به نام خالق یکتا

راه حل سوال های کانتست UICPC Round #20 (Div2)

سوال A: 707A

- سوال، یک سوال پیاده سازی است و تنها کاری که برای حل آن لازم است انجام دهیم این است که همه کارکتر های ورودی را بررسی کنیم و درباره رنگی یا سیاه و سفید بودن تصمیم بگیریم.

C++:

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

int main(){

        ll n,m;
        cin >> n >> m;
        n = m*n;
        bool is_c = false;
        for(ll i=0;i<n;i++){
            char c;
            cin >> c;
            if(c!='W'&&c!='B'&&c!='G')is_c=true;
        }
        if(is_c){
            cout<<"#Color"<<endl;
        }
        else{
            cout<<"#Black&White"<<endl;
        }

    return 0;
}
```

Python:

```python
m,n = input().split()
result = "#Black&White"
for i in range(int(m)):
    for j in input().split():
        if j not in ('B','W','G'): result = "#Color"

print(result)
```

سوال B: 1525B

- اگر آرایه سورت شده باشد به صفر حرکت نیاز داریم.
- اگر آرایه سورت شده نباشد اما عنصر اول آن برابر با ۱ یا عنصر آخر آن برابر با n باشد، تنها به یک حرکت برای سورت کردن نیاز داریم.(انتخاب بازه ای با n-1 عضو)
- اگر آرایه سورت شده نباشد و عنصر اول برابر با n و عنصر آخر برابر با ۱ باشد، به سه جرکت برای سورت کردن نیاز داریم.(چرا؟)
- در آخر در بقیه حالات به ۲ حرکت برای سورت کردن نیاز داریم.

C++:

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

int main(){
    ll t;
    cin >> t;
    while(t--){
        ll n;
        cin >> n;
        vector<ll> num(n);
        for(ll i=0;i<n;i++){
            cin >> num[i];
        }
        if(is_sorted(num.begin(),num.end()))cout<<0<<endl;
        else if(num[0]==1||num[n-1]==n)cout<<1<<endl;
        else if(num[n-1]==1&&num[0]==n)cout<<3<<endl;
        else cout<<2<<endl;

    }
    return 0;
}
```

Python

```python
for _ in range(int(input())):
    n = int(input())
    lst = list(map(int, input().split()))

    if sorted(lst) == lst:
        print(0)
    elif lst[0] == 1 or lst[-1] == n:
        print(1)
    elif lst[0] == n and lst[-1] == 1:
        print(3)
    else:
print(2)
```

سوال C: 1567B

- ابتدا شرط MEX را در نظر بگیرید: کوتاهترین آرایه با MEX a آرایه [۰،۱ ،... ، a − 1] است که طول a دارد. اکنون ما شرایط XOR را در نظر می گیریم.فرض کنید XOR آرایه [۰،۱ ،... ، a − 1] برابر با x باشد. ما سه حالت داریم:

- حالت ۱: x == b پس نیازی به افزودن هیچ عنصری به آرایه نداریم ، بنابراین پاسخ a است.

- حالت ۲: x ≠ b و x⊕b ≠ a. سپس می توانیم x ⊕b را به آرایه اضافه کنیم ، بنابراین MEX همچنان a خواهد بود. سپس XOR آرایه x⊕x⊕b = b خواهد بود. پاسخ a+۱است.

- حالت ۳: x ≠ b و x⊕b = a. پس نمی توانیم عنصر x⊕b را به انتهای آرایه اضافه کنیم. ما فقط می توانیم x⊕b⊕1 و ۱ را اضافه کنیم ، بنابراین XOR آرایه x⊕x⊕b⊕1⊕1 = b خواهد بود. پاسخ a+2 است.

