**Question 1**(1 point)

*Saved*

An adjacency matrix is not symmetric along the major diagonal. This points to the fact that the Graph may be

Question 1 options:

|  |  |
| --- | --- |
|  | None of these |
|  | a Digraph |
|  | an undirected graph |
|  | a hyperbolic graph |
|  | a sparse graph |

**Question 2**(1 point)

*Saved*

Given a graph G(V,E), and |V|=v, and |E|=e, if G is a treen then:

Question 2 options:

|  |  |
| --- | --- |
|  | v+1=e |
|  | v=e+1 |
|  | v=e |
|  | v=e-1 |

**Question 3**(1 point)

*Saved*

What type of graph is required for a **topological sort** to be valid?

Question 3 options:

|  |  |
| --- | --- |
|  | Undirected Graph |
|  | Cyclic Graph |
|  | Directed Acyclic Graph |
|  | Weighted Graph |

**Question 4**(1 point)

*Saved*

Which of the following algorithms is used to determine the shortest path from a source vertex to all other vertices, even with negative weights?

Question 4 options:

|  |  |
| --- | --- |
|  | Dijkstra's Algorithm |
|  | Depth-First Search (DFS) |
|  | Bellman-Ford's Algorithm |
|  | Topological Sort |

**Question 5**(1 point)

*Saved*

Which of the following statements is true about **directed weighted graphs**?

Question 5 options:

|  |  |
| --- | --- |
|  | All paths have equal weight |
|  | Weights can be negative, but must not form negative weight cycles for shortest path calculations |
|  | Only cyclic graphs can have topological sort |
|  | Weights cannot be greater than the number of nodes |

**Question 6**(1 point)

*Saved*

In an **adjacency matrix** representation of a graph, determining if two vertices are adjacent takes **O(V)** time, where V is the number of vertices. -> O(1)

Question 6 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 7**(1 point)

*Saved*

The **time complexity** of the **topological sort** algorithm is **O(|V| + |E|)**.

Question 7 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 8**(1 point)

*Saved*

The **space complexity** of a graph representation using an **adjacency list** is **O(V + E)**.

Question 8 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 9**(1 point)

*Saved*

In a **depth-first search (DFS)** of a graph, which of the following data structures is typically used?

Question 9 options:

|  |  |
| --- | --- |
|  | Priority Queue |
|  | Stack |
|  | Queue |
|  | Linked List |

**Question 10**(1 point)

*Saved*

Which traversal method of a **tree** is most similar to the **breadth-first search (BFS)** of a graph?

Question 10 options:

|  |  |
| --- | --- |
|  | In-order traversal |
|  | Level-order traversal |
|  | Post-order traversal |
|  | Pre-order traversal |

**Question 11**(1 point)

*Saved*

A complete binary tree is a tree where all levels, except possibly the last, are completely filled, and all nodes are as far left as possible.

Question 11 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 12**(1 point)

*Saved*

In a binary search tree (BST), nodes in the left subtree of a node have values greater than the node's value.

Question 12 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 13**(1 point)

*Saved*

In a tree, the root node is the only node that does not have a parent.

Question 13 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 14**(1 point)

*Saved*

In a complete binary tree, every level must be completely filled with nodes. -> Last node might be empty

Question 14 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 15**(1 point)

*Saved*

In a binary search tree (BST), nodes in the left subtree have values less than the root node, and nodes in the right subtree have values greater than the root node.

Question 15 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 16**(1 point)

*Saved*

In a binary tree, the number of children a node can have is called the node’s degree.

Question 16 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 17**(1 point)

*Saved*

Post-order traversal visits the root node before visiting its left and right subtrees. -> Visit root node after visiting left and right

Question 17 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 18**(1 point)

*Saved*

Which of the following traversal methods **visits the root node last**?

Question 18 options:

|  |  |
| --- | --- |
|  | Pre-order |
|  | In-order |
|  | Post-order |
|  | Level-order |

**Question 19**(1 point)

*Saved*

Which of the following is a **correct property** of a **Binary Search Tree (BST)**?

Question 19 options:

|  |  |
| --- | --- |
|  | Inserting values is always done at the root |
|  | The left subtree contains only nodes with values greater than the root |
|  | The left and right subtrees are not restricted in value |
|  | Each node's value must be greater than the values in its left subtree |

**Question 20**(1 point)

*Saved*

What does an AVL tree maintain to ensure balance?

Question 20 options:

|  |  |
| --- | --- |
|  | The height difference between left and right subtrees is at most 1 |
|  | Each node has exactly two children |
|  | The number of nodes in the left subtree is equal to the number in the right subtree |
|  | Each level of the tree is completely filled |

**Question 21**(1 point)

*Saved*

Which of the following correctly defines a **max-heap**?

Question 21 options:

|  |  |
| --- | --- |
|  | Every parent node has a value greater than or equal to its child nodes |
|  | All nodes have the same value |
|  | Every child node has a value greater than its parent node |
|  | The tree is always completely filled |

**Question 22**(1 point)

*Saved*

A single rotation can always fix any imbalance in an AVL tree.

Question 22 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 23**(1 point)

*Saved*

The **balance factor** of a node in an AVL tree must be 0, 1, or -1.

Question 23 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 24**(1 point)

*Saved*

In an AVL tree, a rotation is needed whenever a new node is inserted.

Question 24 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 25**(1 point)

*Saved*

Which of the following is **true** about a hash table?

Question 25 options:

|  |  |
| --- | --- |
|  | It can only store sorted items |
|  | It maps each item to a specific location using a key |
|  | It relies on depth-first search to retrieve items |
|  | It uses a stack to manage collisions |

**Question 26**(1 point)

*Saved*

Which collision resolution method uses linked lists at each bucket?

Question 26 options:

|  |  |
| --- | --- |
|  | Quadratic Probing |
|  | Linear Probing |
|  | Double Hashing |
|  | Chaining |

**Question 27**(1 point)

*Saved*

Which of the following statements about **linear probing** is true?

Question 27 options:

|  |  |
| --- | --- |
|  | It can lead to primary clustering |
|  | It requires linked lists to store all collided items |
|  | It requires the use of a secondary hash function |
|  | It always results in evenly distributed keys |

**Question 28**(1 point)

*Saved*

The **quadratic probing** method ensures that items will always find an empty bucket to be inserted into.

Question 28 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 29**(1 point)

*Saved*

In **double hashing**, the secondary hash function must return a value that can be zero.

Question 29 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 30**(1 point)

*Saved*

A **collision** in a hash table occurs when two different keys map to the same bucket.

Question 30 options:

|  |  |
| --- | --- |
|  | True |
|  | False |