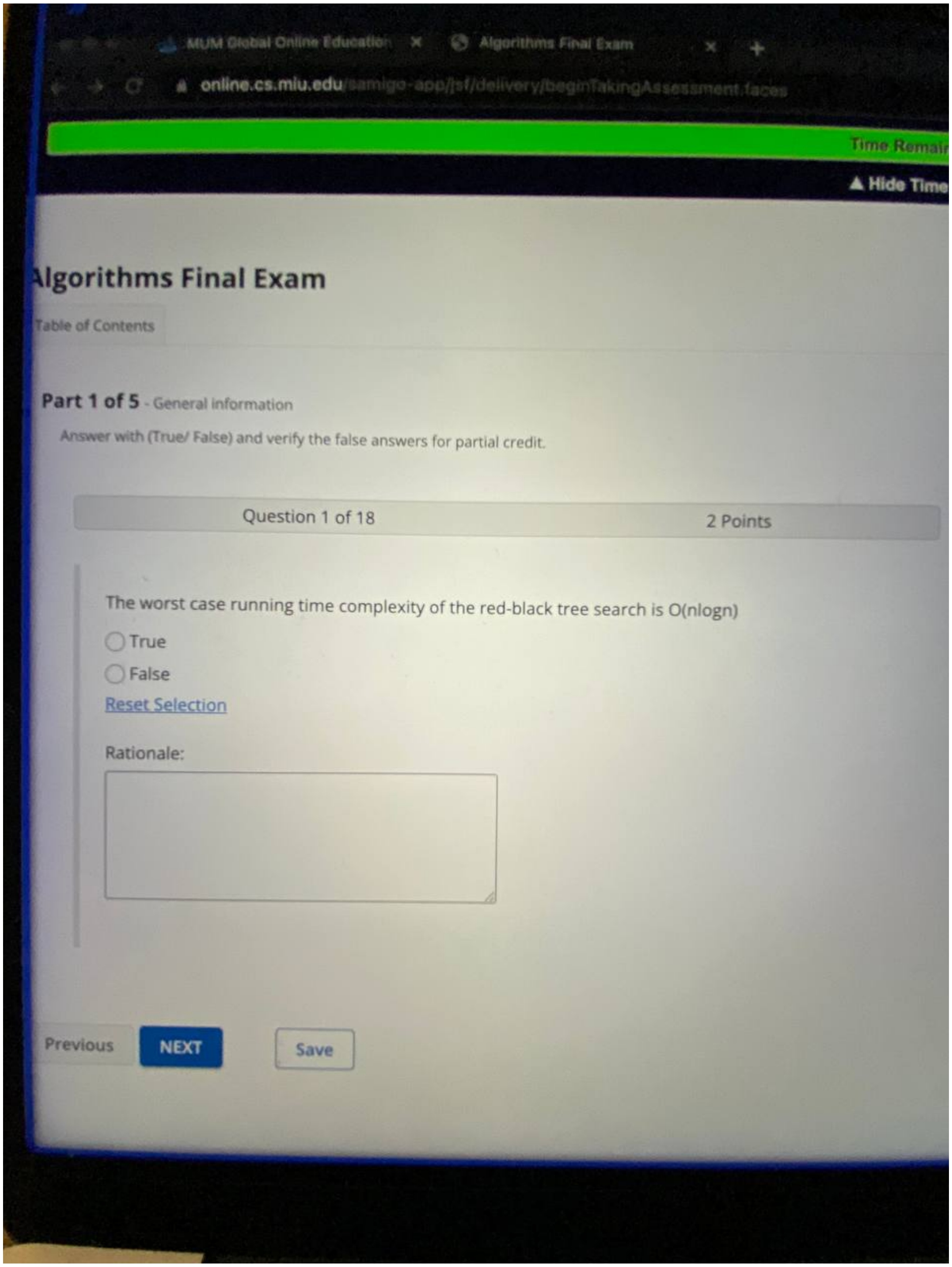
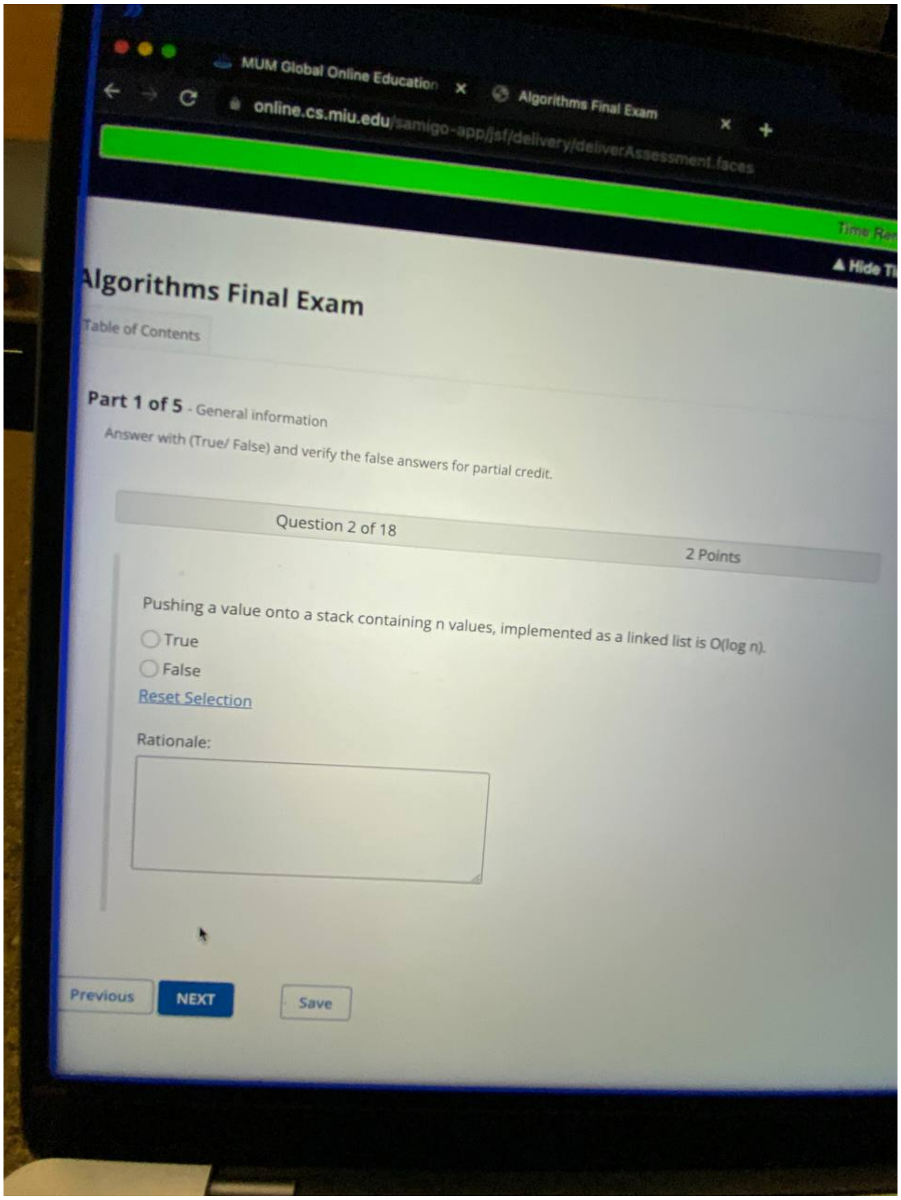
Question 1: Bijay, Nagaa



FALSE
It is O(log n) because it is a balanced binary search tree

The worst case running time complexity of a red-black tree search is O(log n). This is because a red-black tree is a self-balancing binary search tree, which maintains a balance property such that the height of the tree is always O(log n). This ensures that the search operation in a red-black tree has a worst-case time complexity of O(log n), just like a balanced binary search tree.

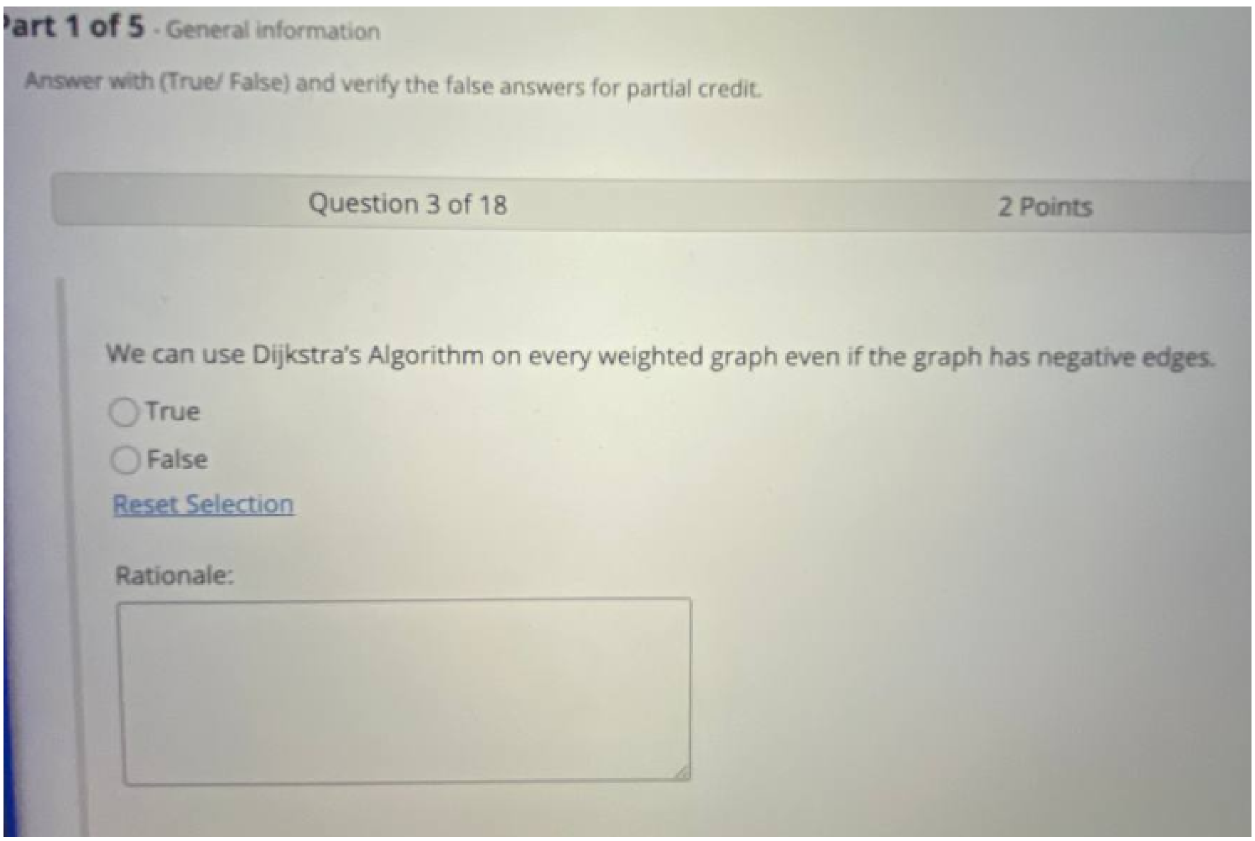
Question 2:Bijay, Nagaa



False. Pushing a value onto a stack implemented as a linked list is an O(1) operation, also known as constant time, because it only involves adding a new node to the top of the stack and updating the pointer to

the top of the stack, regardless of the number of elements in the stack.

