ĐẠI HỌC BÁCH KHOA HÀ NỘI

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

----- 🙡 🕮 🙣 -----



**BÁO CÁO PROJECT I**

***Báo cáo tuần 1***

Giảng viên: **Ngô Lam Trung**

Sinh viên thực hiện: Nguyễn Quang Long

Mã số sinh viên: 20194320

**Hà Nội, năm 2021**

Nội dung đã hoàn thành:

**38. Concatenate Arrays**

a. Đề bài:

Cho 2 mảng số nguyên a và b, nối mảng b vào sau mảng a.

b. Giải pháp:

Sử dụng hàm có sẵn insert().

c. Code:

vector<int> concatenateArrays(vector<int> a, vector<int> b) {

    a.insert(end(a), begin(b), end(b));

    return a;

}

**39. Remove Array Part**

a. Đề bài:

Xóa 1 phần của mảng đã cho từ vị trí l đến r

b. Giải pháp:

Sử dụng hàm splice()

c. Code:

function removeArrayPart(inputArray, l, r) {

  inputArray.splice(l, r - l + 1);

  return inputArray;

}

**40. Is Smooth?**

a. Đề bài:

Cho 1 mảng, middle của mảng được xác định bằng cách:

-Nếu mảng chứa 1 số lẻ phần tử thì middle là phần tử có chỉ số giống nhau khi đếm từ đầu mảng hoặc cuối mảng

-Nếu mảng chứa 1 số chẵn phần từ thì middle là tổng của 2 phần tử có chỉ số hơn kém nhau 1 khi đếm từ đầu mảng và cuối mảng

Mảng được gọi là Smooth nếu phần tử đầu bằng phần tử cuối và bằng middle

Xác định xem mảng đã cho có Smooth không?

b. Giải pháp:

-Tìm phần tử middle của mảng

-Nếu phần tử đầu và phần tử cuối của mảng đều bằng middle thì mảng Smooth, ngược lại thì không.

c. Code:

function isSmooth(arr) {

    var mid;

    if(arr.length%2==0){

        mid = arr[arr.length/2] + arr[(arr.length/2)-1]

    } else {

        mid = arr[Math.floor(arr.length/2)]

    }

    return arr[0] == mid && mid == arr[arr.length-1]

}

**41. Replace Middle**

a. Đề bài:

Cho 1 mảng, middle của mảng được xác định bằng cách:

-Nếu mảng chứa 1 số lẻ phần tử thì middle là phần tử có chỉ số giống nhau khi đếm từ đầu mảng hoặc cuối mảng

-Nếu mảng chứa 1 số chẵn phần từ thì middle là tổng của 2 phần tử có chỉ số hơn kém nhau 1 khi đếm từ đầu mảng và cuối mảng

Nếu middle là tổng của 2 phần tử, thay thế 2 phần tử này bằng middle

b. Giải pháp:

Nếu mảng chứa 1 số chẵn phần tử:

-Tìm vị trí của 2 phần tử đứng giữa

-Tìm giá trị middle

- Thay thế 2 phần tử đứng giữa bằng middle

Còn nếu mảng chứa 1 số lẻ phần tử thì không phải làm gì

c. Code:

function replaceMiddle(arr) {

  if (arr.length % 2 === 0) {

    var midPos = arr.length / 2;

    var mid = arr[midPos] + arr[midPos - 1];

    arr.splice(midPos - 1, 2, mid);

  }

  return arr;

}

**42. Make Array Consecutive 2**

a. Đề bài:

Cho 1 mảng các số nguyên không âm

Cần thêm vào mảng bao nhiêu số nguyên để được 1 số nguyên liên tiếp từ phần tử bé nhất đến phần tử lớn nhất của mảng?

b. Giải pháp:

-Tính số lượng số nguyên từ phần tử bé nhất đến lớn nhất của mảng rồi trừ đi số lượng phần tử của mảng

c. Code:

function makeArrayConsecutive2(statues) {

    return Math.max(...statues)-Math.min(...statues)+1-statues.length;

}

**43. Is power?**

a. Đề bài:

Xác định xem số đã cho có phải là lũy thừa của một số nguyên không âm không?

b. Giải pháp:

Tính lũy thừa của các số từ 2 đến căn n rồi so sánh với n

c. Code:

bool isPower(int n)

{

    if (n == 1) return true;

    for(int t = 2; t <= sqrt(n); ++t){

        int k=t;

        while (k<n) {

            k=k\*t;

        }

        if(k==n) return true;

    }

    return false;

}

**44. Is sum of consecutive 2?**

a. Đề bài:

Tính số cách biểu diễn 1 số nguyên bằng tổng các số nguyên dương liên tiếp

b. Giải pháp:

-Tính tổng các dãy từ 1,2,.. đến n/2 rồi so sánh với n

c. Code

int isSumOfConsecutive2(int n) {

    int count =0;

    for(int i=1;i<=n/2;i++){

        int s=0;

        for(int j=i;j<=n/2+1;j++){

            s+=j;

            if(s==n) {count++;

                        break;}

        }

    }

    return count;

}

**45. Square Digits Sequence**

a. Đề bài:

Cho 1 số nguyên dương, phần tử đằng sau được xác định bằng tổng bình phương các chữ số của phần tử đứng trước, dãy dừng lại khi xuất hiện phần tử giống với phần tử đã có trong dãy. Tính số phần tử của dãy.

b. Giải pháp

Tìm phần tử tiếp theo của dãy, nếu phần tử này đã tồn tại thì dãy kết thúc còn không thì thêm phần tử vào dãy lại tìm phần tử tiếp theo. Kết quả là số phần tử của dãy +1.

c. Code

int check(int n){

    if(n<10) return pow(n,2);

    else return pow(n%10,2)+check(n/10);

}

int squareDigitsSequence(int a0) {

    int n=a0;

    set<int> sequence={n};

    while(1){

        n=check(n);

        if(sequence.count(n)) return size(sequence)+1;

        else sequence.insert(n);

    }

}

**46. Pages Numbering With Ink**

a. Đề bài

Tính số trang cuối cùng có thể đánh số cho sách biết trang hiện tại cần đánh là current và có thể đánh numberOfDigits chữ số nữa.

b. Giải pháp

Nếu số chữ số có thể đánh nhỏ hơn số chữ số của current thì trả về số trang đứng trước current, còn không gọi đệ quy với trang tiếp theo của current và số chữ số trừ đi số chữ số của current.

c. Code

int numberDegree(int n){

    if(n<10) return 1;

    else return 1+numberDegree(n/10);

}

int pagesNumberingWithInk(int current, int numberOfDigits) {

    if(numberOfDigits<numberDegree(current)) return current-1;

    else return pagesNumberingWithInk(current+1, numberOfDigits-numberDegree(current));

}

**47. Comfortable Numbers**

a. Đề bài