ID : team044

PASSWORD : 2jg6hydu

쉬운 dfs와 bfs 코드

```
//https://www.acmicpc.net/problem/1260
#include <iostream>
#include <vector>
#include <queue>
#include <algorithm>
#include <cstdio>
#include <cstring>
using namespace std;

bool visit[1001];
vector<int> a[1001];
void dfs(int s) {
    visit[s] = true;
    printf("%d ", s);
    for (int i = 0; i < a[s].size(); i++) {
        int next = a[s][i];
        if (visit[next] == false) {
            dfs(next);
        }
    }
}
void bfs(int s) {
    queue<int> q;
    visit[s] = true;
    q.push(s);
    while (!q.empty()) {
        int node = q.front();
        q.pop();
        printf("%d ", node);
        for (int i = 0; i < a[node].size(); i++) {
            int next = a[node][i];
            if (visit[next] == false) {
                visit[next] = true;
                q.push(next);
            }
        }
    }
}
int main() {
    int n, m, start;
    scanf("%d %d %d", &n, &m, &start);
    for (int i = 0; i < m; i++) {
        int u, v;
        scanf("%d %d", &u, &v);
        a[u].push_back(v);
        a[v].push_back(u);
    }
    for (int i = 1; i <= n; i++) {
        sort(a[i].begin(), a[i].end());
    }
    dfs(start);
    printf("\n");
    memset(visit, false, sizeof(visit));
    bfs(start);
    printf("\n");
    return 0;
}
```

백준 토마토 문제

```
#include <iostream>
#include <queue>
#include <algorithm>
using namespace std;

int arr[1001][1001];
int day[1001][1001];
int dx[4] = { 1, -1, 0, 0 };
int dy[4] = { 0, 0, 1, -1 };

int main() {
    queue<pair<int, int>> q;
    int m, n;
    scanf("%d %d", &m, &n);
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            scanf("%d", &arr[i][j]);
            day[i][j] = -1;
            if (arr[i][j] == 1) {
                q.push(make_pair(i, j));
                day[i][j] = 0;
            }
        }
    }
    while (!q.empty()) {
        int a = q.front().first;
        int b = q.front().second;
        q.pop();
        for (int i = 0; i < 4; i++) {
            int nx = a + dx[i];
            int ny = b + dy[i];
            if (nx >= 0 && nx < n&&ny >= 0 && ny < m) {
                if (arr[nx][ny] == 0 && day[nx][ny] == -1) {
                    day[nx][ny] = day[a][b] + 1;
                    q.push(make_pair(nx, ny));
                }
            }
        }
    }
    int ans = 0;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (day[i][j] > ans)
                ans = day[i][j];
        }
    }
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (arr[i][j] == 0 && day[i][j] == -1) //토마토 있음(안익은상
태), 방문 안함 -> 평생 안익음
                ans = -1;
        }
    }
    printf("%d\n", ans);
    return 0;
}
```

백준 가장 긴 증가하는 부분 수열

```
#include <iostream>
using namespace std;
int main(){
        int arr[1001] = {0};
        int dp[1001] = {0};
        int cnt=1;
        int n;
        cin >> n;
        for(int i=1; i<=n; i++){
                cin >> arr[i];
        }

        int ans=1;
        dp[1] = 1;
        for(int i=1; i<=n; i++){
                int max_num = 0;
                for(int j=1; j<i; j++){
                        if(arr[i] > arr[j]){
                                max_num = max(max_num,
dp[j]);
                        }
                        dp[i] = max_num+1;
                        ans = max(ans, dp[i]);
                }
        }
        cout << ans;
        return 0;
}
```

백준 LCS

```
#include <iostream>
#include <algorithm>
#include <string>
using namespace std;
int dp[1005][1005];
int main(){
        string a, b;
        int len = 0;
        cin >> a;
        cin >> b;
        for(int i=1; i<=a.size(); i++){
                for(int j=1; j<=b.size(); j++){
                        if(a[i-1]==b[j-1]){
                                dp[i][j] = dp[i-1][j-1]+1;
```

```
                                                }
                                                else{
                                                                dp[i][j]    =    max(dp[i][j-1],
dp[i-1][j]);
                                                }
                                                len = max(len, dp[i][j]);
                                }
                }

                cout << len;
                return 0;
}

//욕심쟁이판다
#include <iostream>
#include <algorithm>
using namespace std;

int arr[501][501];
int dp[501][501];
int n;

int solve(int x, int y) {
    int dx[4] = {1, -1, 0, 0};
    int dy[4] = {0, 0, 1, -1};
    if (dp[x][y]) {
        return dp[x][y];
    }
    dp[x][y] = 1;

    for (int i = 0; i < 4; i++) {
        int nextx = x + dx[i];
        int nexty = y + dy[i];
        if (nextx >= 0 && nexty >= 0 && nexty < n && nextx < n) {
            if (arr[x][y] < arr[nextx][nexty]) {
                dp[x][y] = max(dp[x][y], solve(nextx, nexty)+1);
            }
        }
    }
    return dp[x][y];
}
int main() {
    scanf("%d", &n);

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &arr[i][j]);
        }
    }
    int ans = 0;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            ans = max(ans, solve(i, j));
        }
    }
    printf("%d\n", ans);
    return 0;
}
```

