

```

SET kruskal(EDGE[] A, int n, int m)
    // EDGE[]: vettore di archi

(1) SET  $T \leftarrow \text{Set}$  // insieme (inizialmente vuoto) che conterrà gli archi dell'albero minimo
    MFSET  $M \leftarrow \text{Mfset}(n)$  // insieme disgiunto grande

    // ordino per peso crescente
(2) { ordina  $A[1][m]$  in modo che  $A[1].\text{peso} \leq \dots \leq A[m].\text{peso}$  }

    int  $c \leftarrow 0$  // quanti archi ho aggiunto
    int  $i \leftarrow 1$  // quale arco sto guardando
(3) while  $c < n - 1$  and  $i \leq m$  do // Termina quando l'albero è costruito
    //  $c < n - 1$ : ho raggiunto tutti gli archi necessari per fare un albero
    //  $i \leq m$ : ho esaurito tutti gli archi da guardare (per controllo)
    if  $M.\text{find}(A[i].u) \neq M.\text{find}(A[i].v)$  then // non fanno parte dello stesso albero
         $M.\text{merge}(A[i].u, A[i].v)$  // unisco gli insiemi disgiunti
         $T.\text{insert}(A[i])$  // inserisco l'arco all'albero
         $c \leftarrow c + 1$  // ho aggiunto un altro arco
     $i \leftarrow i + 1$  // guardo il prossimo arco
    return  $T$  // Ritorna l'albero di copertura minimo

```