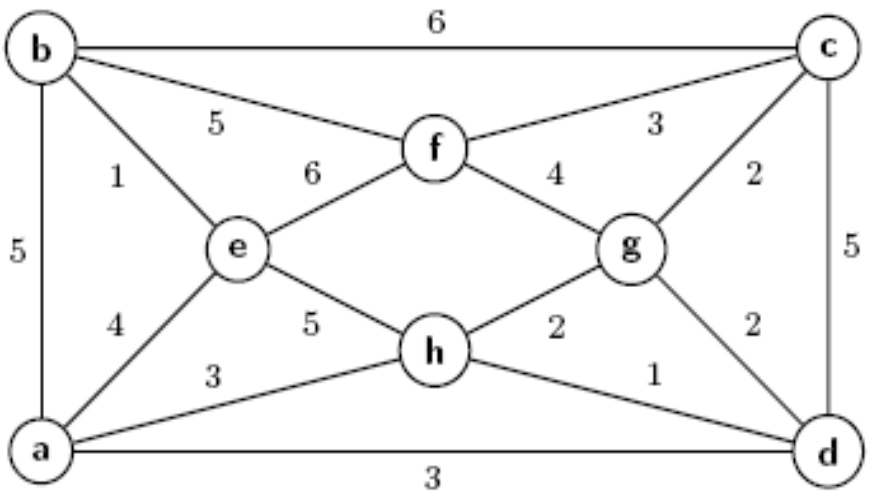
Si descriva un algoritmo efficiente (valutandone la complessità e dimostrandone la correttezza) che calcoli un minimo $F$-spanning tree di $G$.

**ESERCIZIO 2 (Alberi ricoprenti minimi)**

(a) Si descriva l'algoritmo di Kruskal e lo si applichi al grafo a lato.

(b) Ci sono altri alberi ricoprenti minimi?

(c) Sia $G = (V, E)$ un grafo non orientato, pesato e connesso, e sia $w : E \to \mathbb{N}$ una funzione peso per $G$. Si supponga che $G$ contenga *esattamente* 7 archi di peso unitario e nessun arco di peso strettamente inferiore ad 1. Quanti archi di peso unitario dovrà necessariamente contenere un albero ricoprente minimo per $G$? Perché?



**ESERCIZIO 4** (Minimum spanning trees)

(a) Si descriva l'algoritmo di Kruskal (anche mediante il suo pseudo-codice) e lo si applichi al grafo a lato.

(b) Si descrivano i cosiddetti "passi blu" e "passi rossi" negli algoritmi per il calcolo del *minimum spanning tree* nei grafi non-orientati, connessi e pesati. Quindi si enunci l'*invariante del colore* e lo si dimostri limitatamente ai passi rossi.