pages. If pages were stitched at random, then you would not be able to do this. This pages would have been stitched randomly.

Binary search works just like I took the example of the book. do 238 page number so what is the first condition of binary search , the condition is , should be sorted array Okay. Sorted array Array must be sorted , okay. Array needs to be sorted this is its first condition. You can add any number of elements to a sorted array. So I 'll keep track of three things here: Low, High, and Mid. One will keep the track of high , will keep track of mid. So here I am making a table Low , High , and Mid , OK So I would say now my mid is low And my high is what it is , okay And now I will calculate the new mid. I calculate the mid (8+4) /2= 6 My mid turns 6 , so this will be my mid , so it will be the mid. Linear vs Binary Search: For you guys Surely everyone must read this, access this , Downloading from the description , you will get the link somewhere. When I upload the video , it takes me some time to put the notes on the website. I do everything according to my own , right here And I try to do everything in time , put the video and then after that here I have the notes.

I did not talk about this , here it is a little important to tell it So look at this , which is our Linear Search and Binary Search : What do we people have to do , if you are looking at linear search , then its complexity is O (n) But if you look here binary search So I 'm halving the array until it runs out , does not converge speak the same (log n) So here you people will not face any issue , okay So this thing must have been cleared to you guys , I hope. int searchindex = linear search. Do return 1 means I have got this element here in the array or I will do one thing that I will return only the index of the array where I have. Otherwise, what will i do otherwise i will return -1 So let 's see here whether it works or not. As soon as the element is found, while traversing this array Then you return the index where that element was found. Otherwise what you do is -1 If we come to the end while searching in this array then you return -1. As this return ; is written , the function terminates and returns The function 's activation record in this stack gets lost.

The greatest integer will by default the (C) language. If you will do (5 + 6) /2 , Then it would be 11/2 After that it will give you 5 automatically. C language Since it is an integer , the operation between 2 integers will be an integer. So you do n't care about the greatest integer Directly , even if you write like this , It will automatically take that greatest integer. 5 point something will not be this value for 5 and 6 This value will be 5 only. So I 'm not going to search inclusive from here I will search from 100 to 400 in consideration , and repeat the same thing with him. So I will do it (mid+1) because i do n't mean to take it inclusive , I know : not even here : element Elements are from here to here right now , well, I know that So from here till here my element is there. If low and high are converged it means the element which is not available to me. If I do n't get the element then I will return -1. If return-1 needs to be executed it just means that the return did n't happen inside this while loop it has not returned inside the while loop. This means it needed to come out and return it means i did n'get the element inside the array. So let 's run this man for an array.

Introduction to Linked List in Data Structures

pass let 's assume i 'm deleting at index number 2 , Okay So from here i 'll show you the 0 and 1 i have written are the index written here.

It to the next of the head i have selected these 2 pointers one p is here , one q is here and i 'll keep moving forward both p and q. If I want this 8 to be deleted then I have to move it once both p. Then q will come to 8 , I want to delete where q comes you guys can understand So why am I doing this here , you will understand I 'll run a for loop. Once your q comes over there, where , where you want to delete that element Once q comes , then what do you do , you make p 's next to q 's. You guys understand how this is happening here free what i have done here, free i have. Free q means that the q which is there has become free. As soon as your linked list reaches the last element, what you have to do is delete it. So how will you do that, what will you delete from index? Delete at key or delete the value i. e. until the value is n't found till then you will keep deleting from the linked list.

Link is a game of complete links, whose game is it ? Link 's game it is. If you have made the links correctly, all these manipulations have been done, then you 'll have fun. Here we will see how it will look like if I delete the last one. Let me replicate it here and here I will delete it last and make it head. If you have reached the end of a linked list while searching for data, what will you do now ? It may also happen that that data is not in the linked list. So you have to check one more condition that and at the same time the next of q is not NULL as well q 's next should not be NULL either. I think I have told for the fifth time , no I will probably tell this for the tenth time i told it 5 times in previous videos. I want all the people to access this playlist , still there are very few numbers according to which people are watching according to me. Not everyone has accessed the playlist and would like to say thank you to those people who have shared the playlist on instagram.

