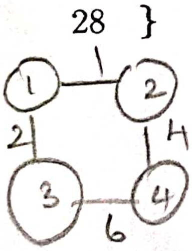
```
Algorithm Prim(E, cost, n, t)
// E is the set of edges in G. cost[1 : n, 1 : n] is the cost
// adjacency matrix of an n vertex graph such that cost[i, j] is
// either a positive real number or ∞ if no edge (i, j) exists.
// A minimum spanning tree is computed and stored as a set of
// edges in the array t[1 : n − 1, 1 : 2]. (t[i, 1], t[i, 2]) is an edge in
// the minimum-cost spanning tree. The final cost is returned.
{
    Let (k, l) be an edge of minimum cost in E;
    mincost := cost[k, l];
    t[1, 1] := k; t[1, 2] := l;
    for i := 1 to n do // Initialize near.
        if (cost[i, l] < cost[i, k]) then near[i] := l;
        else near[i] := k;
    near[k] := near[l] := 0;
    for i := 2 to n − 1 do
    { // Find n − 2 additional edges for t.
        Let j be an index such that near[j] ≠ 0 and
        cost[j, near[j]] is minimum;
        t[i, 1] := j; t[i, 2] := near[j];
        mincost := mincost + cost[j, near[j]];
        near[j] := 0;
        for k := 1 to n do // Update near[ ].
            if ((near[k] ≠ 0) and (cost[k, near[k]] > cost[k, j]))
                then near[k] := j;
    }
    return mincost;
}
```

$\boxed{mincost = 3+4 = 7}$

$n = 4$



Edges, $E = \{(1,2), (1,3), (2,4), (3,4)\}$

cost $\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & \infty & 1 & 2 & \infty \\ 2 & 1 & \infty & \infty & 4 \\ 3 & 2 & \infty & \infty & 6 \\ 4 & \infty & 4 & 6 & \infty \end{array}$

$t[1:3, 1:2]$ → final edges in tree

$\begin{array}{c|cc} & 1 & 2 \\ \hline 1 & 1 & 2 \\ 2 & 3 & 1 \\ 3 & 4 & 2 \end{array}$

near

1	2	3	4
~~2~~ 0	~~1~~ 0	1	2

$(k, l) = (1,2)$, mincost = 1

i = 1 to 3

cost (1, 2) < cost (1, 1) ⇒ 1 < ∞

near [1] = 2

i = 2 cost (2,2) < cost (2,1) ⇒ ∞ < 1

near [2] = 1

i = 3

cost (3,2) < cost (3,1)

∞ < 2

near [3] = 1

near[1] = 0, near[2] = 0

i = 2 to 2

j = 3 [near[3] != 0]

cost [3,1] = 2

min cost = 1 + 2 = 3

K = 1 to 3,

near [4]

cost [4,2] > cos [4,3]

near [4] = 2