Minimum Spanning Tree (MST).

PRIM (V, E, s) — s: starting vertex

1. $d[v] \leftarrow \infty \quad \forall v \in V$
2. $d[s] \leftarrow 0$
3. $PQ \leftarrow$ INSERT all $v \in V$ w/ key $= d[v]$ — PQ is a MIN PQ
4. while PQ is NOT EMPTY
5. $\quad u \leftarrow PQ.extractMin()$
6. $\quad$ for each v adjacent to u
7. $\quad\quad$ if $d[v] > w(u,v)$ && $v \in PQ$
8. $\quad\quad\quad PQ.decreaseKey(v, w(u,v))$
9. $\quad\quad\quad parent[v] \leftarrow u$

line 1 takes V time

line 3 does ~~EXTRACT~~ V INSERTS.

line 5 does V · EXTRACTMINs

line 8 can happen E times @ most so E DECREASEKeys.

$O(V + V*\text{INSERT} + V*\text{EXTRACTMIN} + E*\text{DECREASEKEY})$.

If use a binary min heap implementation of PQ.

INSERT	$O(\lg V)$
EXTR. MIN	$O(\lg V)$
DECREASEKEY	$O(\lg V)$

so PRIM's is

$O(V + V\lg V + V\lg V + E\lg V)$

$= O(E\lg V)$ b/c $E \geq V-1$ b/c CONNECTED G.

for binary min heap implementation

If use an unsorted array implementation of PQ

INSERT	$O(1)$
EXTR MIN	$O(V)$
DECR. KEY	$O(1)$ b/c we have direct access to it.

So PRIM's is

$O(V + V + V^2 + E)$

$= O(V^2)$ b/c $E \leq V^2$

If we a fibonacci heap

INSERT	$O(1)$
EXTR MIN	$O(\lg V)$
DECR. KEY	$O(1)$

} w/ amortized analysis.

So PRIM's is

$$O(V + V + V\lg V + E)$$
$$= O(V\lg V + E)$$

	BIN. MIN. HEAP	UNSORTED ARRAY	FIB. HEAP
PRIM's RT.	$O(E\lg V)$	$O(V^2)$	$O(V\lg V + E)$

for SPARSE graphs prefer binary min heap.

for. DENSE graphs prefer unsorted array

for MEDIUM sized graphs prefer fibonacci heap.