Question 3:Luzan, Purshottam



Part 1 of 5 - General information

Answer with (True/ False) and verify the false answers for partial credit.

Question 3 of 18	2 Points

We can use Dijkstra's Algorithm on every weighted graph even if the graph has negative edges.

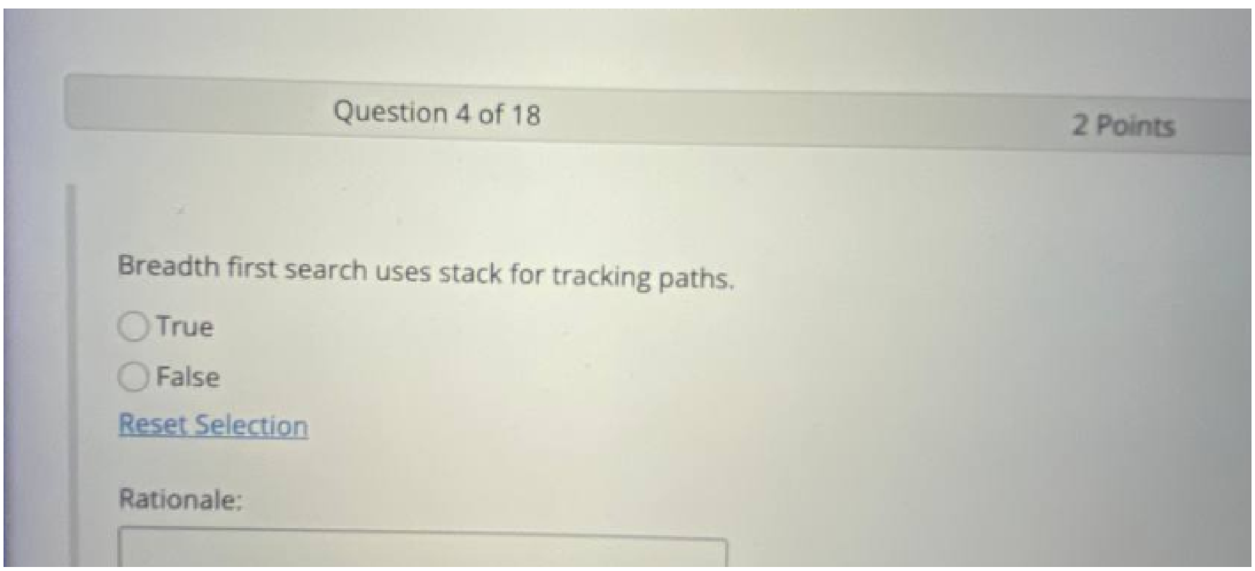
- True
- False

Reset Selection

Rationale:

False.
Dijkstra's algorithm is designed to work on graphs with non-negative edge weights. It relies on the property that the shortest path between two nodes in a graph with non-negative edge weights is also the path with the minimum total weight. This property does not hold for graphs with negative edge weights, which makes Dijkstra's algorithm inapplicable for such graphs.

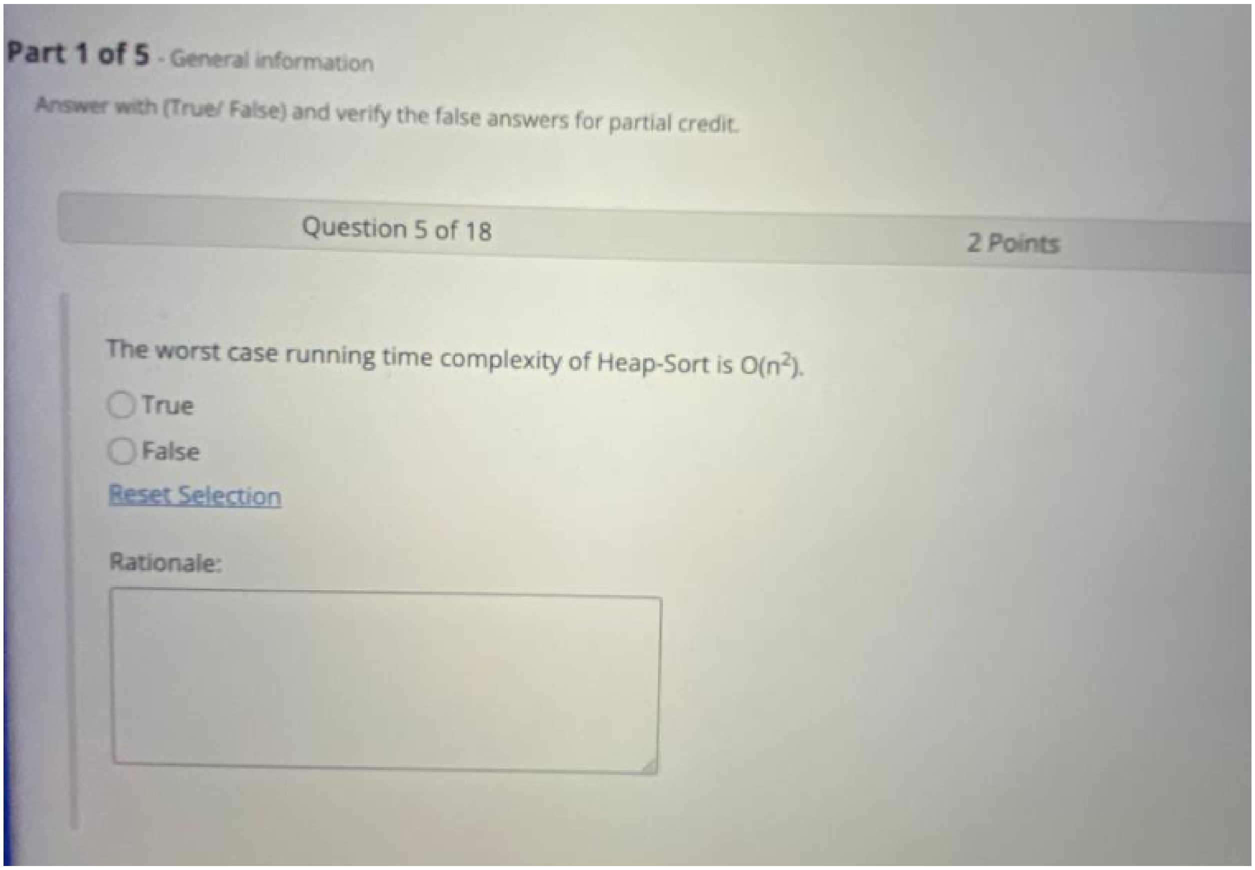
Question 4:Luzan, Purshottam



False - use queue data structure to explore the nodes of a graph or tree. The basic idea behind BFS is to explore all the nodes at the same level before moving on to the next level.

In a BFS algorithm, we start at the root node, and enqueue it into a queue. Then, we repeatedly dequeue a node from the front of the queue, examine it and enqueue all of its unvisited children. This process continues until the queue is empty or we find the desired node.

Question 5: Tes, Mihreteab



Part 1 of 5 - General information

Answer with (True/ False) and verify the false answers for partial credit.

Question 5 of 18 | 2 Points

The worst case running time complexity of Heap-Sort is $O(n^2)$.

