✖

Directed Graphs

Back to Week 1

✓ 3/3 points earned (100%)

Quiz passed!

✓ 1 / 1
points

1.

```
(seed = 489125)
Consider the adjacency-lists representation of a digraph with 8 vertices and 13 edges:

    A:  E
    B:  A G
    C:  B
    D:  C G
    E:  F
    F:  A B
    G:  H F C
    H:  D


Here is a graphical representation of the same digraph:

    (A)<------------(B)<------------(C)<------------(D)
     |^              ^\              ^              /^
     | \             | \             |             / |
     |  \            |  \            |            /  |
     |   \           |   \           |           /   |
     |    \          |    \          |          /    |
     |     \         |     \         |         /     |
     |      \        |      \        |        /      |
     |       \       |       \       |       /       |
     |        \      |        \      |      /        |
     |         \     |         \     |     /         |
     |          \    |          \    |    /          |
     |           \   |           \   |   /           |
     |            \  |            \  |  /            |
     |             \ |             \ | /             |
     v              \|              v|v              |
    (E)------------>(F)<------------(G)------------>(H)

```

Run breadth-first search (using the adjacency-lists representation), starting from vertex A. Give the sequence in which the vertices are dequeued from the FIFO queue.

Your answer should be a sequence of uppercase letters, starting with A.

A E F B G H C D

Correct Response

```
The correct answer is: A E F B G H C D

Here is a trace of the breadth-first search:

enqueue A
dequeue A
   enqueue E
dequeue E
   enqueue F
dequeue F
   check A
   enqueue B
dequeue B
   check A
   enqueue G
dequeue G
   enqueue H
   check F
   enqueue C
dequeue H
   enqueue D
dequeue C
   check B
dequeue D
   check C
   check G


Here are the shortest paths and distances:

A to A (0):  A
A to B (3):  A->E->F->B
A to C (5):  A->E->F->B->G->C
A to D (6):  A->E->F->B->G->H->D
A to E (1):  A->E
A to F (2):  A->E->F
A to G (4):  A->E->F->B->G
A to H (5):  A->E->F->B->G->H
```

✓ 1 / 1 points

2.

```
(seed = 979780)
Consider the adjacency-lists representation of a DAG with 8 vertices and 13 edges:

    A:
    B:  C A F E G
    C:  D G
    D:  G
    E:  A F
    F:  G
    G:
    H:  G D


Here is a graphical representation of the same DAG:

    (A)<------------(B)------------>(C)------------>(D)
     ^              /|\              |              /^
     |             / | \             |             / |
     |            /  |  \            |            /  |
     |           /   |   \           |           /   |
     |          /    |    \          |          /    |
     |         /     |     \         |         /     |
     |        /      |      \        |        /      |
     |       /       |       \       |       /       |
     |      /        |        \      |      /        |
     |     /         |         \     |     /         |
     |    /          |          \    |    /          |
     |   /           |           \   |   /           |
     |  /            |            \  |  /            |
     | /             |             \ | /             |
     |v              v              vvv              |
    (E)------------>(F)------------>(G)<------------(H)

```

Give the topological order of the vertices that results from the DFS-based topological sort algorithm. As usual, perform the first DFS from vertex A.

Your answer should be a sequence of 8 uppercase letters.

H B E F C D G A

Correct Response

```
The correct answer is: H B E F C D G A

Here is a trace of the depth-first search:

dfs(A)
A done
dfs(B)
  dfs(C)
    dfs(D)
      dfs(G)
      G done
    D done
    check G
  C done
  check A
  dfs(F)
    check G
  F done
  dfs(E)
    check A
    check F
  E done
  check G
B done
check C
check D
check E
check F
check G
dfs(H)
  check G
  check D
H done


The postorder is the order in which the vertices are done. The reverse postorder
provides a topological order.
```

✓ 1 / 1 points

3.

```
(seed = 584559)
Consider the adjacency-lists representation of a digraph G with 10 vertices and 17 edges:

    A:  B
    B:  F
    C:  H D B
    D:  E
    E:  J
    F:  A G
    G:  B H
    H:  I B D
    I:  E D
    J:  I


Here is a graphical representation of the same digraph G:

    (A)------------>(B)<------------(C)------------>(D)------------>(E)
     ^              /^^              |              ^^              ^|
     |             / | \             |             / |             / |
     |            /  |  \            |            /  |            /  |
     |           /   |   \           |           /   |           /   |
     |          /    |    \          |          /    |          /    |
     |         /     |     \         |         /     |         /     |
     |        /      |      \        |        /      |        /      |
     |       /       |       \       |       /       |       /       |
     |      /        |        \      |      /        |      /        |
     |     /         |         \     |     /         |     /         |
     |    /          |          \    |    /          |    /          |
     |   /           |           \   |   /           |   /           |
     |  /            |            \  |  /            |  /            |
     | /             |             \ | /             | /             |
     |v              |              \v/              |/              v
    (F)------------>(G)------------>(H)------------>(I)<------------(J)



Compute the strongly-connected components of the digraph using the Kosaraju-Sharir algorithm.
Assume that the first depth-first search of Kosaraju-Sharir computes the reverse postorder of G^R:

    D I J E A F B H C G

Give the sequence of the 10 integers in the id[] array for the vertices A through J.

       v    A  B  C  D  E  F  G  H  I  J
    ------------------------------------
    id[v]
```

1 1 2 0 0 1 1 1 0 0

Correct Response

```
The correct answer is: 1 1 2 0 0 1 1 1 0 0


   v    A  B  C  D  E  F  G  H  I  J
------------------------------------
id[v]   1  1  2  0  0  1  1  1  0  0

The second depth-first search considers the vertices in the following order:

    D I J E A F B H C G

Here is a trace of the second depth-first search:


strong component 0
------------------
dfs(D)
  dfs(E)
    dfs(J)
      dfs(I)
        check E
        check D
      I done
    J done
  E done
D done
------------------

check I
check J
check E

strong component 1
------------------
dfs(A)
  dfs(B)
    dfs(F)
      check A
      dfs(G)
        check B
        dfs(H)
          check I
          check B
          check D
        H done
      G done
    F done
  B done
A done
------------------

check F
check B
check H

strong component 2
------------------
dfs(C)
  check H
  check D
  check B
C done
------------------

check G
```