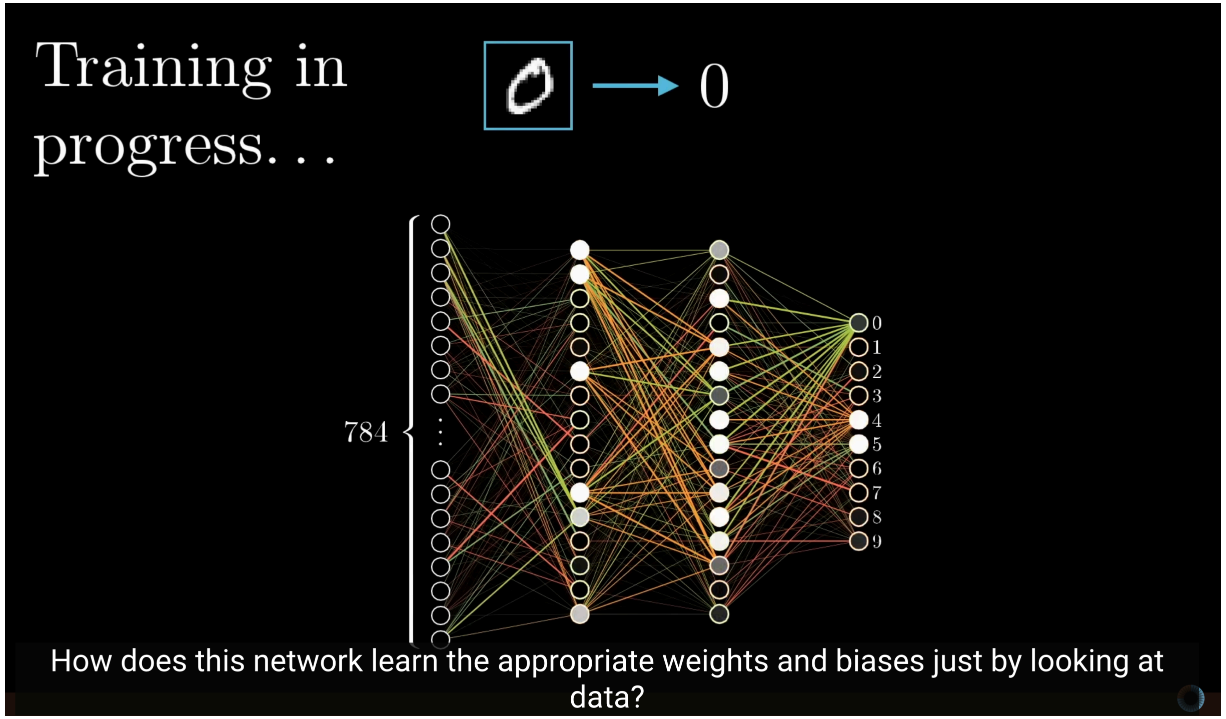
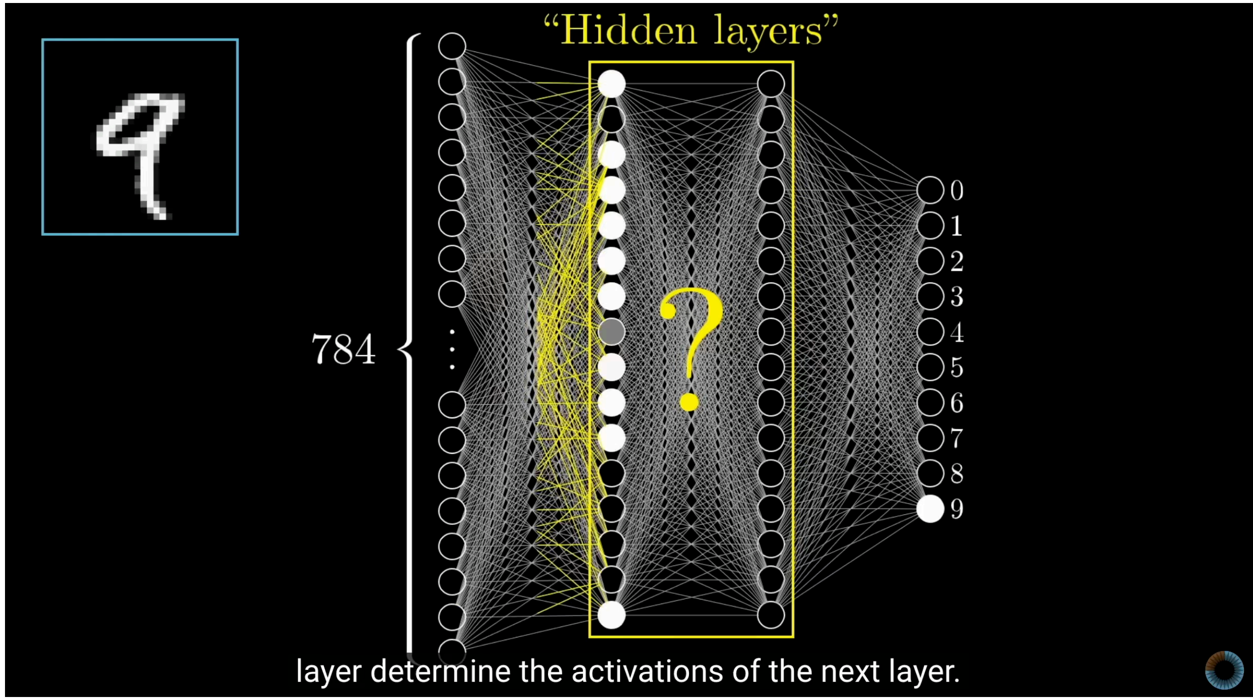
How does information flow in a NeuralNetwork ?



