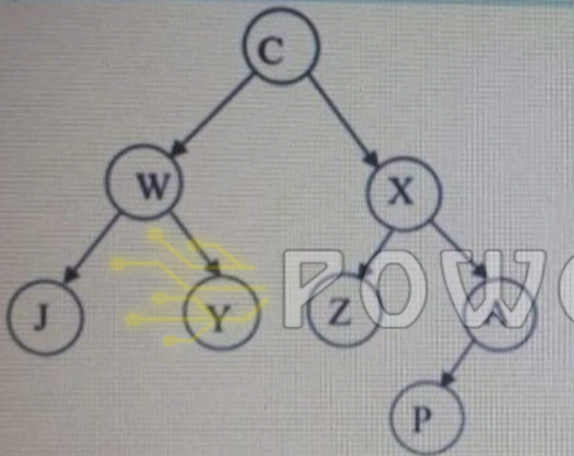
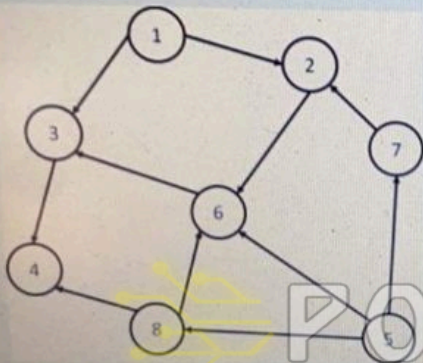


Give an in-Order traversal of the tree shown below, separate the name of the nodes by comma and don't put any space (example of the answer: C,W,X,J,Y,Z,A,P). Estimated solution time is 3-4 minutes.



Answer: J,W,Y,C,Z,X,P,A | I

Given the following directed graph. After applying depth first search from node 2, which vertexes will be marked as visited? You should choose the largest set of vertexes marked as visited.



- 1, 2, 3, 4
- 1, 2, 3, 4, 6
- 1, 2, 3, 4, 5, 6, 7, 8
- 2, 3, 4, 6
- 2, 3, 4, 6, 8

[Clear my choice](#)

In the following set of questions, write only the final answer. For example to write $O(N)$, just write N without $O()$. To write $O(N^2)$, just write N^2 , and so on. To write $O(\log N)$, just write $\log N$, to write $O(N \cdot V)$, just write $N \cdot V$ (Use the star symbol $*$ for multiplication). To write constant time, write 1. The question is automatically graded.

For each of the functions $f(N)$ given below, indicate the resulting **growth order**. (Estimated solution time 3-4 minutes).

$$f(N) = N^3 * (2 * \log N + \log N) + N^4$$

$$f(N) = N^{1/2} + \log N$$

POWERUNIT


$$f(N) = 110 * \log N^2 + 5 * N^2 * \log N$$

$$f(N) = ((N + 2) * (N + 3)) / 2$$

Suppose we have a hash function $H(x)=x$

Using linear probing hash table of size 9, we inserted a set of values in the indexes as shown below, what is a probable sequence of insertions on the hash table?

| | | | | | | | | | |
|--------|----|----|---|---|---|----|----|---|---|
| Key= | 17 | 18 | | 3 | 4 | 12 | 21 | | 8 |
| Index= | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

-  POWERUNIT
- 3, 4, 12, 21, 17, 18, 8
 - 8, 17, 3, 4, 12, 21, 18
 - 3, 8, 4, 21, 17, 18, 12
 - 3, 4, 12, 21, 17, 8, 18
 - 3, 8, 4, 21, 17, 12, 18

By inserting values in a hash table that uses Separate chaining,

- performance of inserts will gradually degrade as more values are inserted
- possibly fail to find a location on some insertions
- none of the answers is correct
- be fine on the first few insertions, but may fail to find a location on any insertions after that

In the following set of questions, write only the final answer. For example to write $O(V)$, just write V without $O(V)$. To write $O(V^2)$, just write V^2 , and so on. To write $O(\log V)$, just write $\log V$, to write $O(N \cdot V)$, just write $N \cdot V$. Use the star symbol (*) for multiplication. The question is automatically graded. To write constant time, write 1. (Estimated solution time 7-8 minutes).

