COMP9312 Advanced Graph Traversal

Outline

- Topological Sorting
- Tracking Unvisited Vertices

Exercise 1: Topological Sorting

- Implement topological sorting

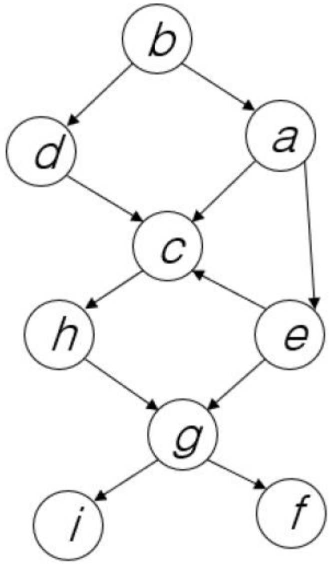
  - Initialize an array of in-degree
  - Create a queue and initialize it with all vertices that have in-degree as 0.

  While the queue is not empty:

    - Pop a vertex from the queue
    - Decrement the in-degree of each neighbor
    - Those neighbors whose in-degree was decremented to zero are pushed onto queue.

  The size of the queue is at most $O(|V|)$

Exercise 1: Topological Sorting



Possible topological sorting sequences:

- $b\,a\,d\,e\,c\,h\,g\,i\,f$
- $b\,a\,d\,e\,c\,h\,g\,f\,i$
- $b\,a\,e\,d\,c\,h\,g\,i\,f$
- $b\,a\,e\,d\,c\,h\,g\,f\,i$
- $b\,d\,a\,e\,c\,h\,g\,i\,f$
- $b\,d\,a\,e\,c\,h\,g\,f\,i$

Now, you have 10 minutes to implement ex1.

Exercise 2: Tracking Unvisited Vertices

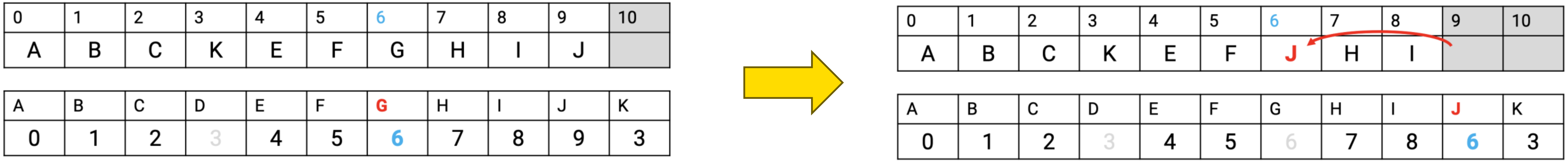
- Create two arrays:
  - unvisited: contain the unvisited vertices
  - loc_in_unvisited: contain the location of vertex in the array

0	1	2	3	4	5	6	7	8	9	10
A	B	C	D	E	F	G	H	I	J	K

A	B	C	D	E	F	G	H	I	J	K
0	1	2	3	4	5	6	7	8	9	10

Exercise 2: Tracking Unvisited Vertices

- Visit vertex which is in the middle position of unvisited array
    - Example:
      Suppose we visit G in entry 6,
      we copy the last unvisited vertex into this location and update the location array for this value



0	1	2	3	4	5	6	7	8	9	10
A	B	C	K	E	F	G	H	I	J	

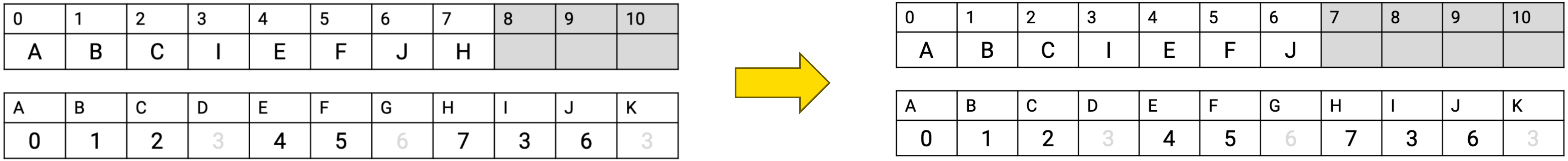
A	B	C	D	E	F	G	H	I	J	K
0	1	2	3	4	5	6	7	8	9	3

0	1	2	3	4	5	6	7	8	9	10
A	B	C	K	E	F	J	H	I		

A	B	C	D	E	F	G	H	I	J	K
0	1	2	3	4	5	6	7	8	6	3

Exercise 2: Tracking Unvisited Vertices

- Visit vertex which is in the last position of unvisited array
  - Example:
    Suppose we visit H in entry 7,
    we simply return the last entry of the unvisited array and return it.



0	1	2	3	4	5	6	7	8	9	10
A	B	C	I	E	F	J	H			

A	B	C	D	E	F	G	H	I	J	K
0	1	2	3	4	5	6	7	3	6	3

→

0	1	2	3	4	5	6	7	8	9	10
A	B	C	I	E	F	J				

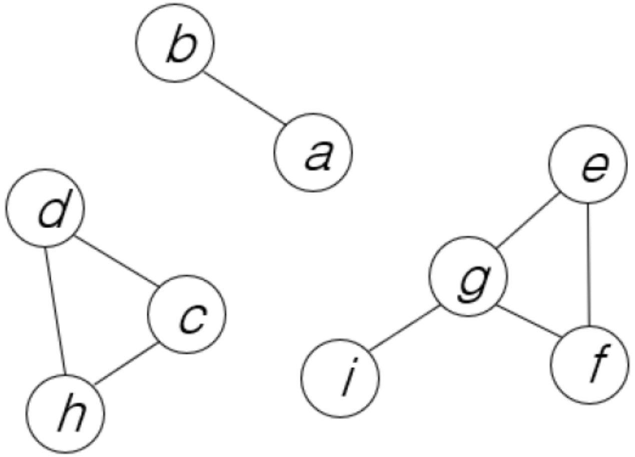
A	B	C	D	E	F	G	H	I	J	K
0	1	2	3	4	5	6	7	3	6	3

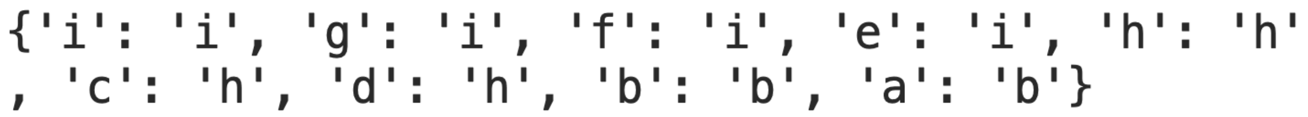
Exercise 2: Tracking Unvisited Vertices

- Complexity:
  - The initialization is O(|V|)
  - Determining if the vertex is visited is fast: O(1)
  - Marking vertex as having been visited is also fast: O(1)
  - Returning a vertex that is unvisited is also fast: O(1)

Exercise 2: Tracking Unvisited Vertices

- Implement the function trackConnectComponents(G) which inputs the graph in Figure 2 and print the connected components.





```
{'i': 'i', 'g': 'i', 'f': 'i', 'e': 'i', 'h': 'h'
, 'c': 'h', 'd': 'h', 'b': 'b', 'a': 'b'}
```

Now, you have 10 minutes to implement ex2.

Q & A