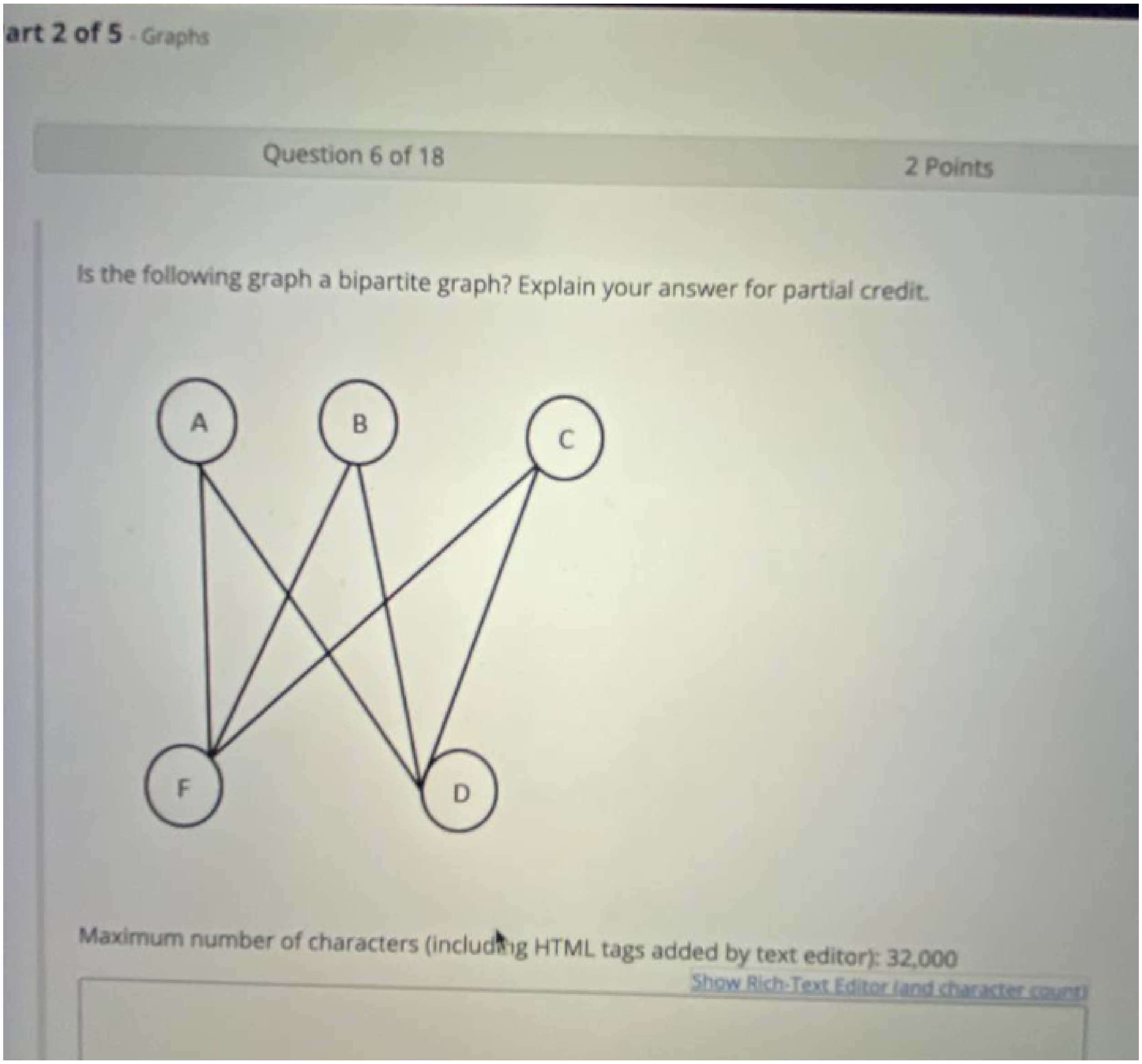
- True
- False

Reset Selection

Rationale:

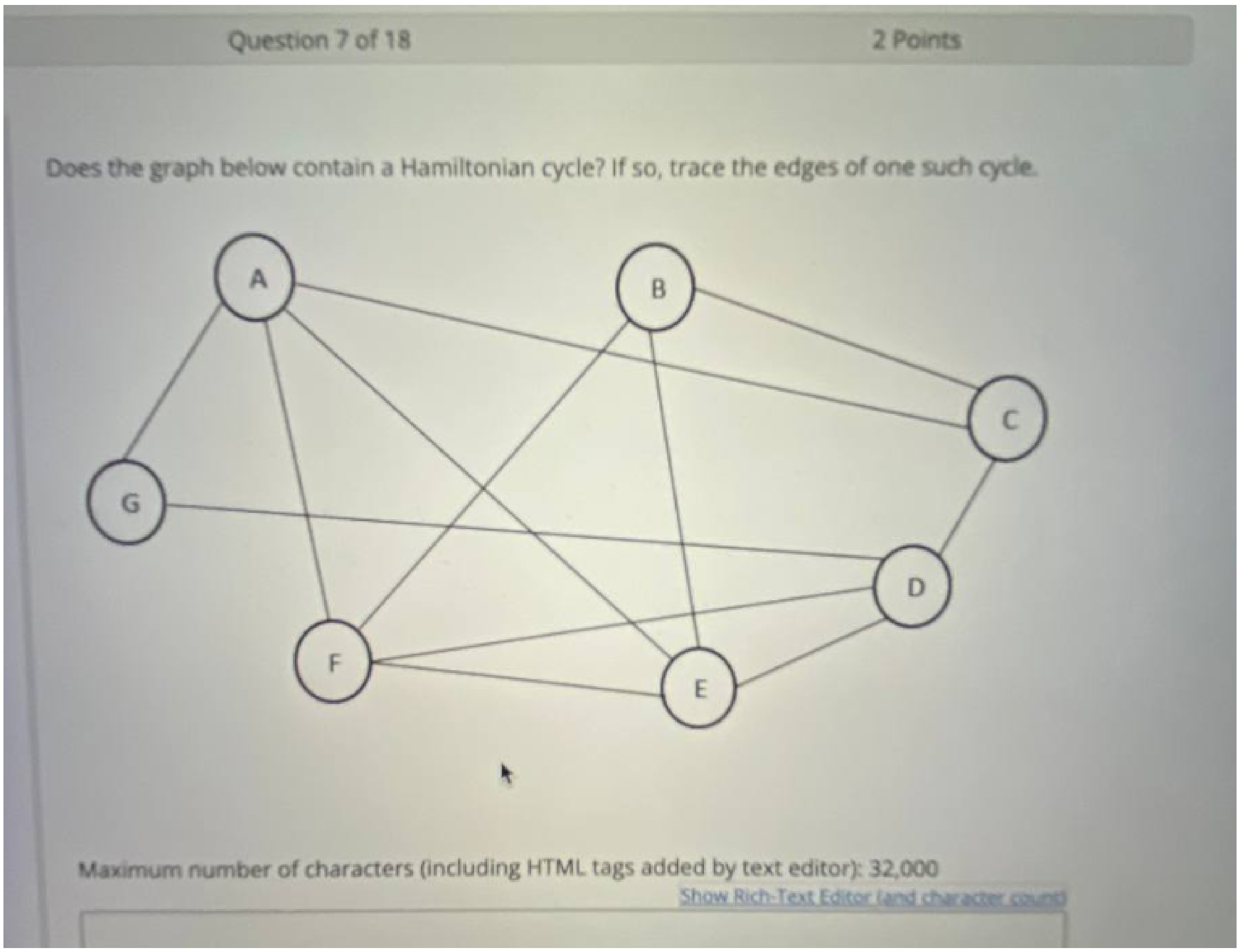
False :-   For heap Sort worst case time complexity is n*log(n)

Question 6:Tes, Mahmoud Anwar



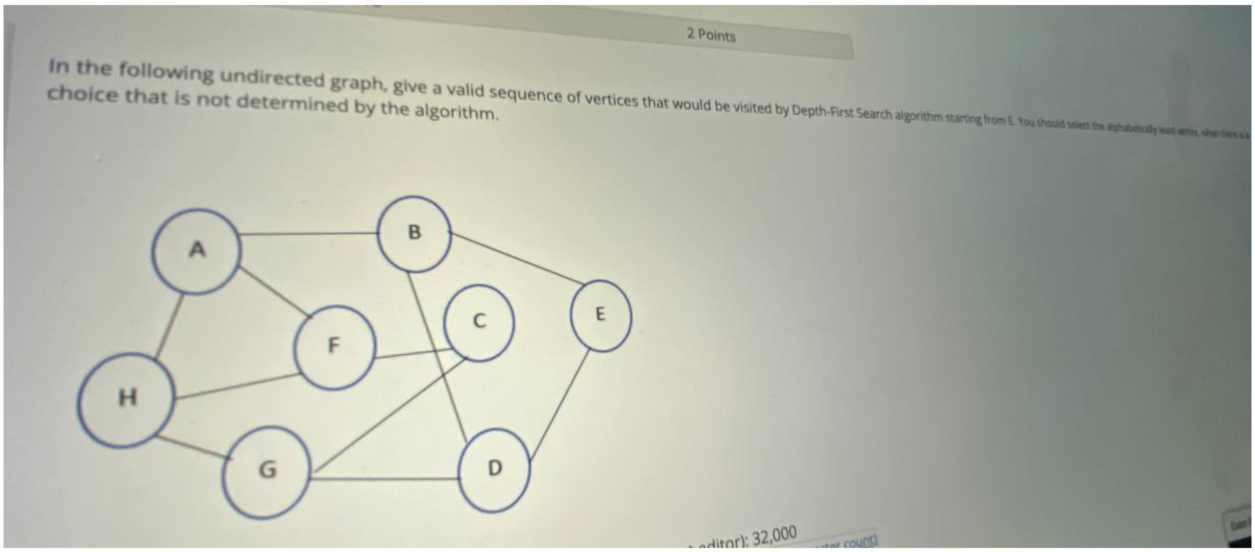
Yes ,It is a bipartite graph.
Checking from Two colorability we can see it is possible to color the vertices of the graph using two color such that no two adjacent vertices have the same color.
{A, B, C} and { F, D}

Question 7:Mahmoud Anwar, Mohamed Elfouly



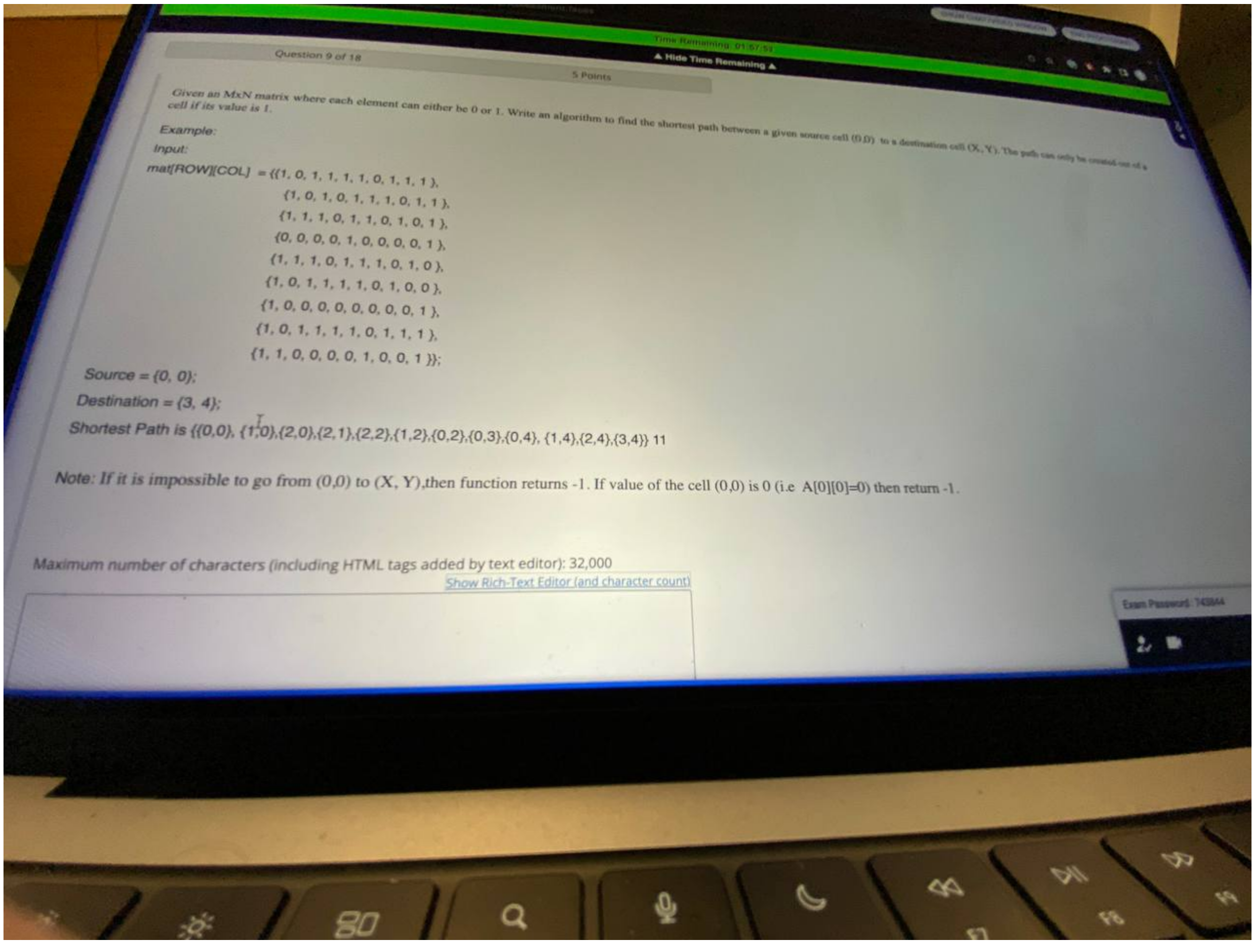
Yes as we can visit each of vertices only once except the root
A-F-E-B-C-D-G-A

