***DFS algorithm: If we explain it by coding, then it can be in periods in graphs, unlike trees, so I used the visited boolean array to avoid visiting a node more than once.***

***Change the above Solution: Note that the application I wrote prints only the peaks that can be obtained from a given hill. For example, if the edges 0-3 and 0-2 were removed, the program would only print 0. To print all the peaks of the graph, we must Print to DFS for each un visited hill.***

***I CAN WRITE IN VERY FORMAL CODES ABOUT DFS, BUT THE MOST IMPORTANT PART OF THIS TOPIC I UNDERSTAND WAS RECURSIVE. I TRIED TO GET TO KNOW YOU AS WELL. I USED DATA STRUCTURE FOR JAVASCRIPT FOR THE FIRST TIME. AS SOON AS I HAVE SEEN ONLY ANIMATIONS IN THE FRONT-END APP, ETC. LET ME KNOW IF I HAVE ANY MISTAKES. THANK YOU FOR READING.***

***• Algorithm:***

***0. Knots created a stack and visited the array.***

***1. Insert the root into the stack.***

***2. Run a bend until the stack is empty.***

***3. Remove the item from the stack and print the item.***

***4. Mark the node for each adjacent and unseen node of the current node and enter it in the stack.***

***For example-***

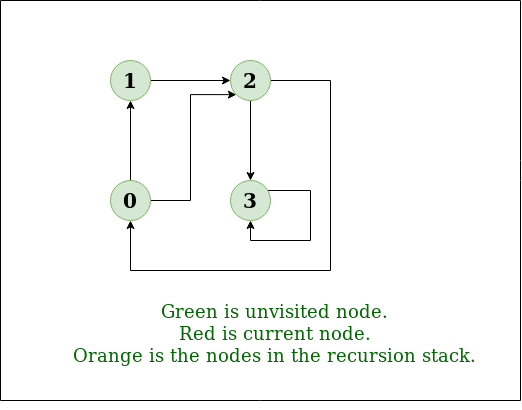
***1) We entered- n = 4 and e = 6***

***0 -> 1, 0 -> 2, 1 -> 2, 2 -> 0, 2 -> 3, 3 -> 3***

***Output: DFS from the 1st hill: 1 2 0 3***

***Explanation:***

***DFS Diagram:***

******