세그먼트 트리

```
//BOJ 2042 구간 합 구하기
// https://www.acmicpc.net/problem/2042

#include <iostream>
using namespace std;
long long arr[1000001];
long long tree[3000002];
long long make(int left, int right, int node){
        if(left == right){
                return tree[node] = arr[left];
        }
        int mid = (left+right)/2;

        tree[node] += make(left, mid, node*2);
        tree[node] += make(mid+1, right, node*2+1);
        return tree[node];
}
long long sum(int node, int left, int right, long long start, long long end){
        if(right<start || end < left)     return 0; // 구간 밖
        if(start <= left && right <= end)         return tree[node];
        int mid = (left+right)/2;
        return sum(node*2, left, mid, start, end) + sum(node*2+1, mid+1, right, start, end);
}

void update(int left, int right, int node, long long change, long long diff){
        if(!(left<=change && change <= right))   return;
        tree[node]+=diff;
        if(left!=right){
                int mid = (left+right)/2;
                update(left, mid, node*2, change, diff);
                update(mid+1, right, node*2+1, change, diff);
        }
}
int main(){
        int n, m, k;
        cin >> n >> m >> k;
        for(int i=1; i<=n; i++){
                scanf("%lld", &arr[i]);
        }
        make(1,n,1);

        for(int i=0; i<m+k; i++){
                long long ck, a, b;
                scanf("%lld %lld %lld", &ck, &a, &b);
                if(ck == 1){
                        long long diff = b-arr[a];
                        arr[a] = b;
```

```
				update(1,n,1,a,diff);
			}
			else if(ck==2){
				printf("%lld\n", sum(1,1,n,a,b));
			}
		}
}
```

문자열 알고리즘

KMP 알고리즘 - 찾기

```
//BOJ 1786 찾기
//https://www.acmicpc.net/problem/1786

#include <iostream>
#include <string>
#include <vector>
using namespace std;
vector<int> idx;
int cnt=0;
vector<int> makeTable(string pattern){
		int j = 0;
		vector<int> table(pattern.size(), 0);
		for(int i=1; i<pattern.size(); i++){
			while(j>0 && pattern[i]!=pattern[j]){
				j = table[j-1];
			}
			if(pattern[i]==pattern[j]){
				table[i] = ++j;
			}
		}
		return table;
}
void KMP(string parent, string pattern){
		vector<int> table = makeTable(pattern);
		int j = 0;
		for(int i=0; i<parent.size(); i++){
			while(j>0 && parent[i]!=pattern[j]){
				j = table[j-1];
			}
			if(parent[i]==pattern[j]){
				if(j == pattern.size()-1){
					cnt++;

idx.push_back(i-pattern.size()+2);
					j = table[j];
				}
				else{
					j++;
				}
			}
		}
		return;
}
int main(){
		string pattern, parent;
		getline(cin, parent);
```

```
        getline(cin, pattern);
        KMP(parent, pattern);
        cout << cnt << "\n";
        for(int i=0; i<idx.size(); i++){
                printf("%d ", idx[i]);
        }
        return 0;
}
```

dp+문자열 - 백준 팰린드롬?

```
#include <iostream>
using namespace std;
int dp[2001][2001];
int arr[2001];
int main(){
        int n;
        cin >> n;
        for(int i=1; i<=n; i++){
                scanf("%d", &arr[i]);
                dp[i][i] = 1;
        }
        for(int i=1; i<n; i++){
                if(arr[i]==arr[i+1])    dp[i][i+1] = 1;
        }
        for(int k=3; k<=n; k++){
                for(int i=0; i<=n-k+1; i++){
                        int j=k+i-1;
                        if(arr[i]==arr[j] && dp[i+1][j-1]==1){
                                dp[i][j] = 1;
                        }
                }
        }
        int m;
        cin >> m;
        while(m--){
                int s, e;
                scanf("%d %d", &s, &e);
                if(arr[s]!=arr[e])      printf("0\n");
                else            printf("%d\n", dp[s][e]);
        }
}
```