- لینک های جالب برای حساب کردن xor به روشی دیگر و با پیچیدگی زمانی بهتر:

https://www.geeksforgeeks.org/calculate-xor-1-n

C++:

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const ll mx  = 3*1e5+10;
ll xs[mx];
int main(){
    for(ll i=1;i<mx;i++){
        xs[i]=i^xs[i-1];
    }
    ll t;
    cin >> t;
    while(t--){
        ll a,b,ans;
        cin >> a >> b;
        if( xs[a-1]==b)ans = a;
        else if((xs[a-1]^b)!=a)ans=a+1;
        else ans = a+2;
        cout<<ans<<endl;

    }
    return 0;
}
```

Python:

```
or _ in range(int(input())):
    a,b = map(int, input().split())
    # print("a,b",a,b)
    p = 0

    if a%4==1:
        p=a-1
    elif a%4==2:
        p=1
    elif a%4==3:
        p=a

    if p==b:
        print(a)
    else:
        if p^b != a:
            print(a+1)
        else:
            print(a+2)
```

سوال D: 1574B

- فرض می کنیم آن آرایه مرتب شده است. اول از همه ، اگر $a[n] \geq a[n-1]+a[n-2]$ باشد ، پاسخ NO است (زیرا در غیر این صورت an نمیتواند کوچکتر از مجموع همسایگان باشد). ادعا می کنیم که در همه موارد دیگر پاسخ بله است. یکی از ساختارهای احتمالی (اگر آرایه قبلاً مرتب شده باشد) این است:

  a[n-2] , a[n],a[n-1] , a[n-3] , a[n-4],,,1

- به راحتی می توان دید که همه اعداد به جز یک حداقل یک همسایه دارند که از خودش کوچکتر نیست.

C++:

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

int main(){
    ll n;
    cin >> n;
    std::vector<ll> num(n);
    for(ll i=0;i<n;i++)cin >> num[i];
    sort(num.begin(),num.end());
    if(num[n-1]>=num[n-2]+num[n-3]){
        cout<<"NO"<<endl;
    }
    else{
        cout<<"YES"<<endl;
        cout<<num[n-3]<<' '<<num[n-1]<<' '<<num[n-2]<<' ';
        for(ll i=n-4;i>=0;i--){
            cout<<num[i]<<' ';
        }
        cout<<endl;
    }
    return 0;
}
```

Python:

```python
input()
a=sorted(map(int,input().split()))[::-1]
a[:2]=a[1::-1]
if a[1]<a[0]+a[2]:print('YES',*a)
else:print('NO')
```

سوال E: 1552B

- در این سوال باید به این نکته توجه کرد که نمیتوان تمام n به توان ۲ حالت را بررسی کرد زیرا زمان سوال سه ثانیه است و راه حل n به توان دو ۲۵ ثانیه طول میکشد!(چرا؟)
- برای اینکه این سوال را در زمان کمتری حل کرد باید سعی کنیم با یکبار iterate کردن superior را پیدا کنیم. نکته قابل توجه این است که حداکثر یک superior وجود خواهد داشت. همچنین اگر superior برای b بازیکن a باشد، قطعا برعکس آن ممکن نیست. پس میتوانیم از همان ابتدای بازی، یک بازیکن را superior در نظر بگیریم و مثلا sup=i قرار دهیم. با بازیکن بعدی چک کنیم تا ببینیم آیا superior آن هست یا نه. اگر نبود، sup=i+1 قرار میدهیم. و با یکبار iterate کردن روی همه بازیکن ها، superior را به دست میاوریم.

C++:

```cpp
#include <bits/stdc++.h>
using namespace std;
const int M = 5;
void solve(){
    int n, x;
    cin >> n;
    vector<vector<int>> ath(n);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < M; j++)
        {
            cin >> x;
            ath[i].push_back(x);
        }
    int gold = 0, cnt = 0;
    for (int i = 1; i < n; i++){
        for (int j = 0; j < M; j++)
            cnt += ath[i][j] < ath[gold][j];
        if (cnt >= 3)
            gold = i;
        cnt = 0;
    }
    cnt = 0;
    for (int i = 0; i < n; i++){
        if (i == gold)
            continue;
        for (int j = 0; j < M; j++)
            cnt += ath[i][j] < ath[gold][j];
        if (cnt >= 3){
            cout << -1 << "\n";
            return;
        }
        cnt = 0;
    }
    cout << gold + 1 << "\n";
}
int main(){
    int t;
    cin >> t;

    for (int i = 0; i < t; i++)
        solve();
}
```