How does this network learn the appropriate weights and biases just by looking at data?



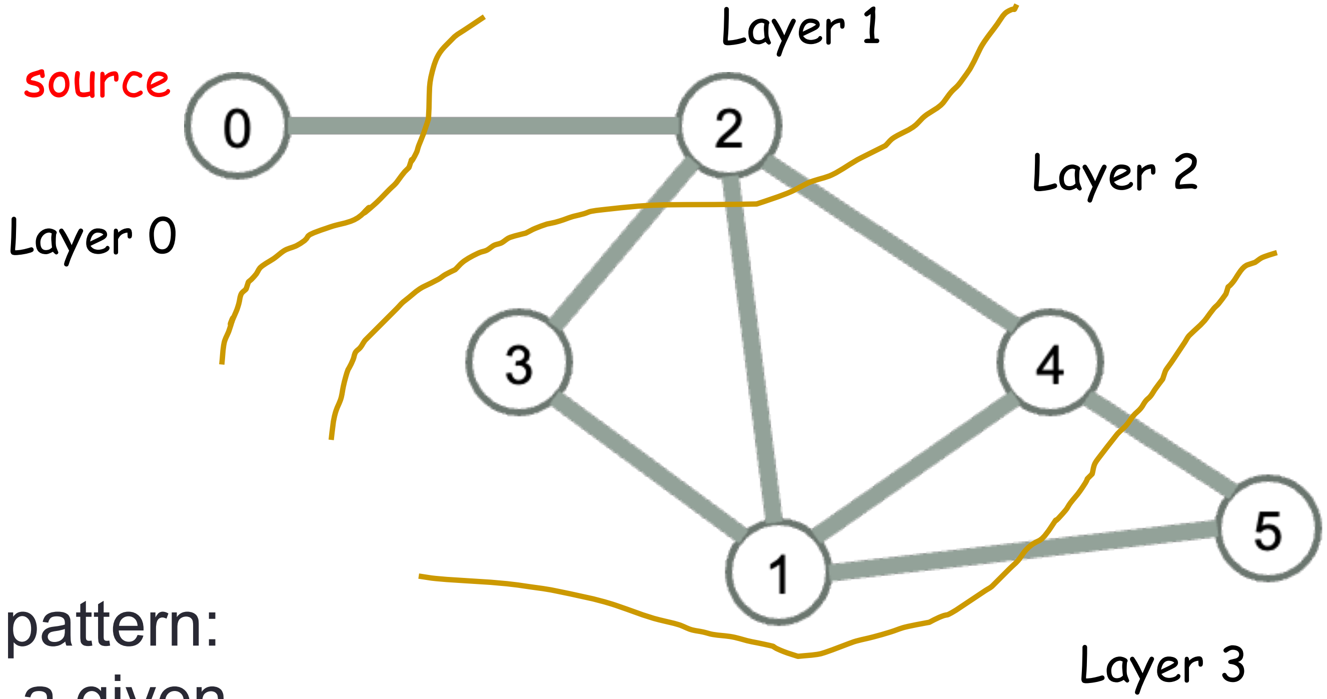
layer determine the activations of the next layer.