Question 8:Mahmoud Anwar, Mohamed Elfouly



E -> B -> A -> F -> C -> G -> D -> H

Question 9: Mahi, Edi



Approach : declare array of visited cells to avoid infinite recursion (don't visit it if already visited)
Then start DFS
DFS mark node as visited then if its not valid or different of 1 return any high value (99999999 example)
At then end if the value is 99999999 mean there is no path between cell ((0,0) and (x,y))

And we need just to print the answer DFS(0,0)
—-------------------

1. Create a queue and add the source cell (0,0) to it.
2. Initialize a 2D array dist of the same size as the matrix, and set all elements to -1. Set dist[0][0] = 0, representing that the distance from the source cell to itself is 0.
3. While the queue is not empty:
   a. Dequeue the front element from the queue and store it in a variable current.

b. Check the four adjacent cells (left, right, up, down) of the current cell. If any of them have a value of 1 and the distance in the dist array is -1, update the distance to be 1 plus the distance of the current cell and enqueue that cell.

c. Mark the current cell as visited.

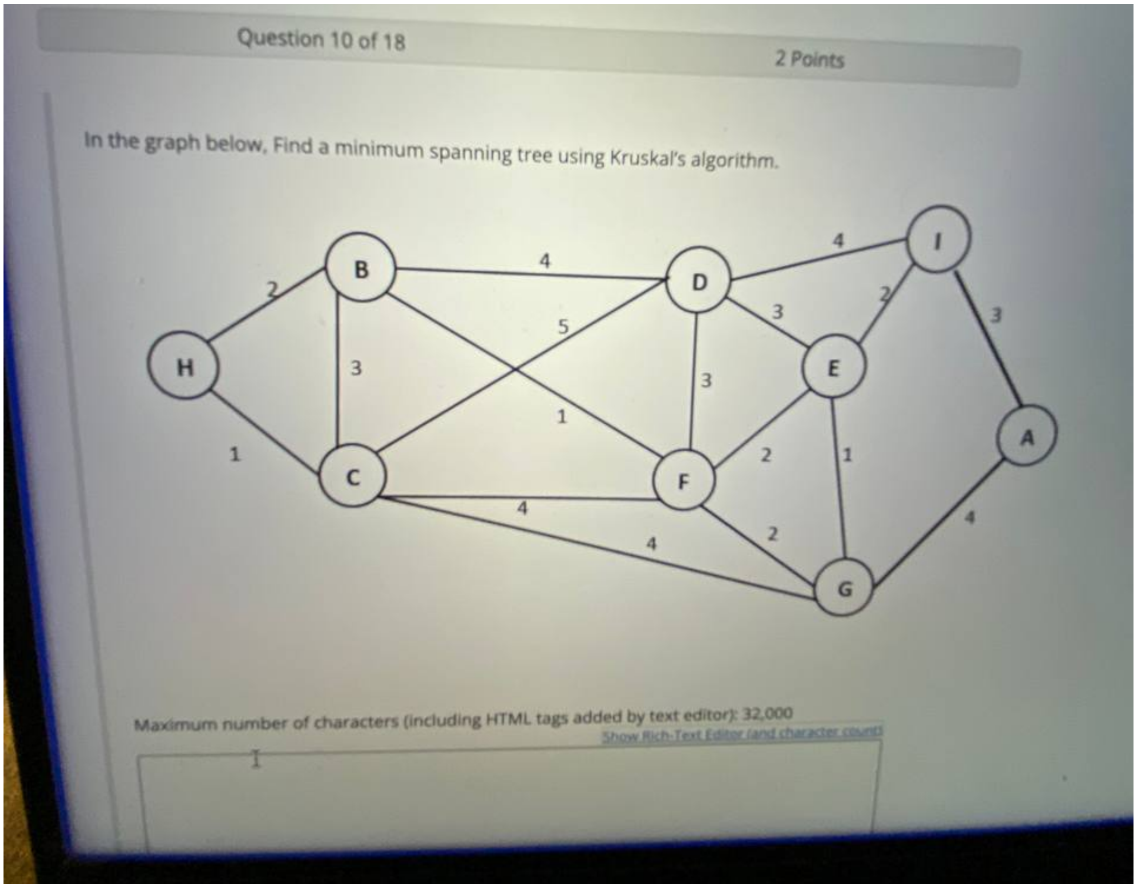
4. Return the distance stored in dist[X][Y]. If this is -1, it means there is no path from the source to the destination.

// code is

// if you will not modify the code !!!!!!!!!!!!!!!!!! Please don't write it ! description above is enough
// you will be marked as cheated

```
Int visited[][] = new int[n][m]

Bool valid(int i,int j){
	Return 0<=i && i<n && 0<=j && j<m;
}

Int dfs(int i,int j){

	Visited[i][j] = 1;

	if(i==x && j==y && mat[i][j] == 1)
		Return 0;
	if(!valide(i,j) || mat[i][j] != 1 || visited[i][j] == 1)
		Return 99999999;
	Int up = dfs(i-1,j);
	Int down = dfs(i+1,j);
	Int left = dfs(i,j-1);
	Int right = dfs(i,j+1);
	Return min(  min(up,down) , min(left,right) ) + 1;
}
```

Question 10:Mahi, Edi



The minimum spanning tree will have following connection
H -> C -> B -> F -> E -> G -> A-> I -> D

Q10 answer
- Minimum spanning tree contain edges : HB - HC- BF - FE- DE - EI - EG -IA
2 + 1 + 1 + 2 + 3 + 2 + 1 + 3 = 15

Kruskal's algorithm is a method for finding the minimum spanning tree (MST) of a connected, undirected graph. The algorithm starts with an empty MST and iteratively adds edges to the MST, selecting the edge with the smallest weight at each step. The algorithm stops when all vertices of the graph are included in the MST.

Here is an outline of the Kruskal's algorithm for finding the minimum spanning tree:

1. Sort all edges of the graph in non-decreasing order of their weight.
2. Create a disjoint-set data structure, initially each vertex is in its own set.

For each edge in the sorted list of edges, do the following:

3. a. If the vertices of the edge are in different sets, union the sets and add the edge to the MST.
4. Return the MST

Question 11:Tsinu, Ahmed

MUM Global Online Educatio | Algorithms Final Exam

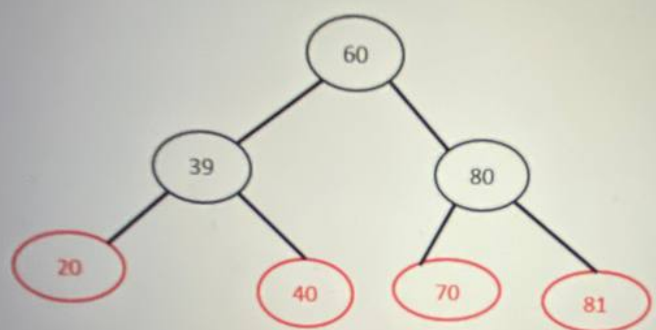
online.cs.miu.edu/samigo-app/jsf/delivery/deliverAssessment.faces

Time Remaining: 01:57:

▲ Hide Time Remaining

Question 11 of 18 | 4 Points

Which one of the following choices is correct while inserting 86 then 87 to the tree? Explain your choice for partial credit



A.

