12/03/2023

(+)	Determine whether and			
\mathcal{L}	becommend which or word	w is	Soxted in	alcending
		đ		ascer, our of
	order or not.	-	4 - 1 - 1	,

1/p → 10 20 30 0/p → True

We will be linearly traversing the array & compare ith and (i+1)th element and if the (i+1)th element is less than ith element, then return false otherwise by recursion we go to the next index. Hence we have simply solved one case & rest recursion will see.

Base case => 9f we traversed full array, then return true as it is sorted that's why false was not returned.

Code

bool checkSorted (int aux [], inti, intsize) {
//Base Case

if (i = = size -1) → Traversed array return true; as i+1 is not valid index

if (our [i] > our [i+1])//Icase solve

return false:

// Recursive relation return checkSorted (aur, i+1, size)

3

Of we have written i = = size as the base case (ase, then when i = size-1, base case

ocanneu wiin call

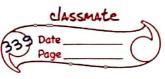
Scarined with CaM

	is not satisfied and hence 1'+1 i'e size-1+1
	Will be size & our [size] is not a valid
	index and hence the base case was written
	as i = = size-1. So last index that will be
	compared in the processing will be i+1 i.e
	compared in the processing will be i+1 i.e size -2+1 = size-1 (last index) and iie
	Size-2 which are valid indexes.
Note	Void, then we can simply write
	void, then we can simply write
. 11.00	(return) checkSorted ();
	return checkSorted ();
(2)	Binary Search using Recursion.
	I I I I I I I I I I I I I I I I I I I
	We have already seen the iterative sold and dry run of binary search. Hence we need to convert iterative to the
	and dry run of binary search. Hence
\	we need to convert iterative to the
\	recursive approach.
	an in the second
	Code
	int search Element (vector (int>4 our, ints)
1	int e, int target) {
	// Base case
1	jf(s>e)
	return - 1: 1/ Element not found
	// 1 case 50/ve
	int mid = $S + (e-s)/2$;
	if (arr [mid] = = target)
	return midi



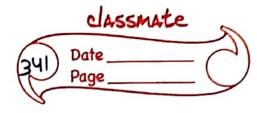
	Page
	// Recursive relation
	else if (arr [mid] > target) {
	return Search Element (aur, 5,
	m'd-1, target); 3 m'd-1, target); Search in
	else {
	return search Element (aur, mid+1, e,
	target); Search
	3 in right
	Si i n va. Luigava gierbre att trak
	Cases in binary search
1	J. J. d. L.
	aur [mid] = = target > return mid.
3)	
3)	at the left side so update end = mid-1;
	arr [mid] < target -> The element is present
	at the right side so update start = mid + 1 i
No	te + labor was can entry is maintained in the
	call stack , this means that memory has
_	been allocated to variables inside the function.
_	S. S. M.
_	Tree 7 10 20 30 40 50 Rey = 20
_	1/
_	S Le interno man interno
_	10 20 1 - index is returned
	m 1 - 11 dex 15 teconted
	20 m
	e

