Prim (1957) [Jarník (1930), Dijkstra (1959)]

Start: \( A = \) set consisting of one (arbitrary) vertex of \( V \)

\( T = \) empty edge set

One iteration:

\[
\begin{align*}
A & \quad \text{and} \quad \mathcal{Q} = V \setminus A \\
\end{align*}
\]

* take edge \( \{u,v\} \) of minimum weight such that \( u \in A, v \in \mathcal{Q} \).
* add the edge \( \{u,v\} \) to \( T \).
* move \( v \) from \( \mathcal{Q} \) to \( A \).

Repeat until \( A = V \) (i.e., \( \mathcal{Q} = \emptyset \)).
Prim:
\[ r = \text{arbitrary vertex of } V; \]
\[ A = \{ r \}; \]
\[ T = \emptyset; \]
while \( A \neq V \):
    find edge \( \{u,v\} \) of minimum weight such that \( u \in A, v \notin V \setminus A \);
    \[ A = A \cup \{v\}; \]
    \[ T = T \cup \{\{u,v\}\}; \]

How to find the edge \( \{u,v\} \): by brute force in \( O(m) \) time.

Total running time = \( O(mn) \),
where \( n = |V|, m = |E| \).
To improve the running time: maintain extra information.

\[ Q = V \setminus A \]

For each vertex \( y \) in \( Q \):

- \( \text{minweight}(y) = \text{minimum weight of any edge between } y \text{ and a vertex of } A \).
- \( \text{closest}(y) = \text{vertex } x \text{ in } A \text{ for which } wt(x, y) = \text{minweight}(y) \).

Observe: Shortest edge \( \{u,v\} \) connecting \( A \) and \( Q \) has weight \( \min \{ \text{minweight}(y) : y \in Q \} \).
What happens if we move $v$ from $Q$ to $A$:

$Q = V \setminus A$

update $\text{minweight}(w)$ and $\text{closest}(w)$ for $w = x, y, z$. 
Prim:
r = arbitrary vertex of V;
A = \{r\};
T = \\emptyset;
for each vertex \( y \neq r \): \( \text{minweight}(y) = \infty \); \( \text{closest}(y) = \text{nil} \);
for each edge \( \{r, y\} \): \( \text{minweight}(y) = \text{wt}(r, y) \);
\( \text{closest}(y) = r \);
\( Q = V \setminus \{r\} \); \( k = 1 \);
while \( k \neq n \) : // \( k = |A| \)
\( v = \text{vertex of } Q \text{ for which } \text{minweight}(v) \text{ is minimum} \);
\( u = \text{closest}(v) \);
\( A = A \cup \{v3\}; Q = Q \setminus \{v\}; T = T \cup \{\{u, v3\}\}; \)
\( k = k + 1 \);
for each edge \( \{v, y\} \):
    if \( y \in Q \) and \( \text{wt}(v, y) < \text{minweight}(y) \):
        \( \text{minweight}(y) = \text{wt}(v, y) \);
        \( \text{closest}(y) = v \);
Store the vertices of $\mathcal{Q}$ in a min-heap, for each $v$, the key of $v$ is $\text{minweight}(v)$.

Store $T$ in a list.

With each vertex of $V$: store one bit indicating whether the vertex belongs to $A$ or to $\mathcal{Q}$.

**Running time:**

Up to the while-loop: $O(n)$  
(this includes the time to build the heap)

One iteration of the while-loop:

- extract-min: $O(\log n)$ time
- $\leq \text{degree}(v)$ many decrease-key operations:
  - $O(\text{degree}(v) \cdot \log n)$ time.
Total time for the while-loop:

\[ O\left(\sum_{v \in V} \text{degree}(v) \cdot \log n\right) = O(m \log n). \]

= 2m

Conclusion: Prim's algorithm computes MST in \( O(m \log n) \) time.