MUM Global Online Education × Algorithms Final Exam ×

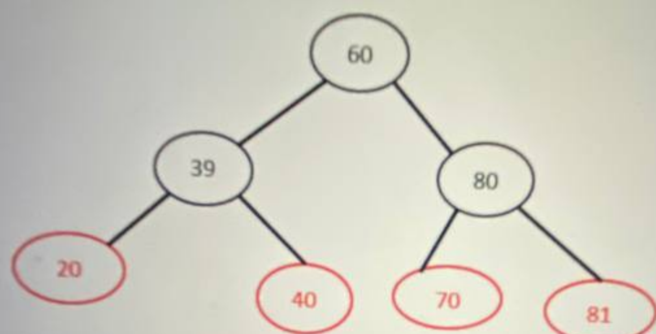
online.cs.miu.edu

Time Remaining

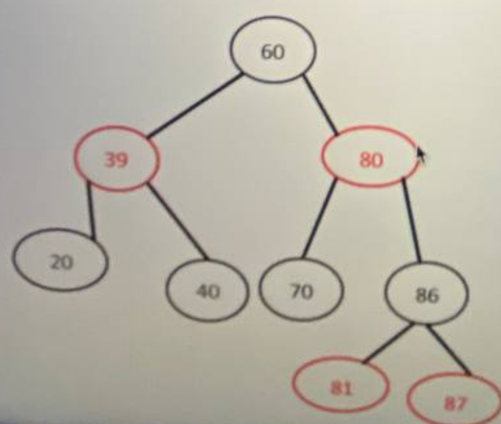
▲ Hide Time Rema

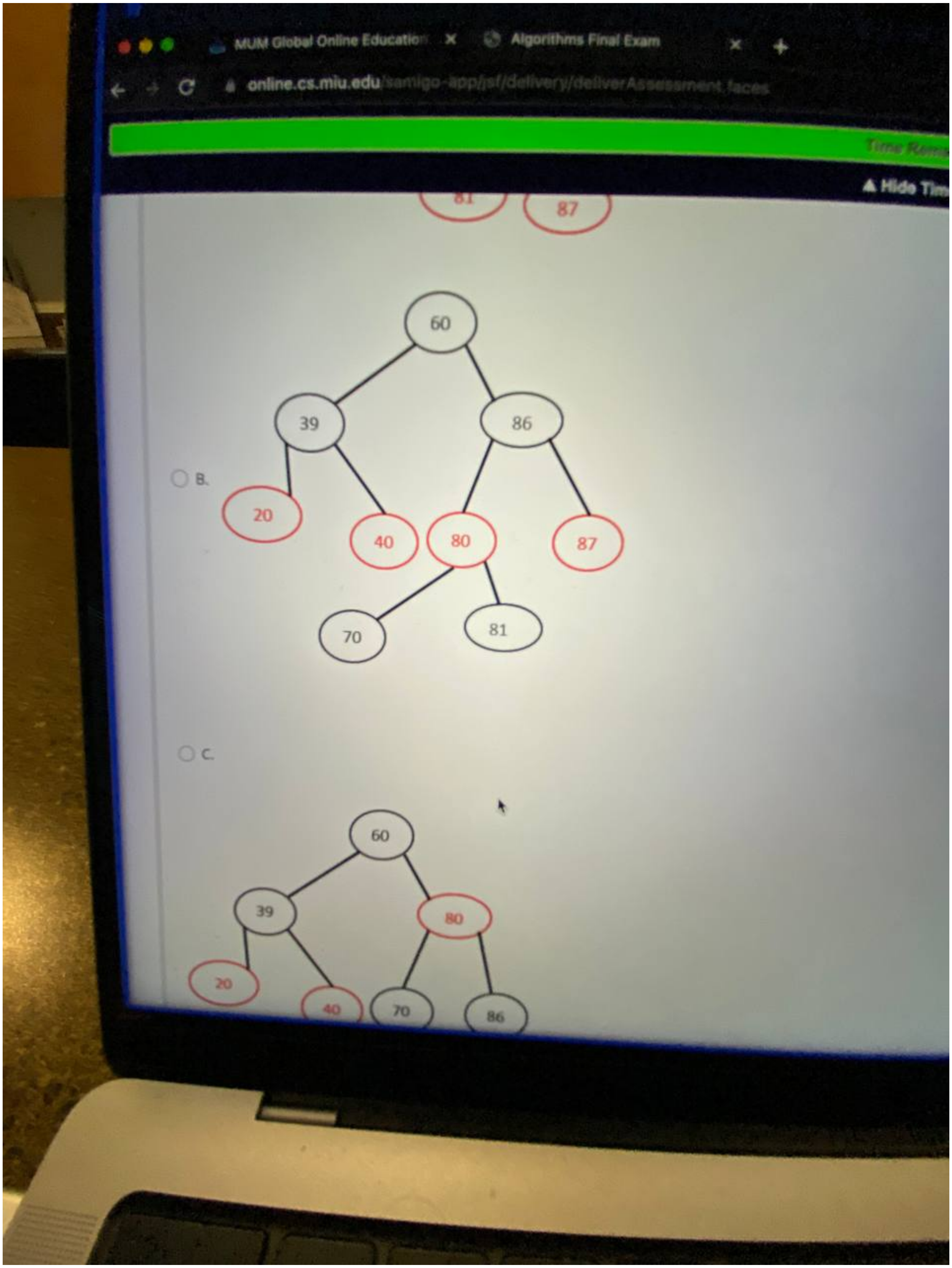
Question 11 of 18 4 Points

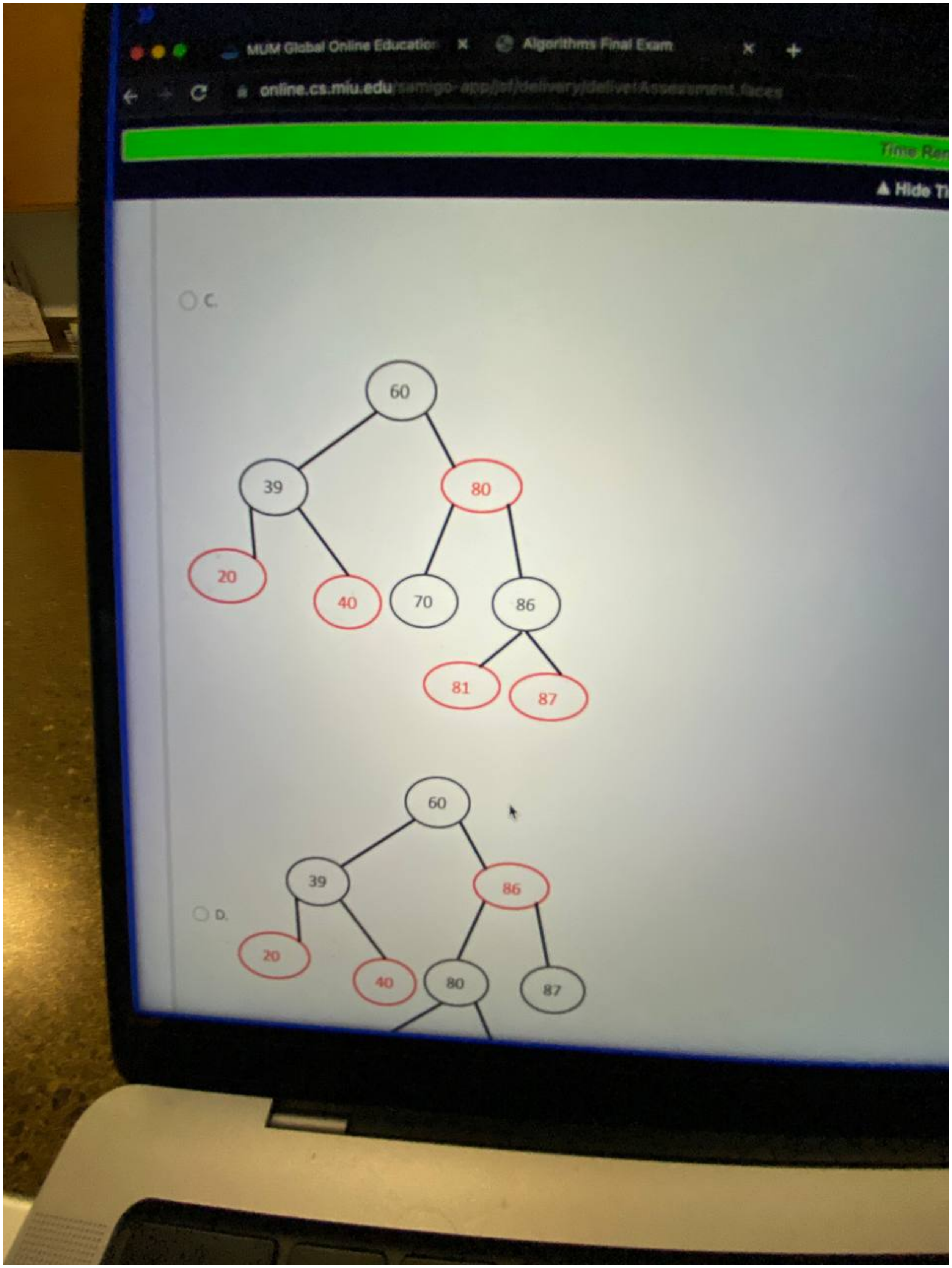
Which one of the following choices is correct while inserting 86 then 87 to the tree? Explain your choice for parti



A.




MUM Global Online Education
Algorithms Final Exam
online.cs.miu.edu/samigo-app/jsf/delivery/deliverAssessment.faces
Hide Tim
87
B.
60
39
86
20
40
80
87
70
81
C.
60
39
80
20
40
70
86


MUM Global Online Educatio
Algorithms Final Exam
online.cs.miu.edu
C.
60
39
80
20
40
70
86
81
87
D.
60
39
86
20
40
80
87

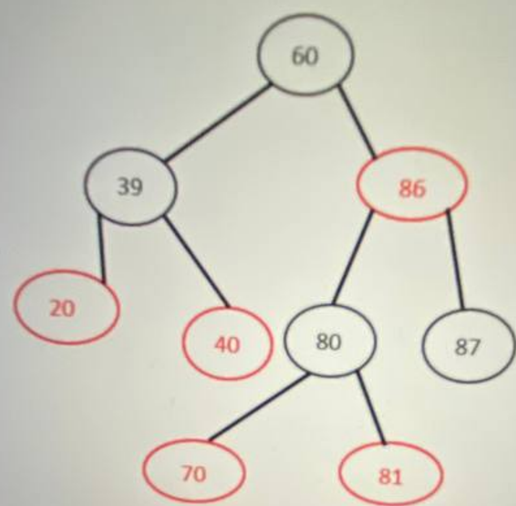
MUM Global Online Education   Algorithms Final Exam

online.cs.miu.edu/samigo-app/jsf/delivery/deliverAssessment.faces

Time R

▲ Hide

D.

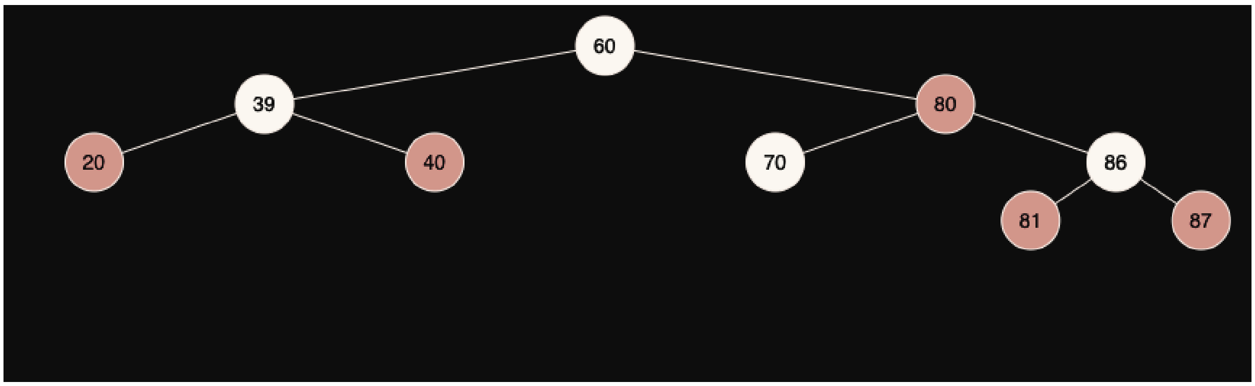


Reset Selection

Rationale:

Previous NEXT Save

Ans: C
Looking at 81, 86 and 87
The will be places as Grand Parent, Parent and Child all at the right line,
By rule on this condition, the Parent goes to the node and Grand Parent and Child will be on leaf.