For each of the following, answer in terms of the number of vertices in the graph, V .

- Suppose a graph has no edges.

- The asymptotic space cost of storing the graph as an adjacency list is:

- The asymptotic space cost of storing the graph as an adjacency matrix is:

- Suppose a graph has every possible edge.

- The asymptotic space cost of storing the graph as an adjacency list is:

- The asymptotic space cost of storing the graph as an adjacency matrix is:

- Is an adjacency list faster or slower than an adjacency matrix for answering queries of the form, "is edge (u, v) in the graph"?

- Is an adjacency list faster or slower than an adjacency matrix for answering queries of the form, "are there any directed edges with u as the source node"?

In the following set of questions, write only the final answer. For example to write $O(N)$, just write N without $O()$. To write N^2 , and so on. To write $O(\log N)$, just write $\log N$, to write $O(N \cdot V)$, just write $N \cdot V$. Use the star symbol ($*$) for multiplication. The question is automatically graded. To write constant time, write 1. (Estimated solution time 9-10 minutes).

The **worst**-case running-time for the best algorithm for finding something in a symbol table implemented with a sorted array is:

The **best**-case running-time for the best algorithm for finding something in a symbol table implemented with a sorted array is:

The **worst**-case running-time for the best algorithm for finding something in a symbol table implemented with a sorted linked list is:

The **best**-case running-time for the best algorithm for finding something in a symbol table implemented with a sorted linked list:

The **complexity** of Pop an item from a Stack implemented with linked list nodes is:

The **complexity** of DeleteMin from a Priority Queue implemented with a binary min heap is:

The **complexity** of Find the **maximum** value in a binary min heap:

In the following set of questions, write only the final answer. For example to write $O(V)$, just write V without $O()$. To write $O(V^2)$, just write V^2 , and so on. To write $O(\log E)$, just write $\log E$, to write $O(E*V)$, just write $E*V$. Use the star symbol (*) for multiplication. To write constant time, write 1. The question is automatically graded. (Estimated solution time 4-5 minutes).

What is the worst case big-O running time of the following operations (use V and E rather than N in your answers). No explanation is required.

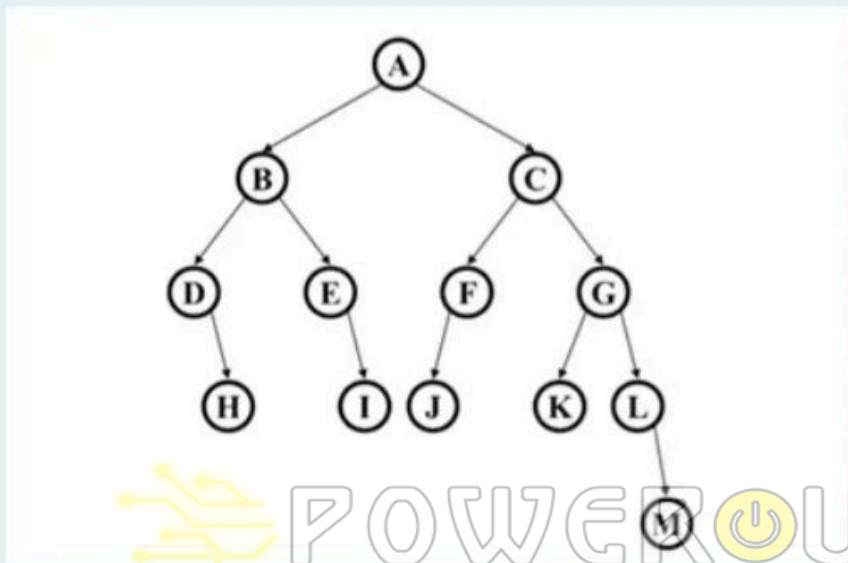
To find the in-degree of a single vertex whose graph is stored in an adjacency matrix.