Training
Learn network parameters

Evaluation/Prediction
Network produces outputs from inputs

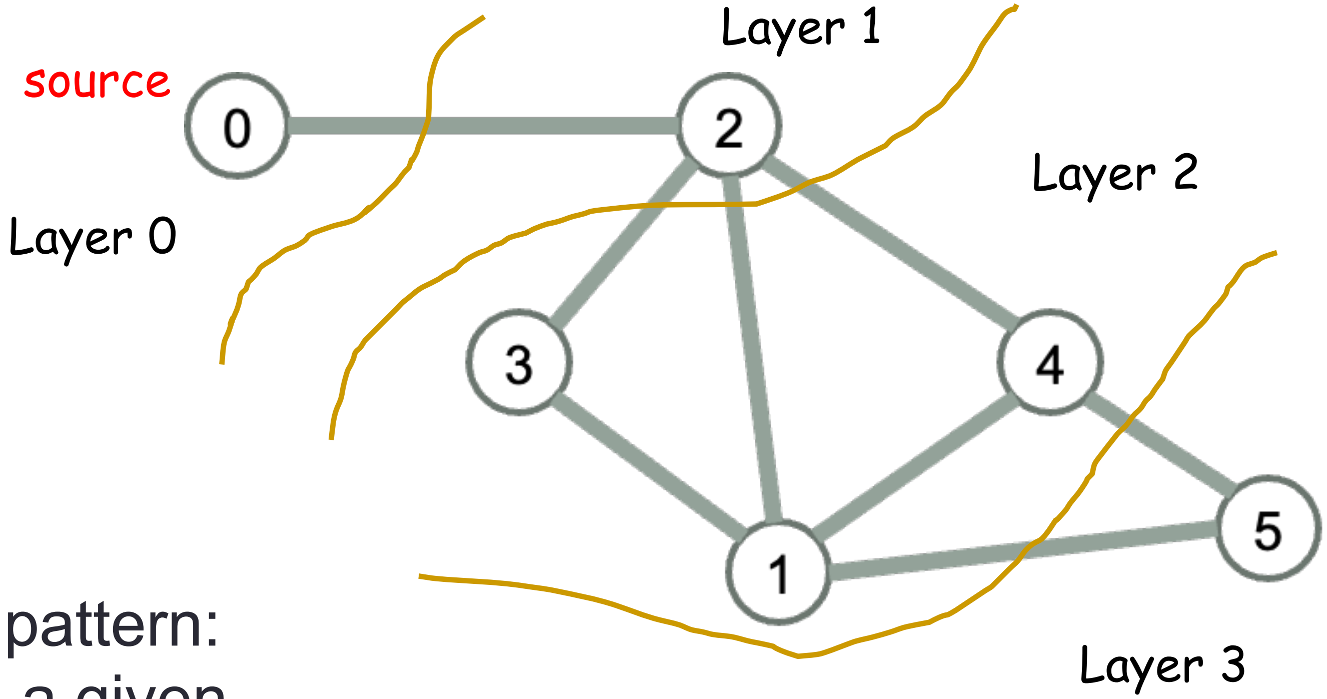
Credits: 3Blue1Brown

Breadth First Traversal: Sketch of Algorithm



Explore the graph in a wave (layered) pattern: explore all the vertices reachable from a given vertex before exploring their neighbors.