Circular Linked List and Operations in Data Structures

circular linked list is one concept that people get confused about circular linked lists. I want to tell you this thing all things are same , this all are one concept do n't understand all this in three different ways. I am giving one practical example if you have seen my jango. If anybody is saying about linked list and he showed you insertion. He showed you search , traversal. He said that we can do insertion here we. can do deletion , can. do search , can do. search and can do traversal we moving forward till when we do. n't get next null starting from the head we do head 's next , that's next. If you have watched my jango 's video so I have made an E-commerce website and there I have shown you that how items can come like this. I have not tell you practical use of linked list but I gave you example that how this things can be useful so here this is circular linked list. When you do like this then item will turn and keep turning suppose after 4 , 5 is hiding so suppose you clicked on this then this you can watch 5 from here. in loop If I put while loop here and till where I will move ? I move till P 's data does not equal to head , P "s next is not equal. to null when eventually P will reach here while moving so P will be equal to. head. so I can do is write it after this while loop that print the. P's data where ever is P put one print here print that too which is missed P. s data. and what happens with do while loop is , what you do you should print one time.

doubly linked list here and one of the uses of it you guys can reverse it easily by swapping the pointers. At the same time, if you have a pointer to the last node then you can walk in both this direction and in this direction. There are other uses too I will also make you guys get the practice sets of linked list etc. in it : you have extra information in this , what is the node with your previous : you can also know this. A lot of people have access but still the number of people who picked up the course : not everyone has access from here I want all to access , to bookmark. I hope you guys must be easy to understand because you guys have accessed : This playlist If you have n't done it then definitely you will have a little problem if you're a beginner. bookmark this playlist by clicking here Click here to save

I prefer to use singly linked list if I feel like it will be more feasible. If I have memory constraints or memory constraints i will be happy with my singly link list. You come in a doubly linked linked list when you may have to go back and forth in both directions. The best thing is that a pointer who is traveling here : That can change his mood and start traveling here. I want to give a challenge to you guys here. I want you guys to write a function that first traverses it in this direction then once your pointer reaches here it comes back and traverses again that is , once a straight linked list is being printed here. Then you see people getting the reverse linked list printed. The person who is in realistic time can finish this course can finish it. We will wrap the topic of linked list and come later we will return to linked list And after that we will see the people who : this , but you have to code the doubly linked list. I am putting it in all the codes nowadays I have taken the feedback very well : Thank you guys So let 's what we will do now.

Introduction to Stack in Data Structures

Today we are going to see stacked data structure. We will talk about what is stacks ? And the notes that I have made for you I am going to give you as a PDF download. What is stack ? Stack is a linear data structure that is similar to a box. If you are from India, here I am talking about Indian railway. And if you get inside it people say the train is full. And even if someone tries to come forcefully Then the thing that happens is called stack overflow. So when that condition happens over here Then we say that the stack is full over here. Now, you ca n't add elements inside it ok. If you try to enter it Like how some people hang to the train which is not recommended by the way ok. It is not at all recommended. The one to come out first If I make plan to remove all items So the first item that will come out That will be item no. 4. You can call it as break if you find the item name strange. So break number 4 It will come. out first. And break number 3 after it break number 2. And the one that went first will. come out last. So it is also called FILO. Along with LIFO it is. also called as FILO and the one. that went inside last that will be out first so it will be last out.

Stack is very useful in designing such systems. Our second application of stack is Infix in postfix conversion. We will talk about this further. But here I will now want to talk about , our 3rd application Parenthesis matching. Sometimes we need to match parenthesis in an expression. If one mathematical expression is written, In which parenthesis are used , So is that mathematical expression valid or not ? Parenthesis in it are matched properly or not? In order to create a stack we need a pointer to the topmost element along with other elements. Which are stored inside the stack. We need a place to store those elements. And all these

elements can be our int datatype If you are using c++ than this can be objects. But for learning purpose Here we will use integer datatypes. So the capacity of your stack According to that , This can be full or not. So the operations that I have defined in stack ADT Whoever will do implementation of my stack by watching this ADT He is very clear what he wants to do. First operation that I said is push. Which means I want to add a element to the stack. Then you add element in that stack. And if I have to remove any element out Then in this way I will pop out the element. I will remove it. So I can easily remove it out. But I am not removing.

Stack can be implemented using an array or a linked list. Stack is a collection of elements with certain operations Flowing LIFO (last in first out) discipline. If our stack is full Then it means we ca n't add more elements to it. If stack is empty It means we remove elements from it. It has no element inside. So these two functions determine that That is our stack empty or full ok. Those who have read it, tell to others also. And if you share the playlist Then I will feel good. Sadly very less people in fact two people have shared playlist. Ayush did it and someone named Gautam did it. So thank you from the bottom of my heart. If you share then it will reach to many students.

