17.3.5
The basic algorithm: Find the $i$th closest vertex
A Basic Strategy

Explore vertices in increasing order of distance from $s$:
(For simplicity assume that nodes are at different distances from $s$ and that no edge has zero length)

Initialize for each node $v$, $\text{dist}(s,v) = \infty$
Initialize $X = \{s\}$,
for $i = 2$ to $|V|$ do
  (* Invariant: $X$ contains the $i-1$ closest nodes to $s$ *)
  Among nodes in $V - X$, find the node $v$ that is the $i$th closest to $s$
  Update $\text{dist}(s,v)$
  $X = X \cup \{v\}$

How can we implement the step in the for loop?
A Basic Strategy

Explore vertices in increasing order of distance from s:
(For simplicity assume that nodes are at different distances from s and that no edge has zero length)

Initialize for each node \( v \), \( \text{dist}(s, v) = \infty \)
Initialize \( X = \{ s \} \),
\[ \text{for } i = 2 \text{ to } |V| \text{ do} \]
(* Invariant: \( X \) contains the \( i - 1 \) closest nodes to \( s \) * )
Among nodes in \( V - X \), find the node \( v \) that is the \( i \)th closest to \( s \)
Update \( \text{dist}(s, v) \)
\( X = X \cup \{ v \} \)

How can we implement the step in the for loop?
Finding the $i$th closest node

1. $X$ contains the $i − 1$ closest nodes to $s$
2. Want to find the $i$th closest node from $V − X$.

What do we know about the $i$th closest node?

Claim

Let $P$ be a shortest path from $s$ to $v$ where $v$ is the $i$th closest node. Then, all intermediate nodes in $P$ belong to $X$.

Proof.

If $P$ had an intermediate node $u$ not in $X$ then $u$ will be closer to $s$ than $v$. Implies $v$ is not the $i$th closest node to $s$ - recall that $X$ already has the $i − 1$ closest nodes.
Finding the $i$th closest node

1. $X$ contains the $i - 1$ closest nodes to $s$
2. Want to find the $i$th closest node from $V - X$.

What do we know about the $i$th closest node?

Claim

Let $P$ be a shortest path from $s$ to $v$ where $v$ is the $i$th closest node. Then, all intermediate nodes in $P$ belong to $X$.

Proof.

If $P$ had an intermediate node $u$ not in $X$ then $u$ will be closer to $s$ than $v$. Implies $v$ is not the $i$th closest node to $s$ - recall that $X$ already has the $i - 1$ closest nodes.
Finding the \( i \)th closest node

1. \( X \) contains the \( i - 1 \) closest nodes to \( s \)
2. Want to find the \( i \)th closest node from \( V - X \).

What do we know about the \( i \)th closest node?

**Claim**

Let \( P \) be a shortest path from \( s \) to \( v \) where \( v \) is the \( i \)th closest node. Then, all intermediate nodes in \( P \) belong to \( X \).

**Proof.**

If \( P \) had an intermediate node \( u \) not in \( X \) then \( u \) will be closer to \( s \) than \( v \). Implies \( v \) is not the \( i \)th closest node to \( s \) - recall that \( X \) already has the \( i - 1 \) closest nodes. \( \square \)
Finding the $i$th closest node repeatedly

An example
Finding the $i$th closest node repeatedly

An example
Finding the $i$th closest node repeatedly

An example
Finding the $i$th closest node repeatedly

An example
Finding the $i$th closest node repeatedly

An example
Finding the $i$th closest node repeatedly

An example
Finding the \(i^{th}\) closest node repeatedly

An example
Finding the $i$th closest node repeatedly

An example
Finding the $i$th closest node repeatedly

An example
Finding the $i$th closest node

Corollary

The $i$th closest node is adjacent to $X$. 
Summary

Proved that the basic algorithm is (intuitively) correct...
...but is missing details
...and how to implement efficiently?
THE END

...  

(for now)