3 Subsequences of a string. By this question, we will be able to understand recursion better. i/p→ "abc" O/p→ Subsequence is basically including some characters & excluding some characters but the ordering remains same as that of i/p string. a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c b c c a b c a b c b c c a b c b c c b c c b c c c c c c c c c		
By this question, we will be able to understand recursion better. I/b -> "abc" O/b -> Subsequence is basically including some characters & excluding some characters but the ordering remains same as that of i/p string. a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c A c X x x x a X x x x a X x x x a X x x x a X x x x a X x x x a X x x x a X x x x a X x x x b X x x a c Mo of subsequences in the string of length n = 2n. Note -> The above question is based on the pattern of include & exclude and this pattern is widely used in the questions Approach a b c y o/p string	3	Subsequences of a string
Understand recursion better. i/p → "abc" O/p → Subsequence is basically including some characters & excluding some characters but the ordering remains same as that of i/p string. a b c abc abc abc xxxxx xxxx xxxx bc xxxxx bc xxxxx cc xxxx bc xxxxx cc xxxxxx		By this question, we will be able to
i/b → "abc" O/b → Subsequence is basically including some characters & excluding some characters but the ordering remains same as that of i/b string. a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c b c c a b c a b c a b c b c c b c c c c c c c c c c c c	10	understand recursion better.
Subsequence is basically including some characters & excluding some characters but the ordering remains same as that of i/b string. a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c b c c b c c a c c The above question is based on the pattern of include & exclude and this pattern is widely used in the questions Approach a b c b c c c c c c c c c c c c c c c c	, .t	
Subsequence is basically including some characters & excluding some characters but the ordering remains same as that of i/b string. a b c a b c a b c a b c a x x x a x x x x b x x x c x x x x		i/p→ "abc"
Subsequence is basically including some characters & excluding some characters but the ordering remains same as that of i/p string. a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c a b c b c		0/b →
Enaracters a excluding some characters but the ordering remains same as that of i/p string. a b c abc abc x x b x x x c x x x x	3	
Enaracters a excluding some characters but the ordering remains same as that of i/p string. a b c abc abc x x b x x x c x x x x		Subsequence is basically including some
Sut the Ordering remains same as that of ilp string. a b c abc x ab x x x a x x x x b x x x c x x b c x x ac Mo of subsequences in the string of length n = 2n. Mote - The above question is based on the pattern of include & exclude and this pattern is widely used in the questions Approach abc y o/p string		characters a excluding some characters
a b c a b c a b c a b c a b c a b c a b c x x x x a x x x x c x x x c x x x c x x a c Mo of subsequences in the string of length n = 2n. Mote - The above question is based on the pattern of include & exclude and this pattern is widely used in the questions Approach a b c y o/p string		but the ordering remains same as that
a b c Abc Abc X X A X X X X X X X X X X X X Ac Moof subsequences in the string of length n = 2n. Note - The above question is based on the pattern of include & exclude and this pattern is widely used in the questions Approach a b c {} i=0 b o/p string.		of 1/p string.
X X X X A X X X X B X X X X C X X X A C X X A C X A C X X A C X A		lases in highly suract
X X X X A X X X X B X X X X C X X X A C X X A C X A C X X A C X A		U b C
X X X X B X X X C X X D C X		
X X X B X X X C X X D C X X A C X		
Mo. of subsequences in the string of length $n = 2^n$. Note - The above question is based on the pattern of include 4 exclude and this pattern is widely used in the questions Approach abc i=0 Approach b o/p string	tors -	The state of the s
Mo. of subsequences in the string of length $n = 2^n$. Note - The above question is based on the pattern of include 4 exclude and this pattern is widely used in the questions Approach abc i=0 Approach b o/p string		X
Mo. of subsequences in the string of length $n = 2^n$. Note - The above question is based on the pattern of include 4 exclude and this pattern is widely used in the questions Approach abc i=0 Approach b o/p string		XXVC
Mo of subsequences in the string of length $n = 2^n$. Note -> The above question is based on the pattern of include 4 exclude and this pattern is widely used in the questions Approach abc bo/p string		
Mo of subsequences in the string of length $n = 2^n$. Note -> The above question is based on the pattern of include 4 exclude and this pattern is widely used in the questions Approach abc i=0 > O/P string		Vixxivac
Note -> The above question is based on the pattern of include & exclude and this pattern is widely used in the questions Approach a b c i=0 Approach b o/p string		
Note -> The above question is based on the pattern of include & exclude and this pattern is widely used in the questions Approach a b c i=0 Approach b o/p string		no of subsequences in the string of
Approach a b c i=0 Approach b o/p string	7	length n = 2 hill said
Approach a b c i=0 Approach b o/p string	\C -	T alama and i
Approach a b c i=0 Approach b o/p string	Note-	The above question is based on the pattern
Approach abc i=0 Approach i=0 Approach i=0 Approach i=0 Approach i=0		of michael & exclude and this pattern is
abc {} i=0 Light of the string		Widely used in the questions
abc {} i=0 Light of the string		Abbroach
i=0 > 0/p string		
		i=0 4 0/p string
		Scarneu with can



	Output string is initially empty.
	i
	de la companya de la
	include / exclude
	i C
	7 a b c "a",
	, include l'exclude
	7 a b c "ab", a b c "a"
,	include exclude exclude
	include abc
Ì	
	STIMPLY PATITE
	and return print print
	- 1 1 1 - Cla out bosso
	The above was for the left subtree.
	a b c exclude
	abc ""
	i exclude
	include 7" " F
	include l'exclude include exclude
	The same of the sa
	(b) (c)
	"bc" т т
	Drint Print Print Print
	Print
	The above was the right subtree.
	L'ans will be call for
7	Lest part of each subtree ith index
	K Lest part of each subtree will be call for include the character at ith index
	Scarned With Call

*	Right part of the subtree will be the
	call for exclude
*	Include means concatenate & exclude
	means bring the string from above level
	as it is.
	We have stop as i moves out of the
	string and hence this will be the base case condition.
	case condition.
	Cod
1 1 1	Code
	Void brint Sub Sea Cobins of at
	Void print SubSeq (string str, string output, inti) {
	str. length() if (i = =) { //Base Case
	if (i = =) { //Base Case
	// Print string & then return cout << output << endl;
	cout << output << endl;
	returni
	// Exclude call (As it is)
	hairt Sut Soo (Ola VIII)
	printSubSeq (str, output, (°+1)); // Include call (Concatenate)
	output. bush back (str(i));
	print Sub Seq (Str, output, 1+1);
•,	2 sty output, 1+1)
	3
	What is the use of output string? Output string is basically a temperory string & we use this because we don't want to change in original string as it will lead to loss of characters
	Output string is basically a tembernau
	String & we use this because we don't
	want to change in original string as it
	WILL read to 2055 of characters.
	Scarned with Carr



Note	e → vector <int>v j q gives runtime everore</int>
www.	V[0] = 5; Jas we don't know
	whether v [0] exists
	or not.
	To fix this problem we need to define The size.
	the ciace
	ITU SIZE
	vector <int> V (10); This will work</int>
	V[O] = 5;

Scanned with Cam