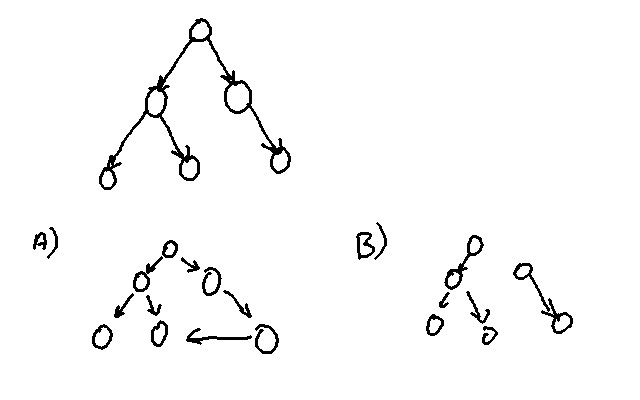
1. Draw a binary tree that contains six nodes. Then modify the figure so that it is no longer a tree but it is: Note: The "connected" graphs are all weakly connected. Additional changes would need to be made for them to be strongly connected.

a. A connected graph.

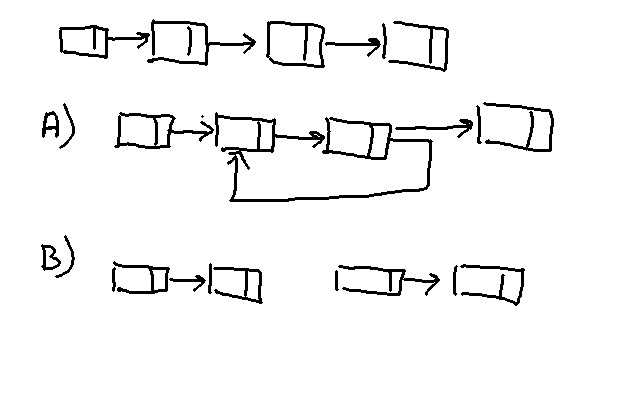
b. A disjoint (unconnected) graph.



2. Draw a singly linked list that contains four nodes. Then modify the figure so that it is no longer a singly linked list but it is:

a. A connected graph.

b. A disjoint (unconnected) graph.



3. Define the terms:

a. Graph - A set of vertices, a set of edges and an "adjacency" matrix that defines how the edges connect the vertices.

b. Undirected graph - A graph where the edges do not have a direction imposed.

c. Directed graph - A graph where the edges have a specific direction.

d. Path - A connected sequence of vertices and edges.

e. Path length - The total weighted length of the path.

f. Unconnected (disjoint) graph - A graph were there are at least one pair of vertices which are not connected by any path in either direction.

g. Cycle - A path that is begins and ends on the same vertex.

h. Complete graph - A graph where there is an edge between any pair of vertices.

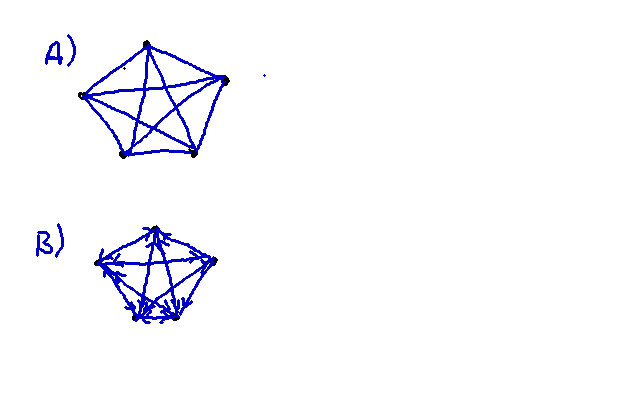
i. Simple path - A path which visits each vertex only once except for the endpoints.

j. Strongly connected digraph - A digraph (a directed graph) where there is a path connecting any pair of vertices.