Question 12:Tsinu, Ahmed

MUM Global Online Education × Algorithms Final Exam ×

online.cs.miu.edu/samigo-app/jsf/delivery/deliverAssessment.faces

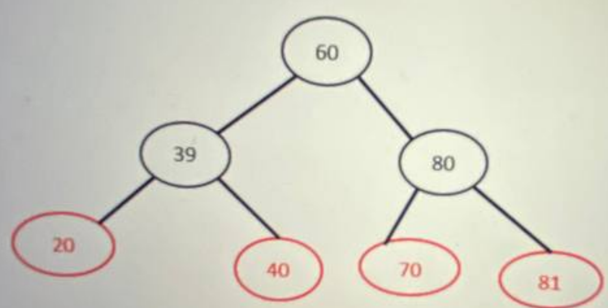
Time Remaining: 01:5

▲ Hide Time Remaining

Part 3 of 5 - Trees

Question 12 of 18 2 Points

Given the following Red /Black Tree. What is the black height of 60 in the given tree?

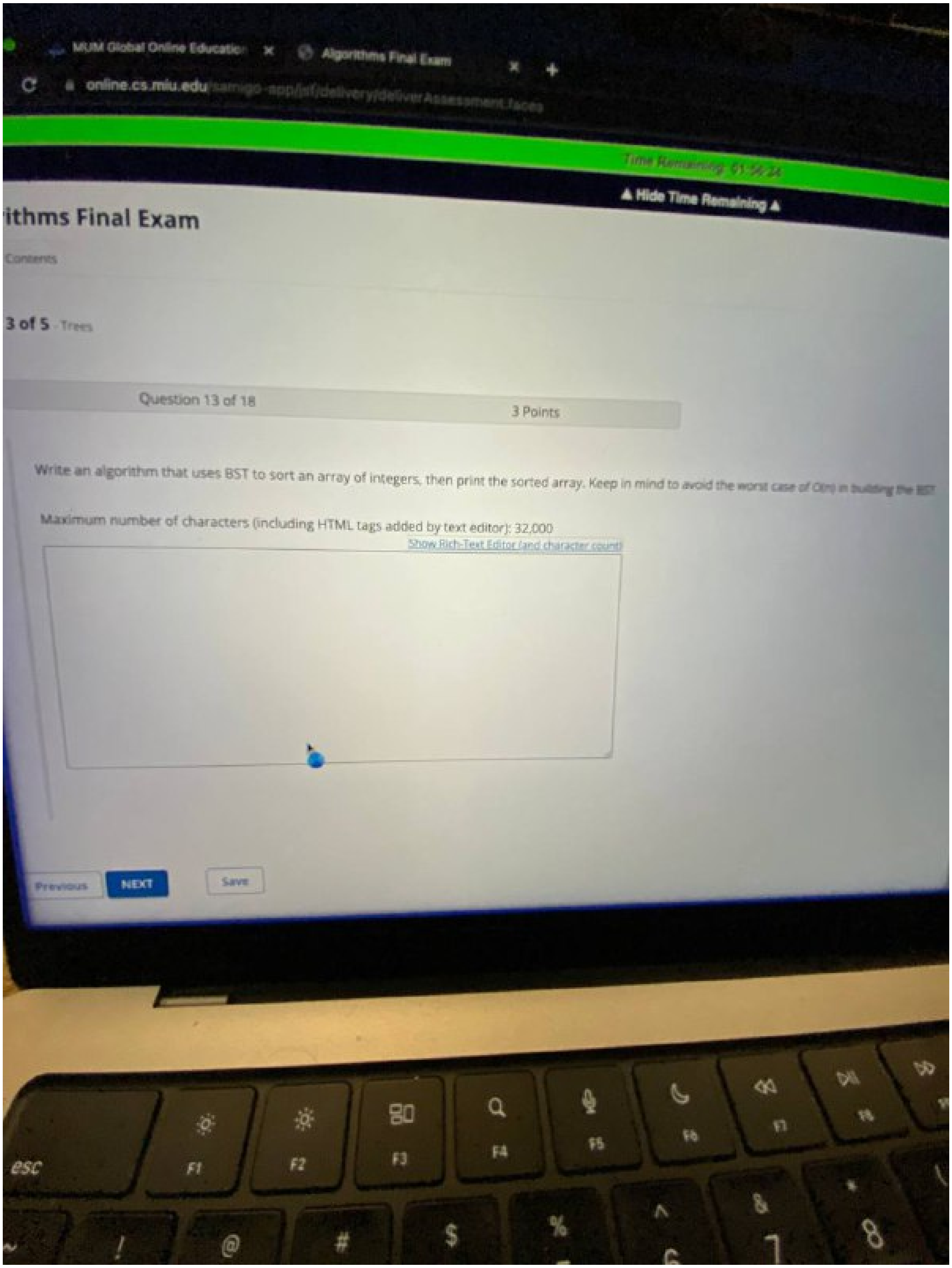


Maximum number of characters (including HTML tags added by text editor): 32,000

Show Rich-Text Editor (and character count)

Ans: The black height of 60 is 2, there are hidden black NIL leaves under the red ones.

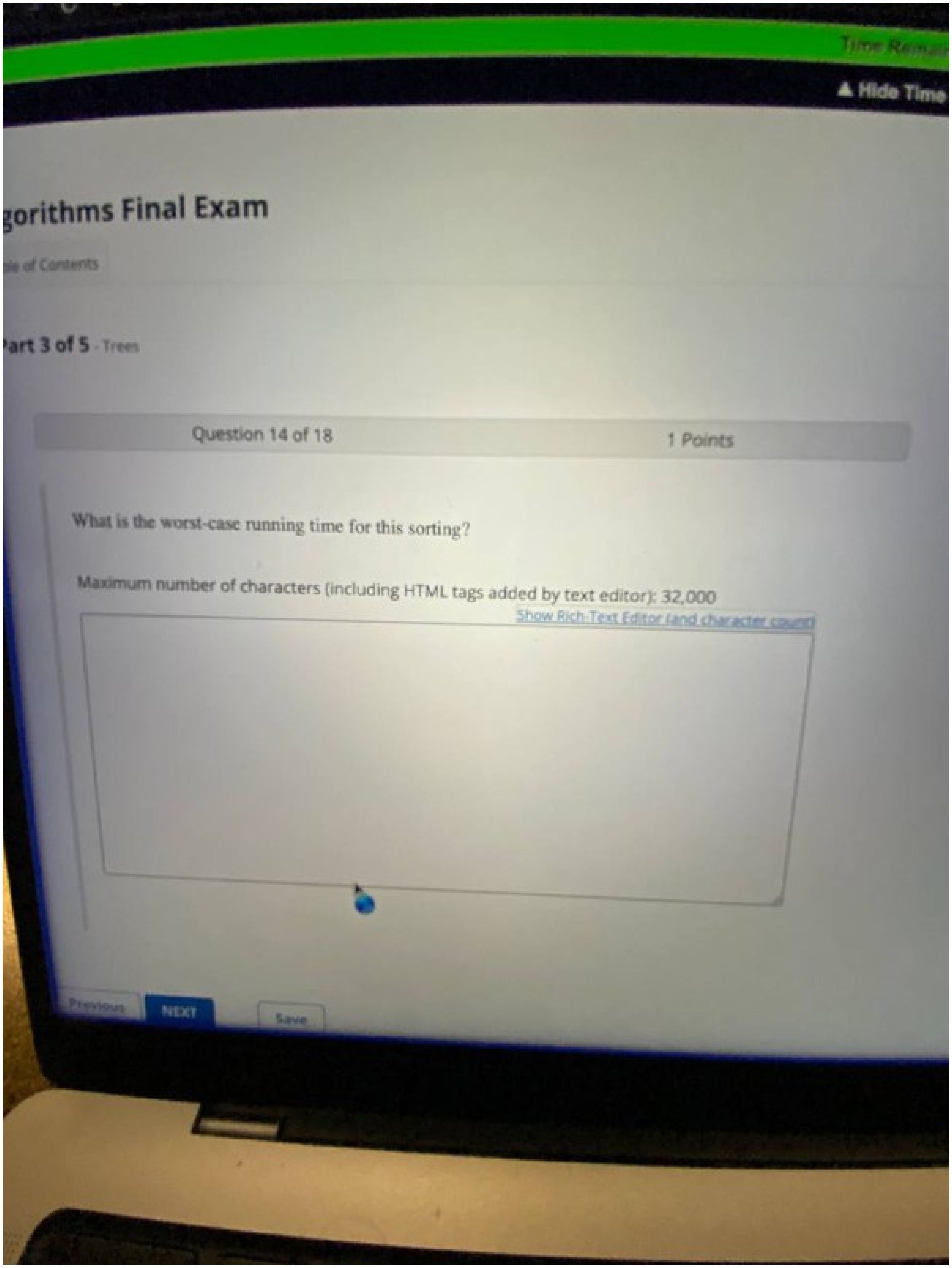
Question 13: Tusime Godwin, Maher Gerges


MUM Global Online Educatio
Algorithms Final Exam
online.cs.miu.edu
▲ Hide Time Remaining ▲
ithms Final Exam
Contents
3 of 5 - Trees
Question 13 of 18
3 Points
Write an algorithm that uses BST to sort an array of integers, then print the sorted array. Keep in mind to avoid the worst case of O(n) in building the BST
Maximum number of characters (including HTML tags added by text editor): 32,000
Show Rich-Text Editor (and character count)
Previous
NEXT
Save

1. Create an empty AVL tree.
2. Iterate through the array of integers and insert each element into the AVL tree using the AVL insert method.
3. Perform an in-order traversal of the AVL tree, which will return the elements in sorted order.
4. Initialize an empty array to store the sorted elements.
5. During the in-order traversal, add each visited node's key value to the empty array.
6. Print the filled sorted array.

```
void buildAVL(integer array, int start, int end)
        If start is grater than end then return;
        int middle <- (start+end)/2;
        root <- insert(root, array[middle]);
        buildAVL(array,start,middle-1);
        buildAVL(array,middle+1,end);

    Node insert(Node node, int key)
        if node isEqualTo null then
            return new Node(key);

        if (key is less than  node.key then
            node.left <- insert(node.left, key);
         else if (key is greater than node.key) then
            node.right <- insert(node.right, key);

        node <- balanceTree(node);
        return node;

    void inorder(Node node) {
        if (node is not equal to null) then
            inorder(node.left);
            print(node.key);
            inorder(node.right);
```

Question 14:Tusime Godwin, Maher Gerges



O(n log n)
This is because the AVL tree is a self-balancing BST, which means that it tries to maintain a balance between the left and right subtrees of every node in the tree. This helps to ensure that the height of the tree is always logarithmic with respect to the number of elements in the tree, which means that the search and insertion operations take O(log n) time.
The algorithm starts with an empty AVL tree and for each element in the input array, it performs an insertion operation in the AVL tree which takes O(log n) time. As there are n elements in the input array, the total time taken for the insertion operation is O(n log n).

