Đã bắt đầu vào lúc	Thứ hai, 19 Tháng mười hai 2022, 4:27 PM
Tình trạng	Đã hoàn thành
Hoàn thành vào lúc	Thứ hai, 19 Tháng mười hai 2022, 4:29 PM
Thời gian thực hiện	1 phút 25 giây
Điểm	**6,00** của 6,00 (**100**%)

Câu hỏi 1

Chính xác

Điểm 1,00 của 1,00

Implement function

```
int foldShift(long long key, int addressSize);
int rotation(long long key, int addressSize);
```

to hashing key using Fold shift or Rotation algorithm.

Review Fold shift:

The **folding method** for constructing hash functions begins by dividing the item into equal-size pieces (the last piece may not be of equal size). These pieces are then added together to give the resulting hash value.

For example:

Test	Result
`cout << rotation(600101, 2);`	26

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

Reset answer

```
int foldShift(long long key, int addressSize)
{
    long long b=1;
    while((key/b)!=0){
        b=b*10;
    }
    int fold=0;
    while(key!=0){
        for(int i=0;i<addressSize;i++){
            b/=10;
            if(b==1) break;
        }
        int split=key/b;
        key-=split*b;
        fold+=split;
    }
    int c=1;
    for(int i=0;i<addressSize;i++){
        c=c*10;
    }
    return fold%c;
}

int rotation(long long key, int addressSize)
{
    long long b=1;
    while(key/b!=0){
        b=b*10;
    }
    b/=10;
    int du=key%10;
    key/=10;
    key=du*b+key;
    //cout<<key<<endl;
    return foldShift(key,addressSize);
}
```

	Test	Expected	Got	
✓	`cout << rotation(600101, 2);`	26	26	✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 2

Chính xác

Điểm 1,00 của 1,00

Implement three following hashing function:

```
long int midSquare(long int seed);
long int moduloDivision(long int seed, long int mod);
long int digitExtraction(long int seed, int* extractDigits, int size);
```

Note that:

In midSquare function: we eliminate 2 last digits and get the 4 next digits.

In digitExtraction: extractDigits is a sorted array from smallest to largest index of digit in seed (index starts from 0). The array has size **size.**

For example:

Test	Result

`int a[]={1,2,5};`
`cout << digitExtraction(122443,a,3);`
	223

`cout <<midSquare(9452);`	3403

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

Reset answer

```
long int midSquare(long int seed)
{
    long long square=seed*seed;
    square/=100;
    return square%10000;


}
long int moduloDivision(long int seed, long int mod)
{
    return seed%mod;
}
long int digitExtraction(long int seed,int* extractDigits,int size)
{
    int b=1;
    while(seed/b!=0){
        b*=10;
    }
    b/=10;
    int count=0;
    long int sum=0;
    for(int i=0;seed!=0;i++){
        int du=seed/b;
        //cout<<du<<endl;
        if(extractDigits[count]==i){
            sum=sum*10+du;
            count++;
        }
        seed=seed-du*b;
        b/=10;
    }
    return sum;
}
```

	Test	Expected	Got	

✓	`int a[]={1,2,5};`
	`cout << digitExtraction(122443,a,3);`
		223	223	✓

	Test	Expected	Got	
✓	`cout <<midSquare(9452);`	3403	3403	✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 3

Chính xác

Điểm 1,00 của 1,00

There are `n` people, each person has a number between `1` and `100000` (`1 ≤ n ≤ 100000`). Given a number `target`. Two people can be matched as a **perfect pair** if the sum of numbers they have is equal to `target`. A person can be matched no more than 1 time.

Request: Implement function:

```
int pairMatching(vector<int>& nums, int target);
```

Where `nums` is the list of numbers of `n` people, `target` is the given number. This function returns the number of **perfect pairs** can be found from the list.

Example:

The list of numbers is `{1, 3, 5, 3, 7}` and `target = 6`. Therefore, the number of **perfect pairs** can be found from the list is `2` (pair `(1, 5)` and pair `(3, 3)`).

Note:

In this exercise, the libraries `iostream, string, cstring, climits, utility, vector, list, stack, queue, map, unordered_map, set, unordered_set, functional, algorithm` has been included and `namespace std` are used. You can write helper functions and classes. Importing other libraries is allowed, but not encouraged, and may result in unexpected errors.

