Lecture 13 Searching and Sorting

1. <https://leetcode.com/problems/average-salary-excluding-the-minimum-and-maximum-salary/>

class Solution {

public:

double average(vector<int>& salary) {

int n = salary.size();

int maxE = INT\_MIN;

int minE = INT\_MAX;

double sum = 0;

for(int i=0; i<n; i++)

{

if(salary[i] < minE)

{

minE = salary[i];

}

if(salary[i] > maxE)

{

maxE = salary[i];

}

sum += salary[i];

}

double res = sum - maxE - minE;

return res/(n-2);

}

};

1. <https://leetcode.com/problems/valid-perfect-square/>

class Solution {

public:

bool isPerfectSquare(int num) {

long long int s = 1;

long long int e = num;

while(s<=e)

{

long long int mid = s + (e - s)/2;

if(mid\*mid == num)

{

return true;

}

else if(mid\*mid > num)

{

e = mid - 1;

}

else

{

s = mid + 1;

}

}

return false;

}

};

1. <https://leetcode.com/problems/search-insert-position/>

class Solution {

public:

int searchInsert(vector<int>& nums, int target) {

int n = nums.size();

int s = 0;

int e = n-1;

while(s<=e)

{

int mid = (s+e)/2;

if(nums[mid]==target)

{

return mid;

}

else if(nums[mid]>target)

{

e = mid-1;

}

else

{

s = mid+1;

}

}

return s;

}

};

1. <https://leetcode.com/problems/find-peak-element/>

class Solution {

public:

int findPeakElement(vector<int>& nums) {

int n = nums.size();

int s = 0;

int e = n-1;

while(s<=e)

{

int mid = (s+e)/2;

if((mid==0 || nums[mid] > nums[mid-1]) && (mid==n-1 || nums[mid] > nums[mid+1]))

{

return mid;

}

else if(mid < n-1 && nums[mid] < nums[mid+1])

{

s = mid+1;

}

else

{

e = mid-1;

}

}

return -1;

}

};

1. <https://leetcode.com/problems/largest-number/>

class Solution {

static bool compare(string a, string b)

{

return a+b>b+a;

}

public:

string largestNumber(vector<int>& nums) {

int n = nums.size();

bool zero = true;

vector<string> ans;

for(int i=0; i<n; i++)

{

ans.push\_back(to\_string(nums[i]));

if(nums[i]!=0)

{

zero = false;

}

}

if(zero)

{

return "0";

}

sort(ans.begin(), ans.end(), compare);

string res;

for(int i=0; i<n; i++)

{

res += ans[i];

}

return res;

}

};