Câu hỏi 1

Chính xác

Điểm 1,00 của 1,00

Cờ câu hỏi

Implement function

```
int binarySearch(int arr[], int left, int right, int x)
```

to search for value x in array arr using recursion.
After traverse an index in array, we print out this index using cout << "We traverse on index: " << index << endl;

Note that middle of left and right is floor((right-left)/2)

For example:

Test	Result

`int arr[] = {1,2,3,4,5,6,7,8,9,10};`
`int x = 10;`
`int n = sizeof(arr) / sizeof(arr[0]);`
`int result = binarySearch(arr, 0, n - 1, x);`
`(result == -1) ? cout << "Element is not present in array"`
`: cout << "Element is present at index " << result;`
	`We traverse on index: 4`
	`We traverse on index: 7`
	`We traverse on index: 8`
	`We traverse on index: 9`
	`Element is present at index 9`

Answer: (penalty regime: 0, 0, 5, ... %)

Reset answer

Câu hỏi 2

Chính xác

Điểm 1,00 của 1,00

Cờ câu hỏi

Implement function

```
int interpolationSearch(int arr[], int left, int right, int x)
```

to search for value x in array arr using recursion.
After traverse to an index in array, before returning the index or passing it as argument to recursive function, we print out this index using cout << "We traverse on index: " << index << endl;

Please note that you can't using key work for, while, goto (even in variable names, comment).

For example:

Test	Result

`int arr[] = { 1,2,3,4,5,6,7,8,9 };`
`int n = sizeof(arr) / sizeof(arr[0]);`
`int x = 3;`
`int result = interpolationSearch(arr, 0, n - 1, x);`
`(result == -1) ? cout << "Element is not present in array"`
`: cout << "Element is present at index " << result;`
	`We traverse on index: 2`
	`Element is present at index 2`

`int arr[] = { 1,2,3,4,5,6,7,8,9 };`
`int n = sizeof(arr) / sizeof(arr[0]);`
`int x = 0;`
`int result = interpolationSearch(arr, 0, n - 1, x);`
`(result == -1) ? cout << "Element is not present in array"`
`: cout << "Element is present at index " << result;`
	`Element is not present in array`

Câu hỏi 4

Chính xác

Điểm 1,00 của 1,00

Cờ câu hỏi

Given an array of distinct integers, find if there are two pairs (a, b) and (c, d) such that a+b = c+d, and a, b, c and d are distinct elements. If there are multiple answers, you can find any of them.

Some libraries you can use in this question:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <algorithm>
#include <iostream>
#include <utility>
#include <map>
#include <vector>
#include <set>
```

Note: The function checkAnswer is used to determine whether your pairs found is true or not in case there are two pairs satistify the condition. You don't need to do anything about this function.

For example:

Test	Result

`int arr[] = { 3, 4, 7, 1, 2, 9, 8 };`
`int n = sizeof arr / sizeof arr[0];`
`pair<int, int> pair1, pair2;`
`if (findPairs(arr, n, pair1, pair2)) {`
`  if (checkAnswer(arr, n, pair1, pair2)) {`
`  printf("Your answer is correct.\n");`
`  }`
`  else printf("Your answer is incorrect.\n");`
`}`
`else printf("No pair found.\n");`
	`Your answer is correct.`

`int arr[] = { 3, 4, 7 };`
`int n = sizeof arr / sizeof arr[0];`
`pair<int, int> pair1, pair2;`
`if (findPairs(arr, n, pair1, pair2)) {`
`  if (checkAnswer(arr, n, pair1, pair2)) {`
`  printf("Your answer is correct.\n");`
`  }`
`  else printf("Your answer is incorrect.\n");`
`}`
`else printf("No pair found.\n");`
	`No pair found.`