For example:

Test	Result

`vector<int>items{1, 3, 5, 3, 7};`
`int target = 6;`
`cout << pairMatching(items, target);`
	2

`int target = 6;`
`vector<int>items{4,4,2,1,2};`
`cout << pairMatching(items, target);`
	2

Answer: (penalty regime: 0, 0, 0, 5, 10, ... %)

Reset answer

```
int pairMatching(vector<int>& nums, int target) {
    int count=0;
    //vector<pair<int,bool>> v;
    sort(nums.begin(),nums.end());
    //for(unsigned int i=0;i<nums.size();i++){
    //    v.push_back(make_pair(nums[i],0));
    //}
    int start=0;
    int end=nums.size()-1;
    while(start<end){
        if(nums[start]+nums[end]>target){
            end--;
        }
        else if(nums[start]+nums[end]<target){
            start++;
        }
        else{
            count++;
            start++;
            end--;
        }
    }
    //while(start<end){
    //    if(v[start].first+v[end.first])
    //}
    return count;
}
```

	Test	Expected	Got	

✓	`vector<int>items{1, 3, 5, 3, 7};`
	`int target = 6;`
	`cout << pairMatching(items, target);`
		2	2	✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 4

Chính xác

Điểm 1,00 của 1,00

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

```
int binarySearch(int arr[], int left, int right, int x)
{
    if(right<left) return -1;
    int mid=(right-left)/2;
    int index=left+mid;
    cout << "We traverse on index: " << index << endl;
    if(arr[index]<x) return binarySearch(arr,index+1,right,x);
    if(arr[index]>x) return binarySearch(arr,left,index-1,x);
    else if(arr[index]==x)return index;
    else return -1;
}
```

	Test	Expected	Got	

✓	`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`
			`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`
				✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 5

Chính xác

Điểm 1,00 của 1,00

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.

Answer: (penalty regime: 0 %)

Reset answer

```
bool findPairs(int arr[], int n, pair<int,int>& pair1, pair<int, int>& pair2)
{
    // TODO: If there are two pairs satisfy the condition, assign their values to pair1, pair2 and return true.
    map<int,pair<int,int>> hash;
    for(int i=0;i<n;i++){
        for(int j=i+1;j<n;j++){
        int sum=arr[i]+arr[j];
        if(arr[i]==arr[j]) continue;
            if(hash.find(sum)==hash.end()){
                hash[sum]=make_pair(arr[i],arr[j]);
            }
            else {
                pair1=hash[sum];
                //cout<<pair1.first<<endl;
                //cout<<pair1.second<<endl;
                pair2=make_pair(arr[i],arr[j]);
                //cout<<pair2.first<<endl;
                //cout<<pair2.second<<endl;
                if(pair1.first!=pair2.first&&pair1.first!=pair2.second&&pair1.second!=pair2.first&&pair1.second!
                return 1;
                else continue;
            }
        }
    }
    return 0;
}
```

	Test	Expected	Got	

✓	`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.`	`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.`	`No pair found.`	✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 6

Chính xác

Điểm 1,00 của 1,00

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

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

Reset answer

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

    if (x < arr[left] || x > arr[right]) return -1;
    if(left<0) return -1;
    if(right<0) return -1;
    if (left > right) return -1;
    //if (left == right)
    //{
    //    cout << "We traverse on index: " << right;
    //    if (arr[right] == x) return right;
    //    return -1;
    //}

    int pos = left + (x - arr[left])*(double)(right - left)/(arr[right] - arr[left]);
    if(pos<0) return -1;
    cout << "We traverse on index: " << pos << endl;

    if (arr[pos] == x) return pos;

    if (arr[pos]<x)
        return interpolationSearch(arr, pos+1, right, x);
   else return interpolationSearch(arr, left, pos-1 , x);

}
```

	Test	Expected	Got	

✓	`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`
			`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`	`Element is not present in array`	✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

BÁCH KHOA E-LEARNING

WEBSITE

HCMUT

MyBK

BKSI

LIÊN HỆ

268 Lý Thường Kiệt, P.14, Q.10, TP.HCM

☎ (028) 38 651 670 - (028) 38 647 256 (Ext: 5258, 5234)

elearning@hcmut.edu.vn