To find the out-degree of a single vertex whose graph is stored in an adjacency list.

To insert an edge in a graph that is stored in an adjacency matrix.

Def: The successor of a node with key = X, is the node that has a key value min
larger than X.

Time left 0:09:0



The successor of node A is:

The successor of node B is:

The successor of node C is:

The Height of the tree is:

The maximum number of nodes that can be added to the tree without increasing its height is:

Write the array representation of the binary max heap that results from inserting the integers: 1,8,3,6,5,7 in that order into an initially empty binary min heap. 1 is inserted first. (The result should be numbers separated by commas and no spaces. example on how to write the answer is: 1,2,3,4,5,6,7,8,9). (Estimated solution time 3-4 minutes).

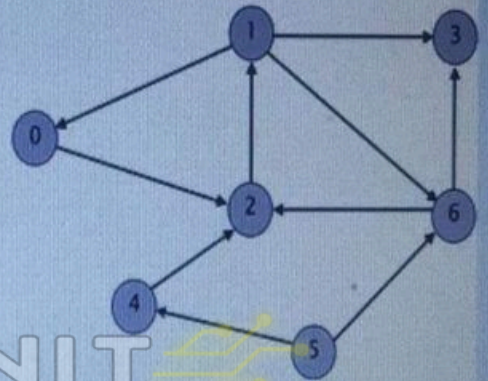
Answer:

POWERUNIT

Next page

Consider the below code and directed graph, what is the printed value of the variable *count* when the DFS function is executed for vertex 0?

```
public class DFS {  
    private boolean[] marked;  
    private int count;  
    public DFS(DGraph G, int s) {  
        count = 0;  
        marked = new boolean[G.numOfVertices()];  
        dfs(G, s);  
        System.out.println(count); // what is the printed value?  
    }  
    private void dfs(DGraph G, int v) {  
        count++;  
        marked[v] = true;  
        for (int w : G.adj(v)) {  
            if (!marked[w]) {  
                dfs(G, w);  
            }  
        }  
    }  
}
```



- 7
- 1
- 6
- 4
- 5

Clear my choice

In separate chaining hash table, if we randomly pick one of the values and delete it:

- this may cause a get on a different value to fail
- none of the above
- this will not affect performance of get operations
- this may improve performance of get operations because we search less number of items
- this may degrade performance of get operations

Time left 0:00

In linear probing, if we randomly pick one of the values and remove it from the table, leaving that location empty:

- none of the above
- this may improve performance of get operations because we search less number of items
- this may degrade performance of get operations
- this may cause a get on a different value to fail
- this will not affect performance of get operations

Clear my choice

assume you have an array $S = \{5, 2, 0, 1, 4, 3\}$ our goal is to sort this array ascendingly. Assume you use the **selection** sort algorithm studied in class, how many compares are performed when sorting the array ?

17

15

5

4

6

20

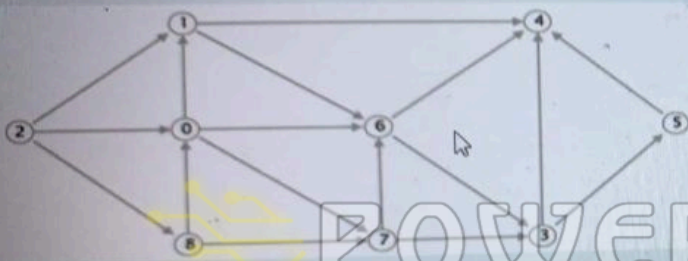
11

21



[Clear my choice](#)

Consider the following directed graph. Assume the graph uses the adjacency list representation. Which of the following vertices has adjacency list of maximum size?



- 1
- 2
- 3
- 5
- 4