Input: Description of an $n$-vertex graph via an $n \times n$ array $w$, such that $w[u,v]$ contains weight of edge $(u,v)$. (Weight is infinity if no edge and 0 for $w[v,v]$.) Starting vertex $s$

Output: Array $A$ containing…?

Goals:
- Design a dynamic programming algorithm for shortest path
- Array dimension?

Example:

Question?

A) $O(n)$ B) $O(n^2)$ C) $O(n^3)$ D) Read more about graph

Where are we using $A$ containing $w$?
- for $i = 1$ to $n$
  - for $v = 1$ to $n$
    - if $i = v$
      - $A[i][v] = 0$
    - else
      - $A[i][v] = \min\{A[i-1][v], A[i][v-1], A[i-1][v-1]\}$

Use adjacency list
- $w[1][2], w[2][3], w[3][1]$
- $w[1][2], w[1][3], w[2][1], w[2][4], w[3][2], w[3][5], w[4][2], w[5][3]$
- $w[1][2], w[1][3], w[1][4], w[2][1], w[2][3], w[3][1], w[3][4], w[4][2], w[4][3], w[5][3], w[5][4]$

Example:

Problem:

A) $O(n)$ B) $O(n^2)$ C) $O(n^3)$ D) Read more about graph

Where are we using $A$ containing $w$?
- for $i = 1$ to $n$
  - for $v = 1$ to $n$
    - if $i = v$
      - $A[i][v] = 0$
    - else
      - $A[i][v] = \min\{A[i-1][v], A[i][v-1], A[i-1][v-1]\}$

Use adjacency list
- $w[1][2], w[1][3], w[3][1]$
- $w[1][2], w[1][3], w[1][4], w[2][1], w[2][3], w[3][1], w[3][4], w[4][2], w[4][3], w[5][3], w[5][4]$

What is the runtime?

Why is this algorithm so fast?