최소 공통 조상 찾기 - LCA

```
//정점들의 거리
#include <iostream>
#include <queue>
#include <vector>
#include <utility>
using namespace std;
#define MAX 40002
vector<pair<int, int> > tree[MAX];
int par[MAX]; int depth[MAX]; int d[MAX];
bool check[MAX];
int lca(int a, int b){
        int ans=0;
        if(depth[a]<depth[b]){
                swap(a,b);
        }
        while(depth[a]!=depth[b]){
                ans+=d[a];
                a= par[a];
        }
        while(a!=b){
                ans+=d[a];
                ans+=d[b];
                a=par[a];
                b=par[b];
        }
        return ans;
}
int main(){
        int n, t;
        scanf("%d", &n);
        for(int i=0; i<n-1; i++){
                int u, v, c;
                scanf("%d %d %d", &u, &v, &c);
                tree[u].push_back(make_pair(v,c));
                tree[v].push_back(make_pair(u,c));
        }
        queue<int> q;
        check[1] = true;
        q.push(1);
        while(!q.empty()){
                int x = q.front();
                q.pop();
                for(int i=0; i<tree[x].size(); i++){
                        int y =tree[x][i].first;
                        if(!check[y]){
                                par[y] = x;
                                depth[y] = depth[x]+1;
                                d[y]=tree[x][i].second;
                                q.push(y);
                                check[y] = true;
                        }
                }
        }
        scanf("%d", &t);
        while(t--){
                int u,v;
                scanf("%d %d", &u, &v);
                printf("%d\n", lca(u,v));

        }
}
```

다익스트라 알고리즘

```
//최소 비용 구하기
#include <iostream>
#include <queue>
#include <vector>
#include <utility>
#define INF 99999999
```

```
using namespace std;
typedef pair<int, int> edge;
int main(){
	int n, m, start, end;
	cin >> n >> m;
	int d[n+1];
	bool c[n+1];
	vector<edge> graph[n+1];
	for(int i=0; i<m; i++){
		int from, to, cost;
		cin >> from >> to >> cost;
		graph[from].push_back(make_pair(to, cost));
	}
	for(int i=1; i<=n; i++){
		d[i] = INF;
		c[i] = false;
	}
	cin >> start >> end;
	d[start] = 0;
	priority_queue<pair<int,int>,    vector<pair<int,    int>    >,
greater<pair<int, int> > > pq;
	pq.push(make_pair(0, start));
	while(!pq.empty()){
		int x = pq.top().second;
		pq.pop();
		if(!c[x]){
			c[x] = true;
			for(int j=0; j<graph[x].size(); j++){
						int y = graph[x][j].first;

if(d[y]>d[x]+graph[x][j].second){
									d[y]          =
d[x]+graph[x][j].second;

pq.push(make_pair(d[y], y));
						}
					}
				}
			}
	cout << d[end] << endl;
}
```

최소 스패닝 트리

```
//MST
#include <iostream>
#include <vector>
#include <queue>
#include <algorithm>
#define MAX 10001
using namespace std;
struct Edge{
	int start, end, cost;
	bool operator < (const Edge& other) const{
		return cost < other.cost;
	}
};
int parent[MAX];
int Find(int x){
	if(parent[x]==x)	return x;
	else	return parent[x]=Find(parent[x]);
}
void Union(int x, int y){
	x = Find(x);
	y = Find(y);
	parent[y] = parent[x];
}
int main(){
	int v, e, ans=0;
	cin >> v >> e;
	for(int i=1; i<=v; i++){
		parent[i] = i;
	}
	vector<Edge> graph(e);
	for(int i=0; i<e; i++){
		cin >> graph[i].start >> graph[i].end >> 
graph[i].cost;
	}
	sort(graph.begin(), graph.end());
	for(int i=0; i<e; i++){
		if(Find(graph[i].start)!=Find(graph[i].end)){
			ans += graph[i].cost;
			Union(Find(graph[i].start), 
Find(graph[i].end));
		}
	}
	printf("%d", ans);
	return 0;
}

//SPFA 최단거리 알고리즘
//웜홀
#include <iostream>
#include <vector>
#include <string.h>
#include <utility>
#include <queue>
#define INF 99999999
using namespace std;
long long int d[501];
bool c[501]={false};
long long int cnt[501] ={0};
int main(){
	int t;
	scanf("%d", &t);
	while(t--){
		vector<pair<int, int> > graph[501];
		bool negative_cycle = false;
		memset(c, false, sizeof(c));
		int n, m, w;
		scanf("%d %d %d", &n, &m, &w);
		for(int i=0; i<m; i++){
			int s, e, t;
			scanf("%d %d %d",&s, &e, &t);
```

```
                graph[s].push_back(make_pair(e,t));
                graph[e].push_back(make_pair(s,t));
        }
        for(int i=0; i<w; i++){
                int s,e,t;
                scanf("%d %d %d",&s, &e, &t);
                graph[s].push_back(make_pair(e,-t));
        }
        for(int i=2; i<=n; i++){
                d[i] = INF;
        }
        queue<int> q;
        for(int j=1; j<=n; j++){
            memset(cnt, 0, sizeof(cnt));
                if(c[j]==true   ||   negative_cycle==true)
continue;
            d[j]=0;
            c[j]=true;
            q.push(j);
            cnt[j]++;
            while(!q.empty()) {
            int from = q.front();
                q.pop();
                c[from] = false;
                if(negative_cycle)      continue;

                for(auto &e : graph[from]){
                        int to = e.first;
```

```
                                int cost = e.second;
                                if(d[to]>d[from]+cost){
                                        d[to]=d[from]+cost;
                                        if(c[to]==false){

q.push(to);
                                                        c[to] =
true;
                                                cnt[to]++;

if(cnt[to]>=n){

negative_cycle = true;
                                                }
                                        }
                                }
                        }
                }

        }
        if(!negative_cycle)         printf("NO\n");
                else        printf("YES\n");
    }
}
```

