III Single-source shortest path

most well known algorithm is **Dijkstra's Algorithm**

main idea is to keep the current set of nodes whose shortest paths from $s$ (source node) are known and extend the set using the link that produces the shortest path to a new node using existing paths plus one link (similar to Prim's Alg. but using distance instead of weight of 1 link)

\[
\text{Dijkstra (int s)} \$
\begin{align*}
&\text{Found} = \emptyset, V_s \notin \emptyset; \\
&\text{dist} [s] = 0; \\
&\text{for all } V_i \neq V_s \\
&\quad \text{if } (V_s, V_i) \in E \text{ dist} [i] = \text{weight}(V_s, V_i); \\
&\quad \text{else dist} [i] = \infty; \\
&\text{while } (\text{Found} \neq V) \{
&\quad \text{min-dist} = \min_{V_i \notin \text{Found}} \text{dist} [i]; \\
&\quad \text{let } (V_i, V_j) \text{ be last edge on path connecting } \text{Vs to } V_i \text{ with distance min-dist}
&\quad \text{Found} = \text{Found} \cup \{V_i\}; \\
&\quad \text{for all } V_k \notin \text{Found} \{
&\quad \quad \text{if } (V_i, V_k) \in E
&\quad \quad \quad \text{dist} [k] = \min(\text{dist} [k], \text{dist} [i] + \text{weight}(V_i, V_k)); \\
&\quad \text{\ commentary }
&\quad \text{\ end for}
&\quad \text{\ comment}
&\quad \text{\ end while}
&\text{\ comment}
&\text{\ end Dijkstra}
\end{align*}
\]