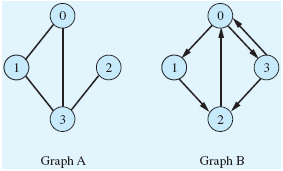
4. Calculate the number of edges in a complete graph containing five vertices, and then draw the graph and count the edges to verify your calculation assuming the graph is:

a. An undirected graph. 10

b. A digraph. 20



5. Give the adjacency matrix for the graphs A and B shown below. (Fill in the boxes below.)



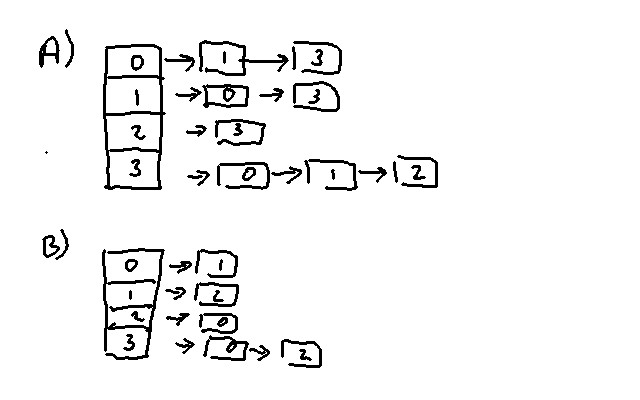
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Graph A** | | | | |  | |  |  | **Graph B** | | | | | |
|  | 0 | 1 | 2 | 3 | |  |  |  | |  | 0 | 1 | 2 | 3 |
| 0 | 0 | 1 | 0 | 1 | |  |  |  | | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | |  |  |  | | 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 0 | 0 | 1 | |  |  |  | | 2 | 1 | 0 | 0 | 0 |
| 3 | 1 | 1 | 1 | 0 | |  |  |  | | 3 | 1 | 0 | 1 | 0 |

6. Give the adjacency list representation of the graphs A and B, shown above, assuming:

a. The number of vertices is fixed. (For each, copy and paste your diagram.)

Graph A

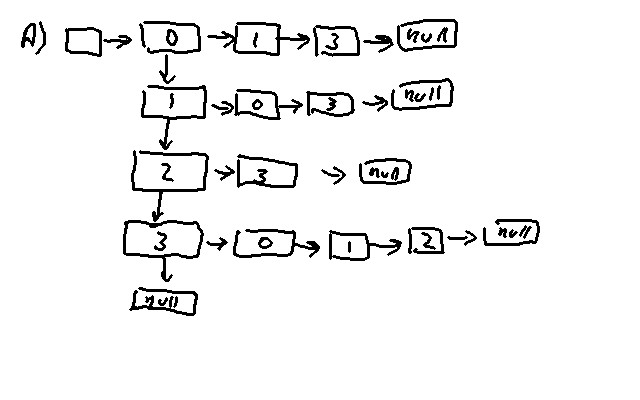
Graph B

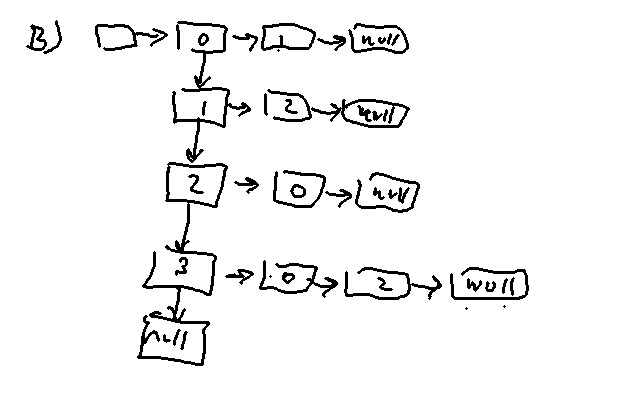


b. The number of vertices can expand to a potentially large value and expanding arrays is a slow operation.