Breadth First Traversal: Sketch of Algorithm

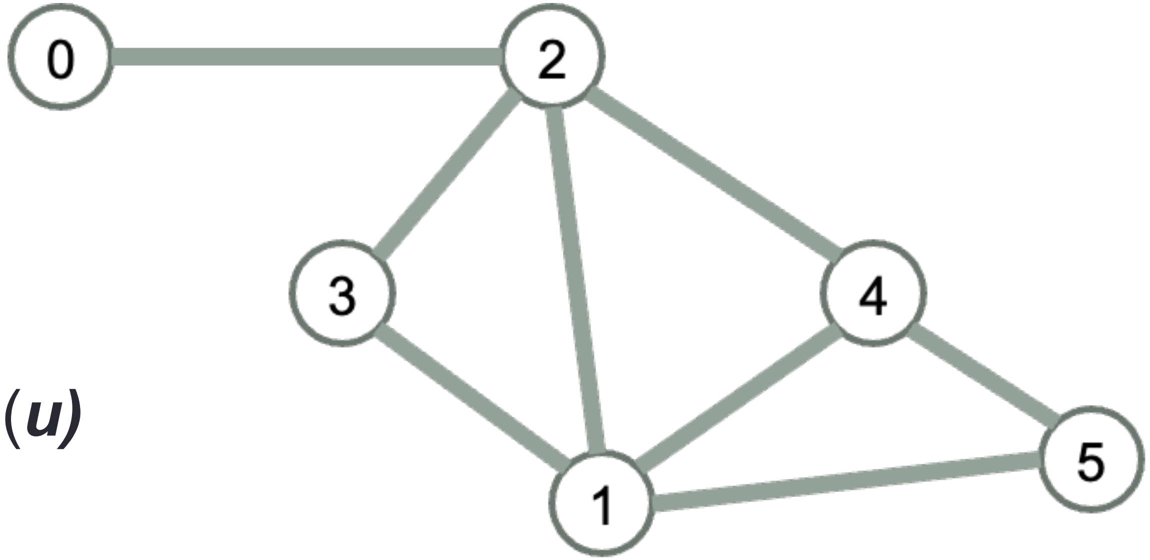


Explore the graph in a wave (layered) pattern: explore all the vertices reachable from a given vertex before exploring their neighbors.

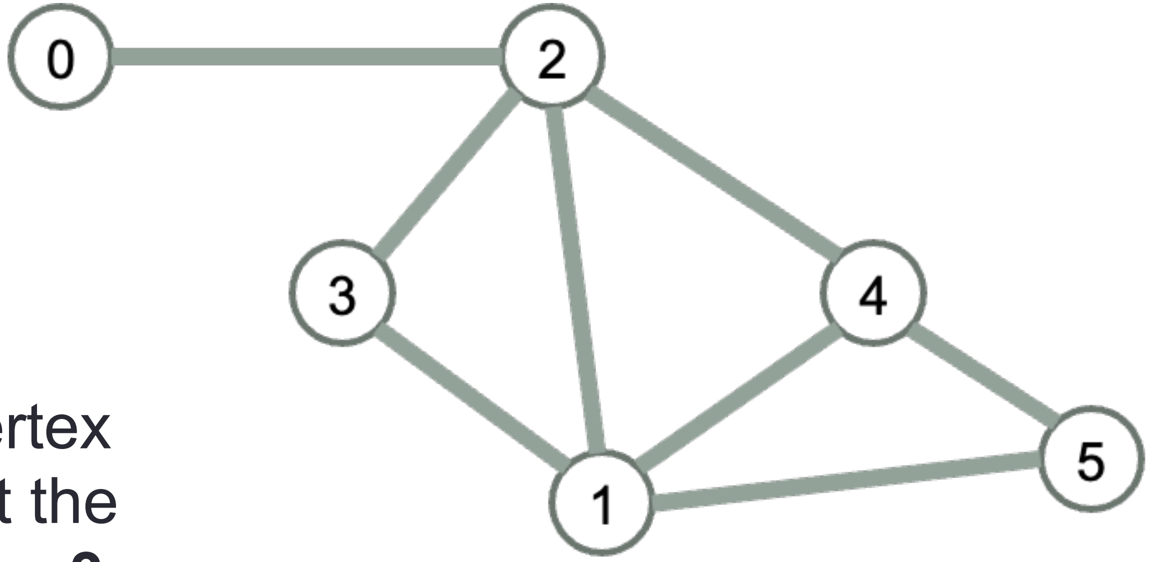
- In general, a search algorithm would explore (or “visit”) from a source vertex
  - all the vertices reachable ,
  - never exploring out from the same vertex twice
- How does the Breadth First Search/Traversal algorithm ensure this?

Breadth First Algorithm

```
Algo exploreBFS (Graph G, vertex s):
• Mark all the vertices as “not visited”
• Visit s
• Push s into a queue
• while the queue is not empty:
    • pop the vertex from the front of the queue (u)
    • for each of u’s neighbor (v)
        • If v has not yet been visited:
            • Visit v
            • Push v in the queue
```



