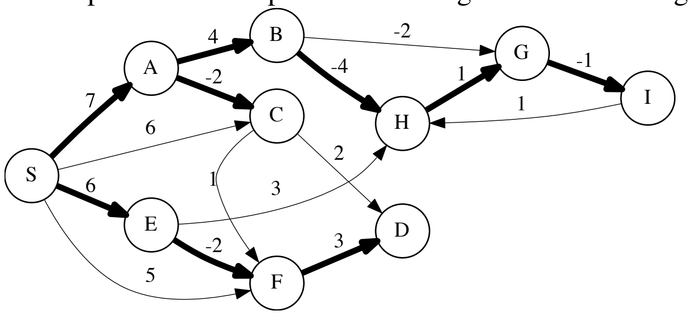
Problem 1. *DPV Problem 4.2.*
Your table depends on what order you iterate through the edges, but after the last iteration you should have:

Vertex	S	A	B	C	D	E	F	G	H	I
Dist	0	7	11	5	7	6	4	8	7	7
Prev	–	S	A	A	F	S	E	H	B	G

The tree of shortest paths is made up of the thick edges in the following.



Problem 2. *PG Problem 434.* Consider how the set S is built. In step 1, we add the start node to S and our subgraph of paths has no edges. In each set after the first, we add a new node to S and our subgraph of paths gets an edge to this new node. The algorithm stops when we have $S = V$. Notice that we have put exactly $|V| - 1$ edges into our subgraph of paths. Hence, this subgraph must be a tree which spans the original graph.

Problem 3. *DPV Problem 4.13.*
(a) Remove all edges of length $> L$ from the graph and then do a DFS from s and see if t is reachable. This takes $O(|V| + |E|)$ time.
(b) Let $e_1 < e_2 < \cdots < e_k$ be the lengths of the edges of E sorted into increasing order; also let $e_0 = 0$. Put $e_0, \ldots, e_k$ in array **edgeLen**$[0..k]$. Use part (a) and **edgeLen** to do a binary search to find an i such that: the trip is feasible if $L = e_i$, but not feasible if $L = e_{i-1}$. Note that $k \leq |E| \leq |V|^2$, so $\log k \leq \log |E| \leq 2 \log |V|$. So the total cost of this algorithm if $O(|V| + |E|) * O(\log k)$ which is $O((|V| + |E|) \log |V|)$.

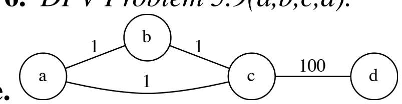
Problem 4. *DPV Problem 4.21.*
(a) The key idea is to use $*$ and max in place of $+$ and min in Bellman-Ford.
procedure exchange(G, r, s, t)
  for each $u \in V$ **do** $\{\ rate[u] \leftarrow 0;\ prev[u] \leftarrow nil\ \}$
  $rate[s] \leftarrow 1$;
  repeat $|V| - 1$ many times
    for each $(u, v) \in E$ **do**
      if $rate[v] < rate[u] * r[u, v]$
        then $\{\ rate[v] \leftarrow rate[u] * r[u, v];\ prev[v] \leftarrow u\ \}$
  return $(rate, prev)$

(b) The anomaly is analogous to a negative weight cyclce. So we just run the algorithm one more iteration and see if there is any change in the *rate* array—just like in the original Bellman-Ford algorithm.

Problem 5. *DPV Problem 5.2(a).*

Set S	A	B	C	D	E	F	G	H
$\emptyset$	0/nil	∞/nil	∞/nil	∞/nil	∞/nil	∞/nil	∞/nil	∞/nil
A		1/A	∞/nil	∞/nil	4/A	8/A	∞/nil	∞/nil
A,B			2/B	∞/nil	4/A	6/B	6/B	∞/nil
A,B,C				3/C	4/A	6/B	2/C	∞/nil
A,B,C,G				1/G	4/A	1/G		1/G
A,B,C,D,G					4/A	1/G		1/G
A,B,C,D,F,G					4/A			1/G
A,B,C,D,F,G,H					4/A			

Problem 6. *DPV Problem 5.9(a,b,c,d).*

(a) False. 

(b) True (Correction!). Suppose by way of contradiction we had a minimal spanning tree. T, that includes an edge (u, v) that is the unique heaviest edge of some cycle. Remove (u, v) from T and let S be the connected component of u in $T - \{(u, v)\}$. Since (u, v) is the heaviest edge on a cycle, there must be a lighter edge crossing the S and $V - S$ partition, that this results in a MST with a cost less than that of T, contradiction. So there cannot be any such MST T.
(c) True. Suppose T is a MST that does not include e. Then $T \cup \{e\}$ must include a cycle. Let e' be an edge on this cycle which is $\neq e$ and let $T' = (T \cup \{e\}) - \{e'\}$. Then T' also must be a spanning tree and $cost(T') \leq cost(T)$. So T' is a MST that includes e.
(d) True. Suppose by way of contradiction that T is a MST that does not include e. Then $T \cup \{e\}$ must include a cycle. Let e' be an edge on this cycle which is $\neq e$ and let $T' = (T \cup \{e\}) - \{e'\}$. Then T' also must be a spanning tree. Since e is the smallest length edge, it follows that $cost(T') < cost(T)$, a contradiction since T is supposed be of minimal cost.