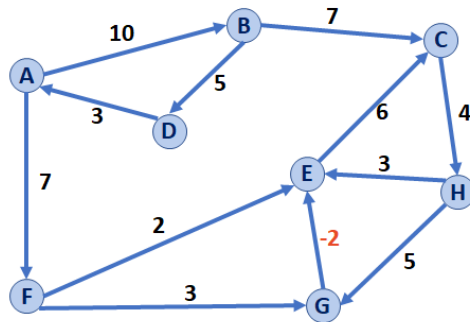**From Friday:**
- Graphs with a negative-weight **cycle** have no finite shortest path. *(We can always take the cycle one more time to get an even shorter path!)*
- Graphs with a negative-weight **edge without a negative-weight cycle** DO have a finite shortest path!
- Does Dijkstra's algorithm find it?

Dijkstra: What if we have a minimum-weight edge, without having a negative-weight cycle?

...what assumption does Dijkstra's algorithm make?

**Q:** Can we transform the graph by adding **+k** to every edge?
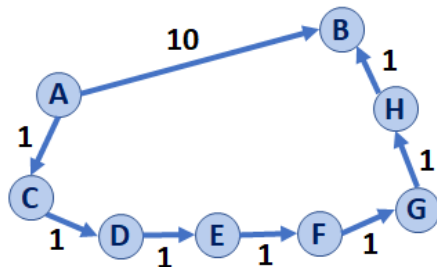
Dijkstra: What is the running time?

Floyd-Warshall Algorithm
Floyd-Warshall's Algorithm is an alternative to Dijkstra in the presence of negative-weight edges (but <u>not</u> negative weight cycles).

Algorithm Design:
- **Goal:** Find the shortest path from vertex **u** to **v**.

- **Setup:** Create an n×n matrix that maintains the best known path between every pair of vertices:
  - Initialize (u, u) to 0.
  - Initialize all edges present on the graph to their edge weight.
  - Initialize all other edges to +infinity.
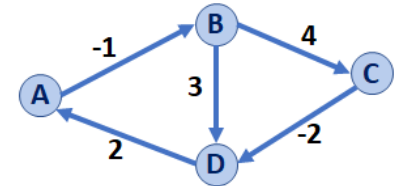
	A	**B**	**C**	**D**
A				
B				
C				
D				

- For every vertex **k**, consider which of the following are shorter:
  - path(u, v)   - or -
  - path(u, **k**) + path(**k**, v)

Big Idea: _____
- Store intermediate results to improve build towards an optimal solution.
- Example application of memorization and **dynamic programming (DP)** – more in CS 374!

Running Time:

```
        Pseudocode for Floyd-Warshall's Algorithm
1   FloydWarshall(G):
2     Input: G, Graph;
3     Output: d, an adjacency matrix of distances between
4                All vertex pairs
5
6     Let d be an adj. matrix (2d array) initialized to +inf
7     foreach (Vertex v : G):
8       d[v][v] = 0
9     foreach (Edge (u, v) : G):
10      d[u][v] = cost(u, v)
11
12    foreach (Vertex u : G):
13      foreach (Vertex v : G):
14        foreach (Vertex k : G):
15          if d[u, v] > d[u, k] + d[k, v]:
16            d[u, v] = d[u, k] + d[k, v]
17
18    return d
```

Overview of Graphs:

Implementations
- Edge List
- Adjacency Matrix
- Adjacency List

Traversals
- Breadth First
- Depth First

Minimum Spanning Tree (MST)
- Kruskal's Algorithm
- Prim's Algorithm

Shortest Path
- Dijkstra's Algorithm *(Single Source)*
- Floyd-Warshall's Algorithm *(All Pairs)*

Maximum Flow
- Ford-Fulkerson (DFS paths) Algorithm
- Edmonds-Karp (BFS paths) Algorithm

...and this is just the beginning. The journey continues to CS 374!

End of Semester :(

CS 225 Final Exam
- The final exam is a 3 hour CBTF exam, is a cumulative exam, and has the format of theory exam + programming exam
  - Expect ~25-30 theory questions
  - Expect ~2-3 programming problems
- Exam begins on Thursday (December 13th), last day of office hours is December 12th *(no office hours once finals begin)*
- **In-lecture review w/ TAs** on Wednesday (December 12th)

"Pre-Final" Grade Update
- As soon as possible after the MP7 deadline, I'll provide a "Pre-Final" grade update in Compass 2g with all grades except for your final exam.

End of Semester Grade Review
- Did we miss something that impacts your final grade? I want to be absolutely sure you get the grade you earned!
- After final grades are posted, I will provide a Google Sheet that allows you to submit a **Grade Review** if you believe the grade review will change your final letter grade.
  - You will have the chance to justify why you received an incorrect grade and how it impacts your letter grade in the course.
  - Instructions on Piazza at the same time as that the final grades are posted.

CS 225 – Things To Be Doing:
1. MP7 due tonight (December 10); *two-day grace period applies!*   2. Final Exam starts Thursday, December 14