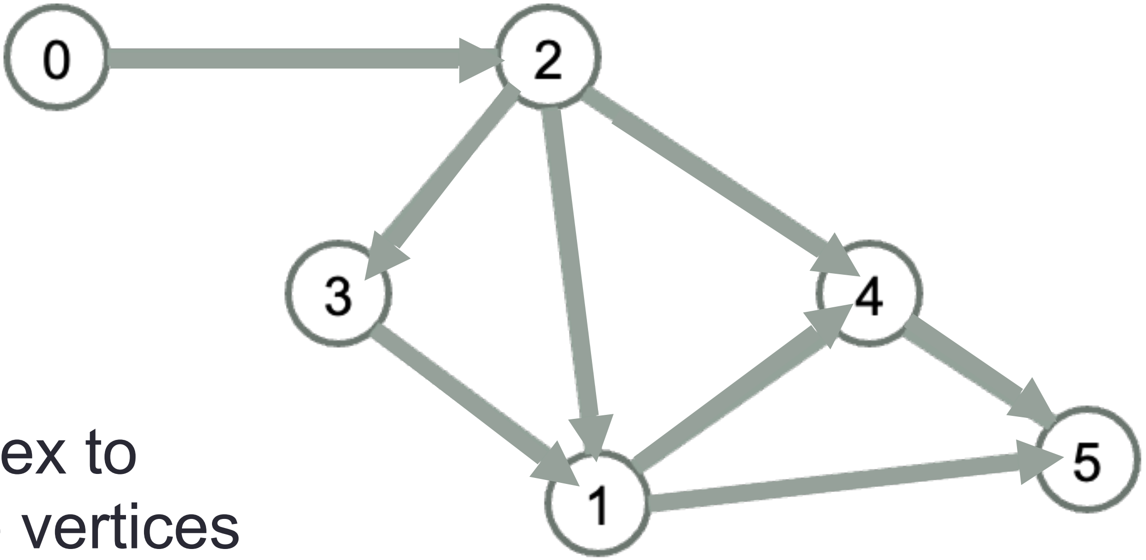
Trace BFS for the example graph



Assume BFS chooses the lower number vertex to explore first, in what order does BFS visit the nodes in this graph starting at **source vertex 3**.

A. 0, 1, 2, 3, 4, 5
B. 0, 1, 3, 2, 4, 5
C. 3, 2, 0, 1, 4, 5
D. 3, 1, 2, 0, 4, 5
E. Something else

Trace BFS (different source vertex)

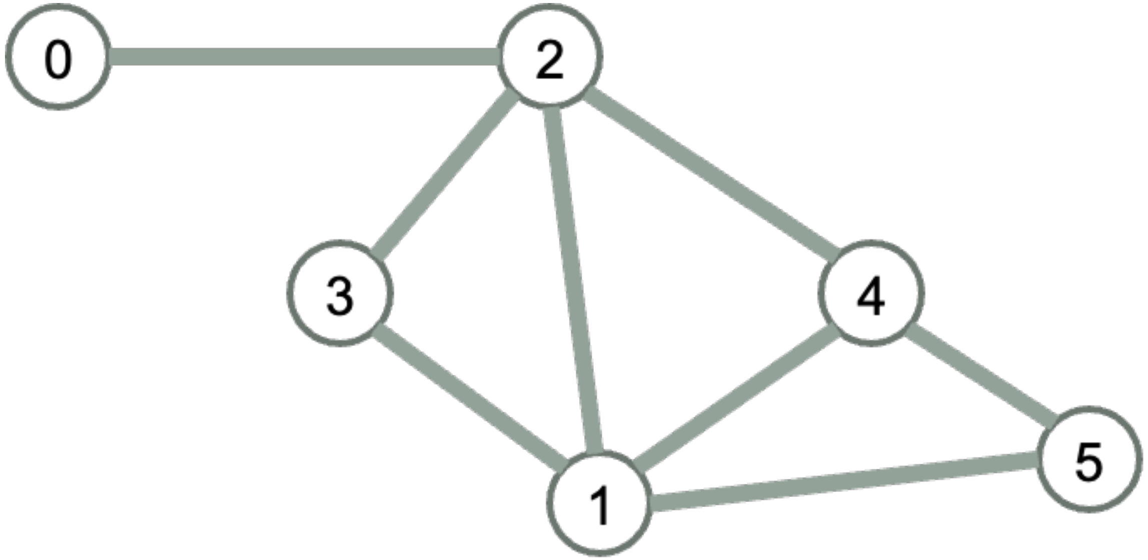


What if edges were directed as shown?
Assume BFS chooses the lower number vertex to explore first, in what order does BFS visit the vertices in this graph starting at **source vertex 3**.