After the AVL tree is constructed, the algorithm performs an in-order traversal of the tree, visiting each node once. The time complexity of an in-order traversal of a binary tree is O(n), as it visits n nodes.

Combining the time complexities of insertion and in-order traversal, the worst-case running time complexity of this algorithm is O(n log n).

OR

The worst-case running time of the above algorithm is O(n log n).

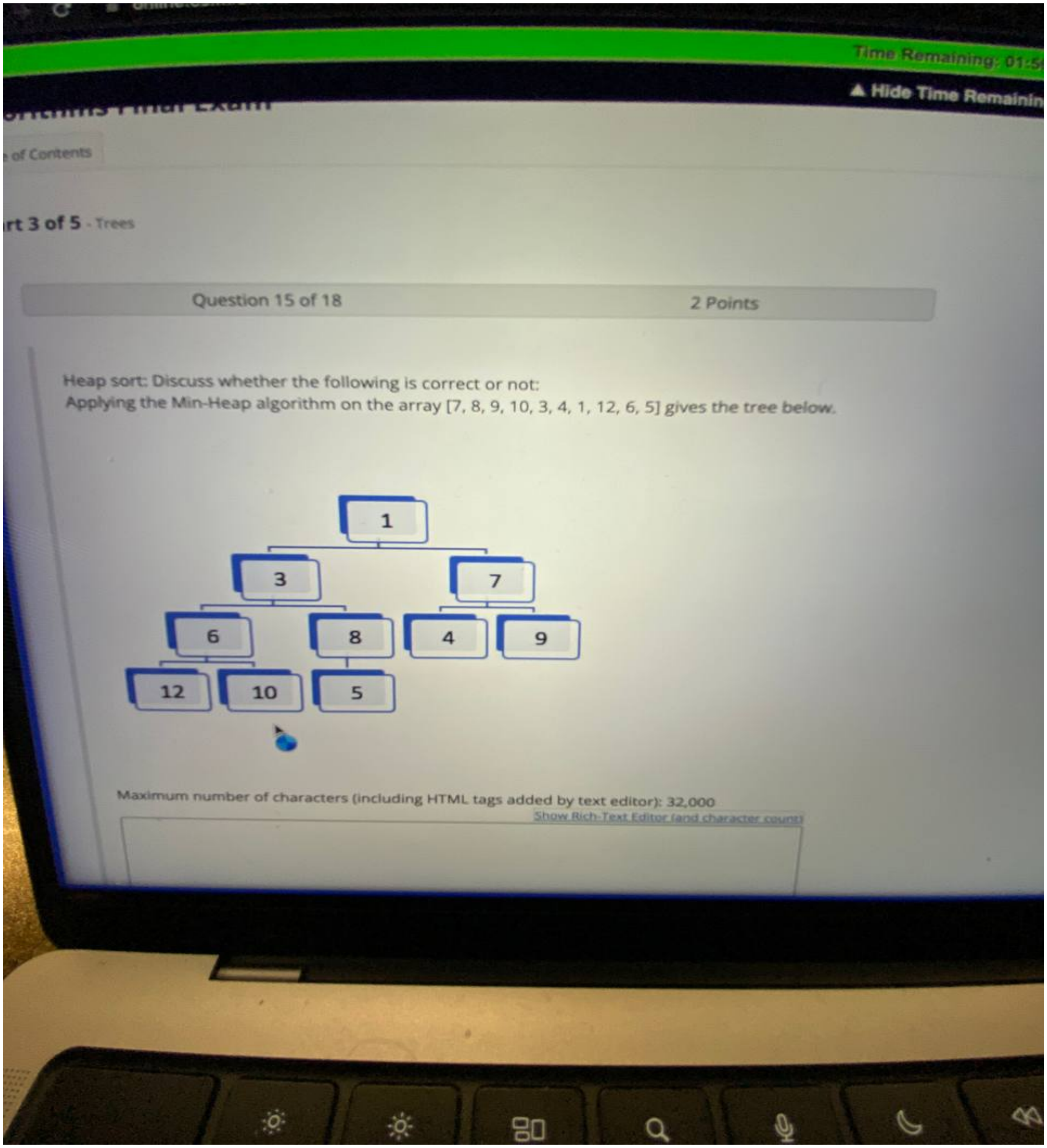
The time complexity of building the balanced BST using the divide-and-conquer approach is O(n log n) in the worst case. This is because at each level of recursion, the algorithm splits the input array into two equal halves and insert the middle element into the balanced BST.

The time complexity of performing an in-order traversal of a balanced BST is O(n) in the worst case, which occurs when the tree is completely balanced. This is because each node in the tree is visited exactly once and the time spent visiting each node is constant.

Finally, the time complexity of printing the sorted array is O(n), which is the number of elements in the array.

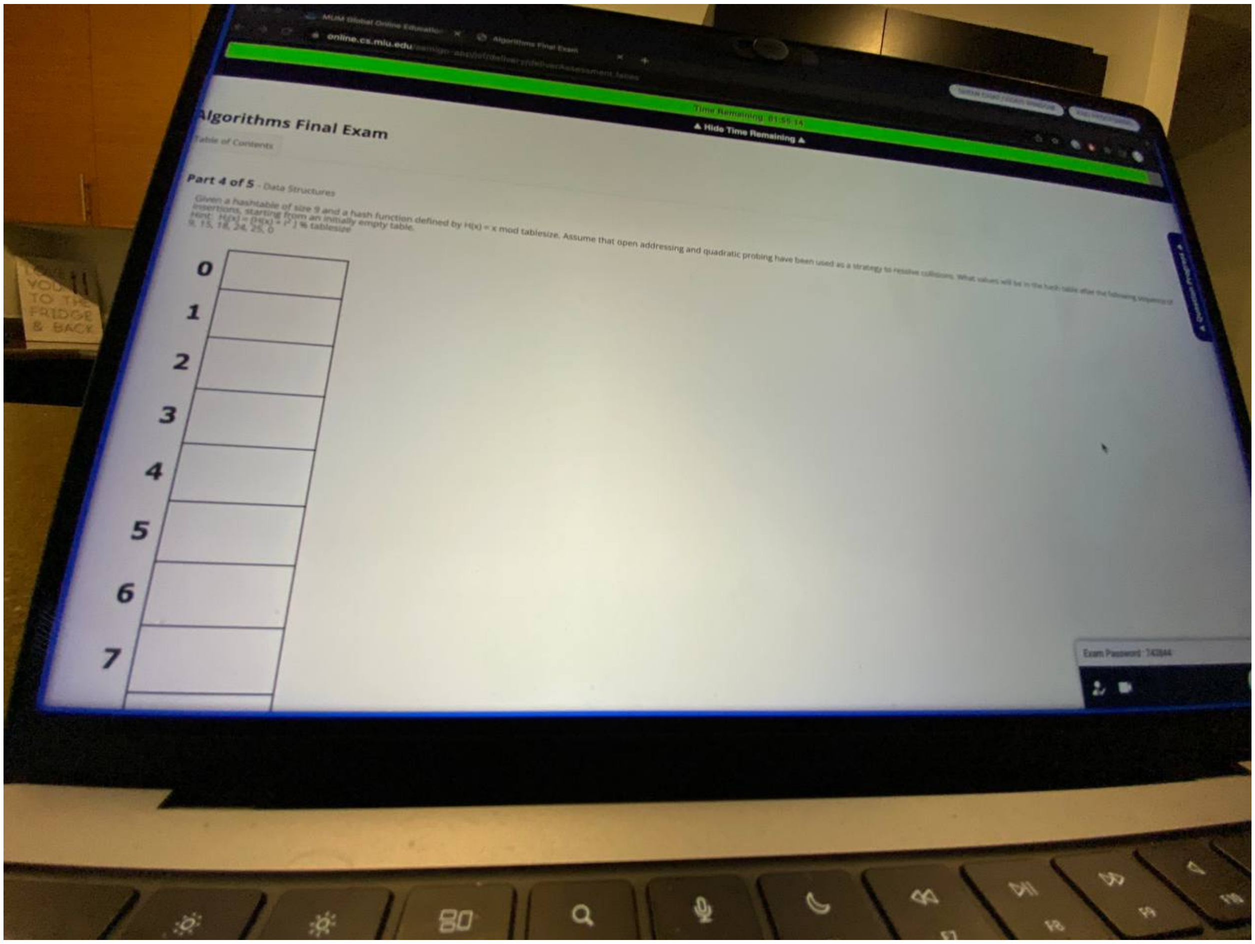
Therefore, the total worst-case running time of the algorithm is O(n log n) + O(n) + O(n) = O(n log n).

Question 15: Abenezer, Megdi



This is not valid Min heap , because in Min heap always parent is less than children , and we have 8 is parent and 5 is the children , and 8 > 5

Question 16:Abenezer, Megdi



MUM Global Online Education × Algorithms Final Exam × +

online.cs.miu.edu/samigo-app/jsf/delivery/deliverAssessment.faces

Time Remaining

▲ Hide Time Re

8

Question 16 of 18 4 Points

Fill in the values of table above based on the sequence described below. Show your work for partial credit.