이분탐색

//공유기 설치

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
int n,c;
vector<int> v;
bool check(long long x){
	int temp = 1;
	int t=v[0];
	for(int i=0; i<n; i++){
		if(v[i]-t >= x){
			t = v[i];
			temp++;
		}
	}
	if(temp >= c)		return true;
	else		return false;
}
int main(){
	int m;
	scanf("%d %d", &n, &c);
	for(int i=0; i<n; i++){
		scanf("%d", &m);
		v.push_back(m);
	}
	sort(v.begin(), v.end());
	long long mid, l=1, r, ans=0;
	r = v[n-1]-v[0];
	while(l<=r){
		mid = (l+r)/2;
		if(check(mid)){
			l = mid+1;
			if(ans < mid){
				ans = mid;
			}
		}
		else{
			r = mid - 1;
		}
	}
	printf("%lld", ans);
	return 0;
}
```

분할정복

```
//Z
#include <iostream>
#include <math.h>
using namespace std;
 int n, r, c;
int result;
 void recursion(int x, int y, int len) {
   if (y == r && x == c) {          //찾는 좌표의 결과값 출력
      cout << result << endl;
      return;
   }
```

```
    if (len == 1) {                  // +1
       result++; return;
    }
    if (!(y <= r && r<y + len && x <= c && c<x + len)) { //
       result += len * len;
       return;
    }
    recursion(x, y, len / 2);                        //2사분면
    recursion(x + len / 2, y, len / 2);              //1사분면
    recursion(x, y + len / 2, len / 2);              //3사분면
    recursion(x + len / 2, y + len / 2, len / 2);    //4사분면
} int main() {
    cin >> n;             // 2의 n제곱 크기
    cin >> r;             // x좌표
    cin >> c;             // y좌표
    recursion(0, 0, pow(2, n));       // pow  =  2의 n제곱
     return 0;
}
```

위상정렬

```
//문제집
#include <iostream>
#include <queue>
#include <vector>
#define MAX 32001
using namespace std;
vector<int> problem[MAX];
int ind[MAX]={0};
void topology(int n){
          priority_queue<int, vector<int>, greater<int> > pq;
          for(int i=1; i<=n; i++){
                    if(ind[i]==0){
                              pq.push(i);
                    }
          }
          for(int i=1; i<=n; i++){
                    int x = pq.top();
                    pq.pop();
                    printf("%d ", x);
                    for(int j=0; j<problem[x].size(); j++){
                              int y = problem[x][j];
                              ind[y]--;
                              if(ind[y]==0){
                                        pq.push(y);
                              }
                    }
          }
}
int main(){
          int n, m;
          cin >> n >> m;
          for(int i=0; i<m; i++){
                    int a, b;
                    scanf("%d %d", &a, &b);
                    problem[a].push_back(b);
                    ind[b]++;
```