Graph A

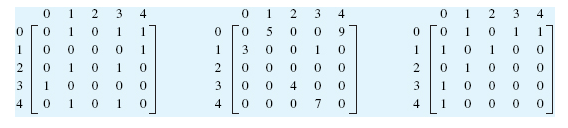
Graph B

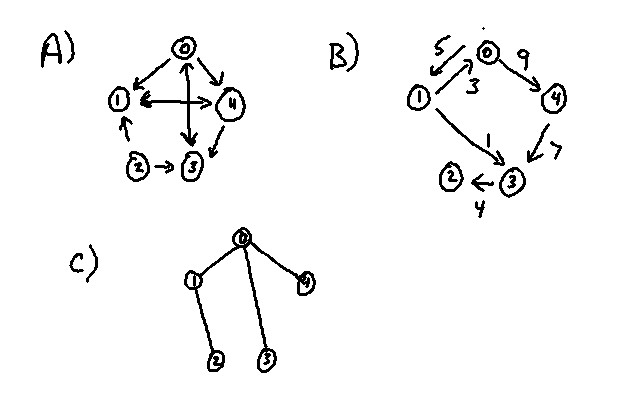




7. Draw the graph whose edges are represented by the following matrices. Assume the vertices are named V0 through V4. (For each copy and paste your diagram into your document.)

A B C





8. Consider the graphs in the previous exercise. Which of them are:

a. Disjoint? - None of them are disjoint.

b. Directed? - A,B

c. Weighted? - B

d. Undirected? - C

9. The number of vertices and edges for two undirected and two directed graphs are given in the following table. Fill in the last column of the table to indicate which representation presented in Figure 9.12 would be the best performing representation for each graph from a time-complexity viewpoint. Assume array expansion is slow.

|  |  |  |  |
| --- | --- | --- | --- |
| Type of Graph | Maximum Number of Vertices in the Graph | Maximum Number of Edges in the Graph | Best Representation Figure 9.12a, 9.12b, or 9.12c |
| Undirected | 10 | 35 | a |
| Undirected | 10 | 16 | b |
| Directed | 10 | 35 | a |
| Directed | unknown | unknown | c |

10. Assuming that an undirected, unweighted graph is represented as shown in Figure 9.12a and that operations are performed in the vertex number mode, ***give the signature*** and ***describe the actions*** of a method that:

a. Deletes an edge from the graph.

boolean deleteEdge(int vertex1, int vertex2);

1) Look for the appropriate entry in the Adjacency matrix.

2) If it's a 1 then set it to 0 and return true.

If it's a zero then return false.

b. Deletes a vertex from the graph.

boolean deleteVertex(int vertex)

1) Search for the vertex.

2) If found, remove from list and remove all lines that terminate on vertex from adjacency matrix. Return true

If not found, return false.

c. Updates an edge in the graph.

boolean updateEdge(int oldStart, int oldEnd, int newStart, int newEnd)

1) Search for old edge.

2) If found do deleteEdge followed by insertEdge return true

otherwise return false.

11. Repeat the previous exercise assuming the graph is:

a. Directed and unweighted.

These are not in essence different from those in the previous exercise. The main issue is the "remove from list etc." step in the deleteVertex method. There is actually no difference in the code since the only thing to do is zero out the column and row of the adjacency matrix. Reducing the size of the matrix is not needed and time consuming. One could keep track of the deleted nodes and use the empty slots when new nodes are added. None of the signatures are different.

b. Undirected and weighted.

For these the major change is keeping track of the weights of the edges. This would be easily done by creating an Edge object that also contains a Listing. The Listing could contain the weight. For our example this could be the mileage or the fare.

The signature of updateEdge is the only one that is different:

boolean updateEdge(int oldStart, int oldEnd, int newStart, int newEnd, Listing newListing)

12. Assume a graph is represented as shown in Figure 9.13 and that operations are performed in the vertex number mode. Give the errors that could occur during:

a. An insert vertex operation. - Graph is Full.

b. An insert edge operation. - Endpoint does not exist.

c. A show vertex operation. - Vertex does not exist.

14. Is an NLR traversal of a binary tree a depth-first or breadth-first traversal? The NLR traversal is depth first.

15. Which traversal algorithm uses a queue, breadth-first, or depth-first, and why?

The Breadth-First traversal uses a Queue. This is to ensure that siblings are searched before descendants.

16. Assuming vertex 1 in graph A of Exercise 5 is visited first, give the order in which the vertices of the graph are visited if the traverse is a:

a. Depth-first traversal. 1-2-3-0

b. Breadth-first traversal. 1-0-3-2

17. In a connected undirected graph, there is always a path from vertex A to B, for all A and B. True or false? True.

18. In a connected digraph, there is always a path from vertex A to B, for all A and B. True or false? False.