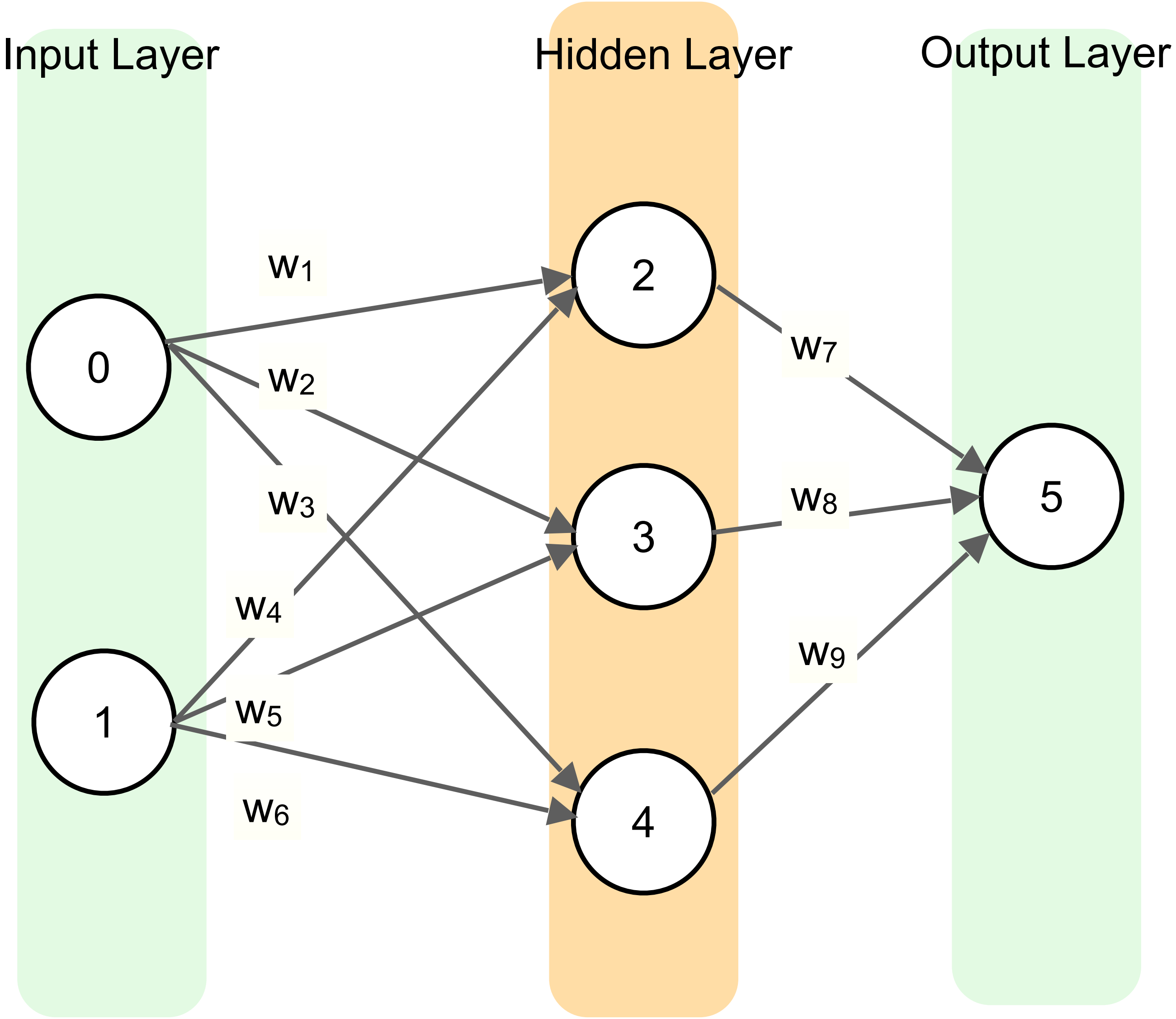
A. 0, 1, 2, 3, 4, 5
B. 0, 1, 3, 2, 4, 5
C. 3, 2, 0, 1, 4, 5
D. 3, 1, 2, 0, 4, 5
E. Something else

```
class graph{
   public:
      graph(int n = 0) { // n is the number of vertices
         ________________________________
      }
      void addEdge(int from, int to);
      vector<bool> exploreBFS(int source) const;

   private:
      vector<__________________________> adjList;
};
```



PA03: Modify BFS to perform prediction



In the prediction step, activation values in one layer determine the activation values of the neurons in the next layer.

Your task: Modify the standard BFT algo to do the necessary computation in the expected (layer by layer) order.

Complete the provided NeuralNetwork::predict() function