18.4.3
Floyd-Warshall algorithm
Floyd-Warshall Algorithm
for All-Pairs Shortest Paths

\[
d(i, j, k) = \min \left\{ d(i, j, k - 1), d(i, k, k - 1) + d(k, j, k - 1) \right\}
\]

for \( i = 1 \) to \( n \) do
    for \( j = 1 \) to \( n \) do
        \( d(i, j, 0) = \ell(i, j) \) \hspace{1cm} (* \( \ell(i, j) = \infty \) if \( (i, j) \notin E \), 0 if \( i = j \) *)

for \( k = 1 \) to \( n \) do
    for \( i = 1 \) to \( n \) do
        for \( j = 1 \) to \( n \) do
            \[ d(i, j, k) = \min \left\{ d(i, j, k - 1), d(i, k, k - 1) + d(k, j, k - 1) \right\} \]

for \( i = 1 \) to \( n \) do
    if \( (\text{dist}(i, i, n) < 0) \) then
        Output \( \exists \) negative cycle in \( G \)

Running Time: \( \Theta(n^3) \).
Space: \( \Theta(n^3) \).
Correctness:
via induction and recursive definition
Floyd-Warshall Algorithm
for All-Pairs Shortest Paths

\[ d(i, j, k) = \min \left\{ d(i, j, k - 1), \right. \]
\[ \left. d(i, k, k - 1) + d(k, j, k - 1) \right\} \]

for \( i = 1 \) to \( n \) do
  for \( j = 1 \) to \( n \) do
    \( d(i, j, 0) = \ell(i, j) \)
    (* \( \ell(i, j) = \infty \) if \( (i, j) \notin E \), \( 0 \) if \( i = j \) *)

for \( k = 1 \) to \( n \) do
  for \( i = 1 \) to \( n \) do
    for \( j = 1 \) to \( n \) do
      \( d(i, j, k) = \min \left\{ d(i, j, k - 1), \right. \]
      \[ \left. d(i, k, k - 1) + d(k, j, k - 1) \right\} \]

for \( i = 1 \) to \( n \) do
  if \( (\text{dist}(i, i, n) < 0) \) then
    Output \( \exists \) negative cycle in \( G \)

Running Time: \( \Theta(n^3) \).
Space: \( \Theta(n^3) \).
Correctness:
via induction and recursive definition
**Floyd-Warshall Algorithm**

for All-Pairs Shortest Paths

\[
\begin{align*}
d(i, j, k) &= \min \left\{ d(i, j, k - 1), \\
&\quad d(i, k, k - 1) + d(k, j, k - 1) \right\}
\end{align*}
\]

for \( i = 1 \) to \( n \) do

for \( j = 1 \) to \( n \) do

\[ d(i, j, 0) = \ell(i, j) \]

(* \( \ell(i, j) = \infty \) if \( (i, j) \notin E \), \( 0 \) if \( i = j \) *)

for \( k = 1 \) to \( n \) do

for \( i = 1 \) to \( n \) do

for \( j = 1 \) to \( n \) do

\[ d(i, j, k) = \min \left\{ d(i, j, k - 1), \\
&\quad d(i, k, k - 1) + d(k, j, k - 1) \right\} \]

for \( i = 1 \) to \( n \) do

if \( (\text{dist}(i, i, n) < 0) \) then

Output \( \exists \) negative cycle in \( G \)

**Running Time:** \( \Theta(n^3) \).

**Space:** \( \Theta(n^3) \).

**Correctness:**

via induction and recursive definition
Floyd-Warshall Algorithm

for All-Pairs Shortest Paths

\[
d(i, j, k) = \min \left\{ \begin{array}{ll}
d(i, j, k - 1) \\
d(i, k, k - 1) + d(k, j, k - 1)
\end{array} \right. 
\]

for \(i = 1\) to \(n\) do
  for \(j = 1\) to \(n\) do
    \(d(i, j, 0) = \ell(i, j)\)
    (* \(\ell(i, j) = \infty\) if \((i, j) \notin E\), \(0\) if \(i = j\) *)

for \(k = 1\) to \(n\) do
  for \(i = 1\) to \(n\) do
    for \(j = 1\) to \(n\) do
      \(d(i, j, k) = \min \left\{ \begin{array}{ll}
d(i, j, k - 1), \\
d(i, k, k - 1) + d(k, j, k - 1)
\end{array} \right. \)

for \(i = 1\) to \(n\) do
  if \((\text{dist}(i, i, n) < 0)\) then
    Output \(\exists\) negative cycle in \(G\)

Running Time: \(\Theta(n^3)\).
Space: \(\Theta(n^3)\).

Correctness:
via induction and recursive definition
Floyd-Warshall Algorithm: Finding the Paths

**Question:** Can we find the paths in addition to the distances?

1. Create a $n \times n$ array `Next` that stores the next vertex on shortest path for each pair of vertices.
2. With array `Next`, for any pair of given vertices $i, j$ can compute a shortest path in $O(n)$ time.
Floyd-Warshall Algorithm: Finding the Paths

**Question:** Can we find the paths in addition to the distances?

1. Create a $n \times n$ array $\text{Next}$ that stores the next vertex on shortest path for each pair of vertices
2. With array $\text{Next}$, for any pair of given vertices $i, j$ can compute a shortest path in $O(n)$ time.
Floyd-Warshall Algorithm
Finding the Paths

for $i = 1$ to $n$ do
  for $j = 1$ to $n$ do
    $d(i, j, 0) = \ell(i, j)$
    (* $\ell(i, j) = \infty$ if $(i, j)$ not edge, 0 if $i = j$ *)
    $Next(i, j) = -1$
  Next ($i$, $j$) = $k$
for $k = 1$ to $n$ do
  for $i = 1$ to $n$ do
    for $j = 1$ to $n$ do
      if ($d(i, j, k - 1) > d(i, k, k - 1) + d(k, j, k - 1)$) then
        $d(i, j, k) = d(i, k, k - 1) + d(k, j, k - 1)$
        $Next(i, j) = k$
  for $i = 1$ to $n$ do
    if ($d(i, i, n) < 0$) then
      Output that there is a negative length cycle in $G$

Exercise: Given $Next$ array and any two vertices $i, j$ describe an $O(n)$ algorithm to find a $i-j$ shortest path.
THE END
...
(for now)