9, 15, 18, 24, 25, 0

Maximum number of characters (including HTML tags added by text editor): 32,000

Show Rich-Text Editor (and character count)

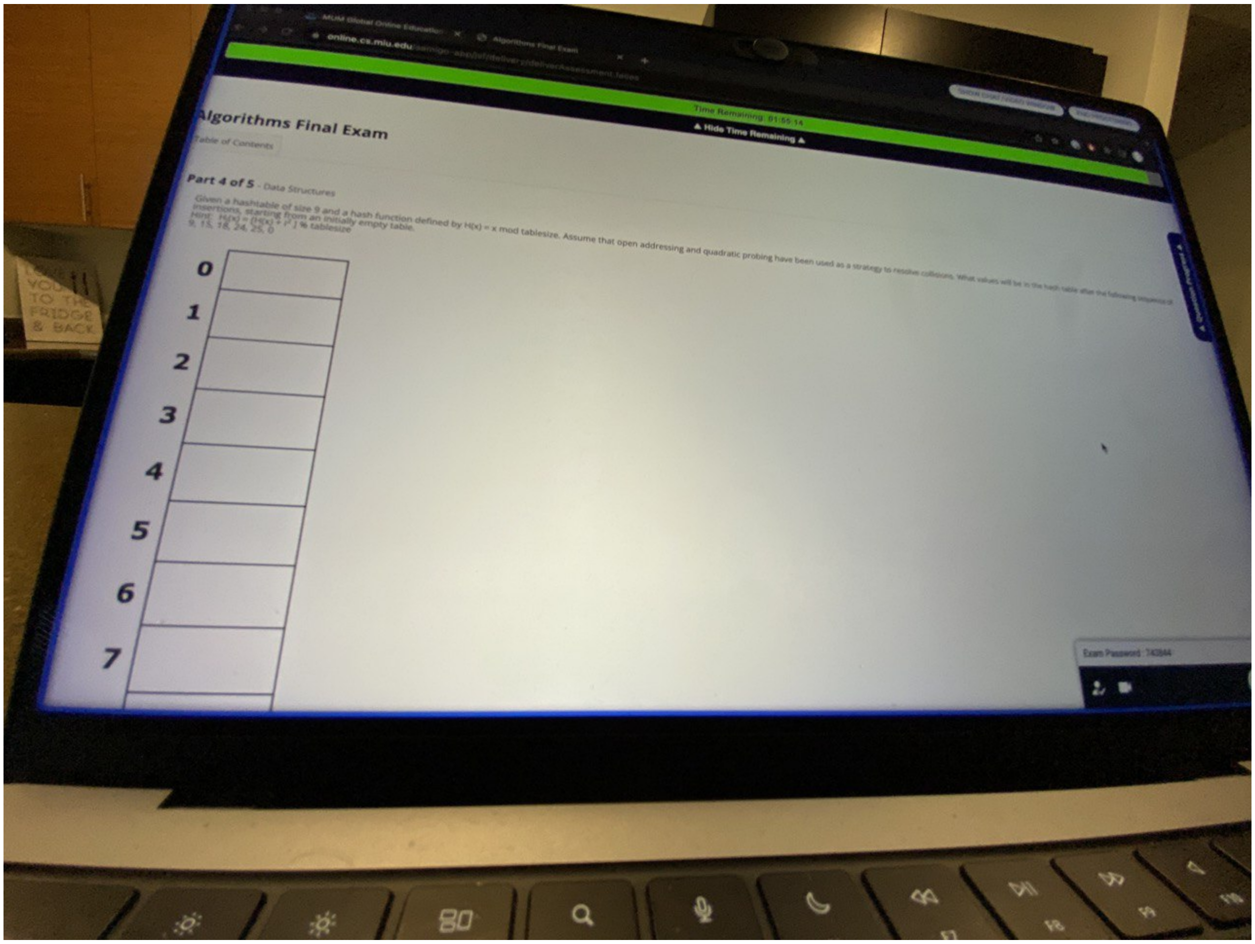
Previous NEXT Save

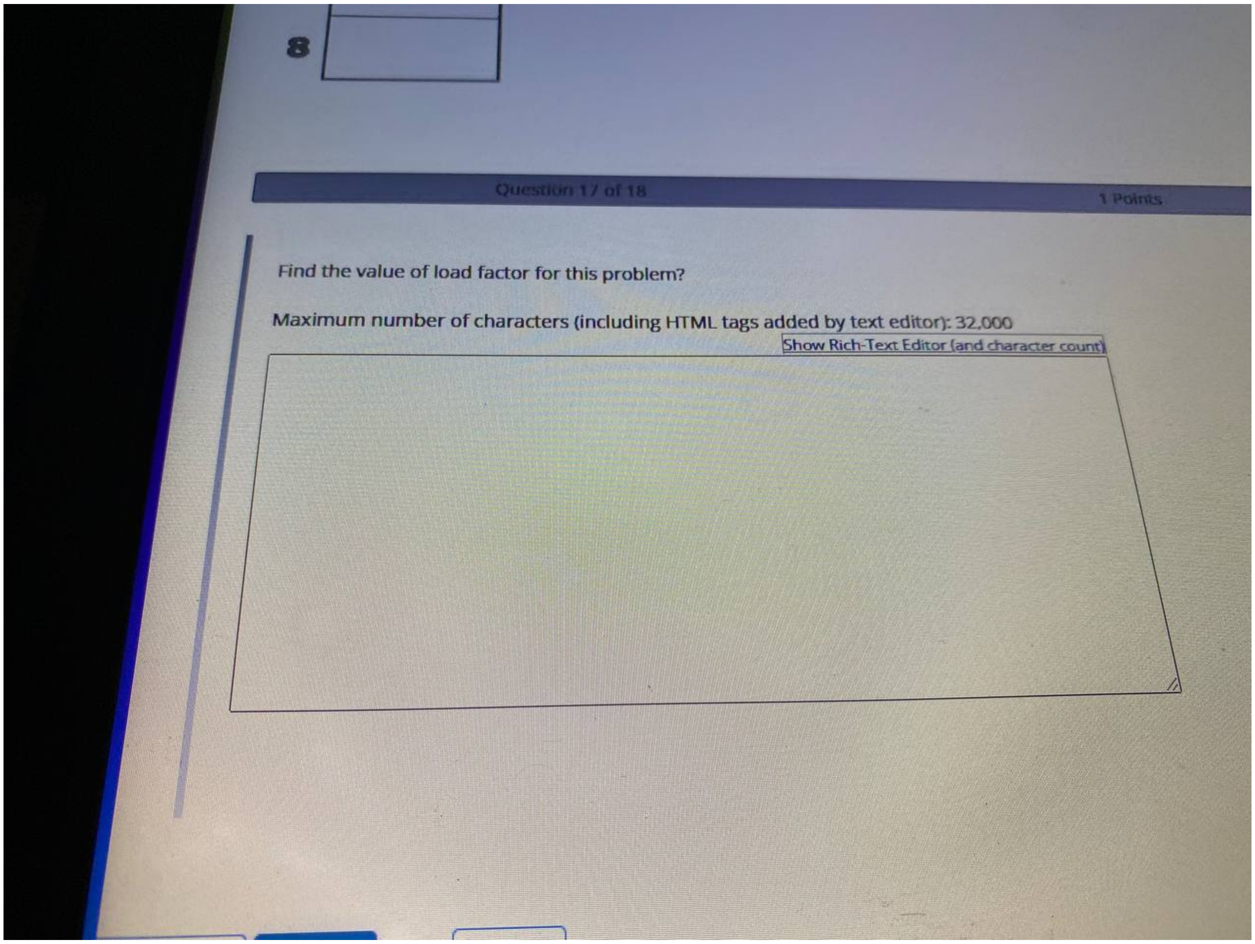
The final table will look like this
| 9 | 18 | 25 | 0 | | | 15 | 24 | |

0	9
1	18
2	25
3	0
4	
5	
6	15
7	24
8	

Explanation
First Value 9 => 9%9 = 0, so 9 goes to 0 index
Second Value: 15 => 15 % 9 = 6, so goes to 6th index
Similarly,
18 => 18 % 9 = 0, but 0 is occupied,
In this case we use quadratic probing to find next available index
(0 + 1^2) % 9 = 1, 1 is empty so it goest to index 1
24 => 24 % 9 = 6, 6 is occupied by 15,
(6 + 1^2) % 9= 7, so it goes to 7th index.
For value 25=> 25 % 9 = 1, 1 is occupied so
(1 + 1 ^ 2) % 9 = 2, so it goes to index 2
For value 0, 0 %9 = 0, which is occupied
(0 + 1^2) % 9 =1
(0 + 2^2) % 9 = 4
1 is occupied but 4 is empty so 0 goes to 4

Question 17: Mihreteab, Maher Gerges


online.cs.miu.edu
Time Remaining: 01:55:14
▲ Hide Time Remaining ▲
Algorithms Final Exam
Table of Contents
Part 4 of 5 - Data Structures
Given a hashtable of size 9 and a hash function defined by H(x) = x mod tablesize. Assume that open addressing and quadratic probing have been used as a strategy to resolve collisions. What values will be in the hash table after the following sequence of insertions, starting from an initially empty table.
Hint: H(x) = (H(x) + i²) % tablesize
9, 15, 18, 24, 25, 0
0
1
2
3
4
5
6
7
Exam Password: 743844
YOU TO THE FRIDGE & BACK



Load factor = number of elements / size = 6 / 9 = 0.66

The load factor for a hash table is the ratio of the number of elements stored in the table to the number of total slots in the table. It is represented by the formula:

load factor = number of elements in the table / table size

In this problem, we have a table of size 9 and we inserted 6 elements into the table. So the load factor for this problem is:

load factor = 6 / 9 = 2/3 = 0.66

Question 18 : Godwin, Tsinu, Tes

MUM Global Online Education × Algorithms Final Exam

online.cs.miu.edu/samigo-app/jsf/delivery/deliverAssessment.faces

Time Remaining: 0

▲ Hide Time Remai

Algorithms Final Exam

Table of Contents

Part 5 of 5 - (5 points) SCI Question

Question 18 of 18 | 5 Points

Write a short essay that connects one of the studied topics to the Science of Creative Intelligence (SCI). You can pic

Maximum number of characters (including HTML tags added by text editor): 32,000

Show Rich-Text Editor (and character count)

Previous NEXT Save SUBMIT FOR GRADING