Implementing Stack Using Array in Data Structures

Moving forward with our data structure algorithm discussion we will see how to implement stack with the help of array. If you have not seen my last video then in that video I have told you what stack data structure is. and why we would want to use this type of data structure. I will come back here and in our discussion I will show you stack. Top is the value of index. It is not a pointer. Top 's default value will be -1 if there are no elements. If there are two elements in this then top will be 1. Means my stack is filled up till here. And I will delete this and I will use this array only , my array is holding two elements so my top is filled in here in stack. The top value will be 0 if my stack is filled till here, 1 if till here and 2 if there is no element in this and with that I 'll write s. size is equal to 80. The default value is -1 but it will store the index of topmost element. This way I have implemented my stack. Can create array then can set top and then my stack will be ready.

We use integer as data type because it is the simplest way to understand things. But we can create custom data types stack too. It can be that whatever you are putting inside stack can be classes of C++. So I want to ask you one thing that how will you pop and push an item ? Then what you will do ? So which method will be best for me ? Most of the operations are performed in O (1) Operation can be complicated too. So I will write most of the. operations here. In O (1) means most of. operations are done in constant time. Now suppose I want to add 8 in it so I write it here and I will add 1 in this so this will become 1. And I did my push operation so we chose this and not this. So for pop operation then give me II if I show you pop operator what it does. To implement stack with the help of array what I did first ? First of all I created a fixed size array you can see here. In this stack , I created array , size and top which tells us the size of array. Then I created stack and kept its size 80 so that if I need to do a big number of push operations then my stack can handle it without problem so this is how we implemented stack here. I hope this is clear to you and this is my structure which I have created.

full. So when I am pushing what will I see ? I will see isempty ? or isfull ? So I am making some changes in implementation modifying it to make it better. Let 's do one thing I will print it that if isfull then printf stack overflow. This is the stack here which is shown. Our bottom most element is 78 and topmost element is 8. So see first 8 than 7 then 78 because I am pushing the elements here. Pop an element from this stack and tell me what it is. Pop of top means the topmost pointer of my stack/linked list. If You 're watching this directly then it was fast and I will say your linked list is not clear. When you are pushing this you are updating top again and again. But when you are sending top to pop then the changes you make in top are a pointer variable If it is changing in function its value is changing. If I make top 8 or 9 or 10 will this change this top? If I want it to change then I will send it 's address. But here I will have to treat top as a pointer of a pointer. So it will get dereference first and then it's next and this will definitely work. So here it is showing pooped element is % d lets add \n too.

I have given all the source code to you. And I have told you everything here and How is this linked list working and representing stack that also I have explained. Now I have n't showed you stacktop , stackottom and peek operation. So I am leaving this on you , you make functions for it. And more stack topics which we will discuss. This was it for today 's video. Thank you so much for watching this video. And I will see you next time. This is the weekly Newsquiz. Check out the latest video from iReport. com every day at 10 a. m. ET. Back to the page you came from.

peek(), stackTop() and Other Operations on Stack Using Linked List (with C Code)

We discussed about stack with the help of linked list in last video. Today we will discuss the remaining operations which were left. This number which I have inserted here is in which way the 28 number is in the last here. And then I can write 18 like this here and 15 like this and all these are inside our stack. So the top most element is this and hence we call it as top as we discussed. Top is nothing else but head but referring as top in case of stack. Because the topmost element of our stack is 7 so I am saying as top not head. PTR is not equal to null and I++ Inside this loop we will increment our pointer. If someone move it , if you want to get till 5 then you will move 4 times your PTR will become null. Stack bottom is homework and you have to comment. I have uploaded 30 videos here and 31st video I am creating. 31st_stackop2ll. c Op2ll. Op2 is how peek operation on stack of a linked list runs. This was a peek operation I hope you understood it.

The first operation was peek which is kind of important so looked carefully. I will make stack node * PTR and will equal it to top as I did in onenote. So, I 'll make a for loop and will use two conditions in it. And here int I is equal to zero , I is less than pos -1 and with that I will write and PTR is not equal to null. at position % d is here I and I. And here \n , after this lets run it. So it will be kind of traversal program which I wrote here. And I am keep on peeking it. One more thing I can do while peek will not return I -1 till then I will peek position. So here it is showing 7 on 0 position.

Parenthesis Matching Problem Using Stack Data Structure (*Applications of Stack)*