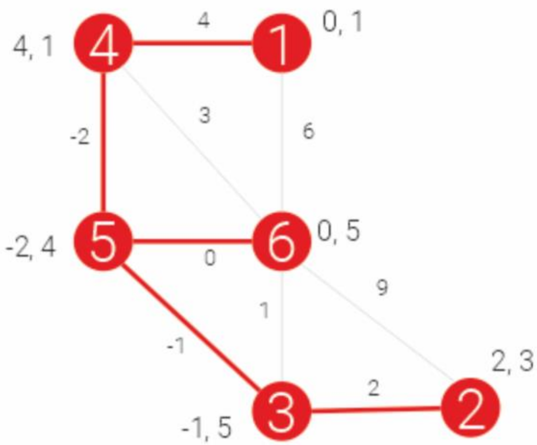
Implementando el algoritmo de Prim

Algoritmo de Prim, 1957

```
T=∅
Para todo nodo i de V hacer coste[i] = infinito
M={1}
coste[1]=0
pred[1]=1
Mientras en T no haya n-1 aristas hacer
        sea u el último nodo que entró en M
        para todo j adyacente a u en V-M hacer
                si coste[j] es peor que w(u, j) entonces
                        coste[j]=w(u,j)
                        pred[j] = u
        sea u = arg min coste [j] para todo j en V-M
        M = M ∪ {u}
        T = T ∪ {(u, pred[u])}
```



Algoritmo de Prim

```
Para todo nodo i de V hacer coste[i] = ∞;

T= ∅; M={1}; coste[1]=0; pred[1]=1

Mientras en T no haya n-1 aristas hacer
        sea u el último nodo que entró en M
        para todo j adyacente a u en V-M hacer
           si coste[j] es peor que w(u, j) entonces
                   coste[j]=w(u,j)
                   pred[j] = u
        sea u = arg min coste [j] para todo j en V-M
        M = M ∪ {u}
        T = T ∪ {(u, pred[u])]
```

	1	2	3	4	5	6
coste	0	2	-1	4	-2	0
pred	1	3	5	1	4	5

$$\Gamma_1 = \{6, 4\}$$
$$\Gamma_2 = \{6, 3\}$$
$$\Gamma_3 = \{5, 6, 2\}$$
$$\Gamma_4 = \{1, 6, 5\}$$
$$\Gamma_5 = \{4, 6, 3\}$$
$$\Gamma_6 = \{1, 4, 5, 3, 2\}$$

Revisamos la adyacencia de u=6, {1, 4, 5, 3, 2};
el nodo 1 está en M
el nodo 4 está en M
el nodo 5 está en M
el nodo 3 está en M
el nodo 2, comparamos coste desde 6, **no actualizamos**
el nodo fuera de M con menor coste es u=2: actualizamos

En T tenemos n-1 aristas: FIN

El MST está formado por las aristas (ver pred): (2, 3), (3, 5), (1, 4), (4, 5), (5, 6)
y tiene un coste (ver coste): 3