2. After every update that changes a distance, the new distance is shown.

```
from D     A  B  C  D  E  F
distance  -- -- --  0 -- --
      D A : to A now 7
      D B : to B now -1
      D F : to F now 3
      F A : to A now 4
      F C : to C now 7
      F E : to E now 9
distance   4 -1  7  0  9  3
      A C : to C now 6
      B E : to E now -3
      B F : to F now 2
      E C : to C now -2
      F A : to A now 3
distance   3 -1 -2  0 -3  2
      C A : to A now 0
distance   0 -1 -2  0 -3  2
      A F : to F now 1
      F D : to D now -1
distance   0 -1 -2 -1 -3  1
      D B : to B now -2
distance   0 -2 -2 -1 -3  1
      B E : to E now -4
      E C : to C now -3
distance   0 -2 -3 -1 -4  1
parent     C  D  E  F  B  A
negative weight cycle
```

Many students gave distances from A. Here is that data.

```
from A     A  B  C  D  E  F
distance   0 -- -- -- -- --
      A C : to C now 2
      A D : to D now 7
      A F : to F now 1
      C E : to E now 3
      D B : to B now 6
      F B : to B now 4
      F D : to D now -1
distance   0  4  2 -1  3  1
      B E : to E now 2
      D B : to B now -2
distance   0 -2  2 -1  2  1
      B E : to E now -4
      E C : to C now -3
distance   0 -2 -3 -1 -4  1
      C A : to A now -1
```

```
distance  -1 -2 -3 -1 -4  1
      A F : to F now 0
      F D : to D now -2
distance  -1 -2 -3 -2 -4  0
      D B : to B now -3
distance  -1 -3 -3 -2 -4  0
parent     C  D  E  F  B  A
negative weight cycle
```

3. At this point, `dist[v]` is the weight of a shortest path from s to v. Why?

   Well, after `update(s,a)` is called,

   `dist[a]` $\leq$ `wt(s,a)`, so

   after `update(a,q)` is then called,

   `dist[q]` $\leq$ `dist[a] + wt(a,q)` $\leq$ `wt(s,a) + wt(a,q)`, so

   after `update(q,v)` is then called,

   `dist[q]` $\leq$ `dist[q] + wt(q,v)` $\leq$

   `wt(s,a) + wt(a,q) + wt(q,v)`.

   We are told that (s, a, q, v) is a shortest path from s to v. So, at this time, since `dist[v]` $\leq$ `wt(s,a) + wt(a,q) + wt(q,v)`, which is the weight of a shortest path from s to v, and since `dist[v]` can never be less than the weight of a shortest path to v, it must be equal to the weight of a shortest path to v.

4.
```
<   for _ in range(len(G)-1):
<     updates = 0
---
>   iterations = 0
>   for _ in range(len(G)):
>     updates, iterations = 0, iterations+1
---
>   if iterations == len(G): print "negative weight cycle"
```

```
from S     A  B  S
distance  -- --  0
distance   1 --  0
distance   0  2 -1
distance  -1  1 -2
parent     S  A  B
negative weight cycle
```

5. (i) first in preorder (ii) the root of a tree is printed last in preorder, so the nodes before `F` in postorder are the other nodes of the first tree

```
   F       E        I
  / \      |       / \
 J   D     K      H   A
          / \         |
         B   G        M
         |   |        |
         L   N        C
```

6. (i) If it is not a source, then there is some arc directed into it. But all arcs go forward in the ordering, contradiction.

   (ii) There are many correct answers, because at any point in the order, any source in the remaining subgraph can go next. E.g. here the first vertex can be any one of {E,H,I}, since these are the three sources. One answer is `I A H B D E C G F J`. Another answer is `E H I A B D C G F J`.

   (iii) In postorder, a vertex always appears before its parent. So the reverse of the postorder traversal sequence is a topsort of an acyclic graph. This takes linear time, so $\Theta(n^2)$ time for an n-vertex graph represented by an adjacency matrix.

7. (i) There can be more than one answer, depending on how you pick edges when there is more than one with the same weight. Here, I give the answer if you always pick the lowest weight edge which comes first in alphabetic order.

   (i) The edges with weight 1, AD and CF, are picked first. The edges with weight 2, BJ and CI, are picked next. The edges with weight 3, AB and AJ, are considered next: only one of these can be picked, so AB. The edges with weight 4 are considered next (AI, BF, CH); AI and CH are picked (BF would create a cycle). Continuing in this fashion, the final list of picked edges, in order, is

   `AD CF BJ CI AB AI CH JK HG CE` .

   (ii) Again, there can be more than one answer, depending on which edge you pick whenever there is more than one choice (here, first in alphabetic order), and which vertex you start from (here, A).

   `AD AB BJ AI CI CF CH HG JK CE` .