**045. MÃ GRAY**

* **Bài toán:**

Một hình tròn được chia làm 2n hình quạt đồng tâm, các hình quạt được đánh số từ 1 tới 2n theo chiều kim đồng hồ. Hãy chỉ ra một cách xếp tất cả số từ 0 tới 2n-1 vào các hình quạt, mỗi số vào một hình quạt sao cho bất cứ hai số nào ở hai hình quạt cạnh nhau đều chỉ khác nhau đúng 1 bít trong biểu diễn nhị phân của nó.

Ví dụ: Với n = 4:  
0 = 0000  
1 = 0001  
2 = 0010  
3 = 0011  
4 = 0100  
5 = 0101  
6 = 0110  
7 = 0111  
8 = 1000  
9 = 1001  
10 = 1010  
11 = 1011  
12 = 1100  
13 = 1101  
14 = 1110  
15 = 1111  
Dữ liệu: Nhập từ bàn phím số nguyên dương n. Giới hạn (1 ≤ n ≤ 20).  
Kết quả**:** Ghi ra File (of LongInt) GRAYCODE.OUT gồm 2n số nguyên kiểu LongInt theo đúng thứ tự từ số ghi trên hình quạt 1 tới số ghi trên hình quạt 2n.

* **Ý tưởng:**

Với mỗi số n ta sẽ phải biểu diễn cơ số 2 của số từ 1 đến 2n nhưng thứ tự của nó không phải tăng dần mà có sự đối xứng, chúng ta tìm ra quy luật đối xứng của nó

A[1]=0 đối xứng với A[n]= 2n

A[2]=1 đối xứng là A[n-1]=2n

Ta có quy tắc tính toán

Quy ước: với n = 1 là Gray thứ 1, n = 2 là Gray thứ 2

Gray thứ 2 là sao chép và đảo ngược của Gray thứ nhất, nửa đầu thêm 0 và nửa sau thêm 1

Như vật muốn tìm Gray thứ n ta phải tìm Gray thứ n-1.

Việc còn lại là biểu diễn nhị phân của dãy theo độ dài n là xong

* Code minh hoạ

#include <bits/stdc++.h>

#include <fstream>

using namespace std;

int main(){

string s[10000];

int n;

cin>>n;

int k =pow(2,n);

s[1]="0";

s[2]="1";

if(n>1){

int t =2;

for(int i = 2; i<=n; i++){

int l=pow(2,i);

for(int j= 1; j<=t; j++){

s[l-j+1] = "1" + s[j];

s[j] = "0" + s[j];

}

t= l;

}

}

ofstream f("GRAYCODE.OUT");

for(int j = 1; j<=k; j++){

cout<<s[j]<<" ";

f<< s[j]<<" ";

}

f.close();

}