```
	}
	topology(n);
	return 0;
}
```
이분매칭

```
//노트북 주인을 찾아서
#include <iostream>
#include <vector>
using namespace std;
vector<int> v[101];
bool c[5001];
int d[5001];
bool labtop(int s){
	for(int i=0; i<v[s].size(); i++){
		int t = v[s][i];
		if(c[t])	continue;
		c[t] = true;

		if(d[t]==0 || labtop(d[t])){
			d[t] = s;
			return true;
		}
	}
	return false;
}
int main(){
	int n, m, cnt=0;
	cin >> n >> m;
	for(int i=0; i<m; i++){
		int a, b;
		scanf("%d %d", &a, &b);
		v[a].push_back(b);
	}
	for(int i=1; i<=n; i++){
		if(labtop(i))		cnt++;
		fill(c, c+5001, false);
	}
	cout << cnt;
}
```

multiset

```
//이중 우선순위 큐
#include <iostream>
#include <algorithm>
#include <set>
using namespace std;
int main() {
	int t;
	cin >> t;
	while (t--) {
		multiset<int> ms;
		int q, n;
		char c;
		cin >> q;
		while (q--) {
			cin >> c >> n;
```

```
            if (c == 'I') {
                ms.insert(n);
            }
            else if (c == 'D') {
                if (ms.empty())
continue;
                 if (n == 1) {

ms.erase(--ms.end());
                 }
                 else if (n == -1) {

ms.erase(ms.begin());
                 }
            }
        }
        if (ms.empty()) {
            printf("EMPTY\n");
        }
        else {
            cout << *(--ms.end()) << " " <<
*(ms.begin()) << "\n";
        }
    }
    return 0;
}
```

그래프 문제

```
//적록색약
#include <iostream>
using namespace std;

char color[101][101];
bool visit[101][101]={false};
int n;
int dx[4] = {0,0,1,-1};
int dy[4] = {1,-1,0,0};

void dfs(int p,int q, char c){
    visit[p][q] = true;
    for(int i=0; i<4; i++){
        int nextx = p+dx[i];
        int nexty = q+dy[i];
        if(nextx >= 0 && nextx < n && nexty >=0 &&
nexty <n){
            if(!visit[nextx][nexty] && c ==
color[nextx][nexty]){
                dfs(nextx, nexty,
color[nextx][nexty]);
            }
        }
    }
}
int all(){
    int cnt = 0;
    for(int i=0; i<n; i++){
```

```
		for(int j=0; j<n; j++){
			if(!visit[i][j]){
				dfs(i,j,color[i][j]);
				cnt++;
			}
		}
	}
	return cnt;
}
int main(){
	int count1, count2;
	cin >> n;
	for(int i=0; i<n; i++){
		scanf("%s", &color[i]);
	}
	count1 = all();
	for(int i=0; i<n; i++){
		for(int j=0; j<n; j++){
			visit[i][j] = false;
			if(color[i][j] == 'G')  color[i][j] = 'R';
		}
	}
	count2 = all();

	cout << count1 << " " << count2;
}
```

이진 검색 트리

```
#include <stdio.h>
#include <stdlib.h>
typedef struct tree_node *tree_ptr;
struct tree_node{
	int data;
	tree_ptr left;
	tree_ptr right;
};

void insert_BST(tree_ptr tree,int item);
void postorder(tree_ptr ptr);
int main(){
	tree_ptr tree = (tree_ptr)malloc(sizeof(struct tree_node));
	int n;
	scanf("%d\n", &n);
	tree->data = n;
	tree->left = NULL;
	tree->right = NULL;
	while(scanf("%d\n", &n)!=EOF){
		insert_BST(tree, n);
	 }
	postorder(tree);
}
void insert_BST(tree_ptr tree,int item){
	tree_ptr node = (tree_ptr)malloc(sizeof(struct tree_node));
	tree_ptr temp, prev;
	node->data = item;
	node->left = NULL;
```

```
        node->right = NULL;

        if(tree == NULL)    tree = node;
        else{
                temp = tree;
                prev = NULL;
                while(1){
                        prev = temp;
                        if(item < prev->data){
                                temp = temp->left;
                                if(temp == NULL){
                                        prev->left      =
node;
                                        return;
                                }
                        }
                        else if(item > prev->data){
                                temp = temp->right;
                                if(temp==NULL){
                                        prev->right     =
node;
                                        return;
                                }
                        }
                        else    return;
                }
        }
}
```

```
void postorder(tree_ptr ptr){
        if(ptr){
                postorder(ptr->left);
                postorder(ptr->right);
                printf("%d\n", ptr->data);
        }
        else    return;
}
```

슬라이딩 윈도우

```
//최솟값 찾기

//BOJ 11003 최솟값 찾기
//https://www.acmicpc.net/problem/11003

#include <iostream>
#include <vector>
#include <deque>
using namespace std;
int main(){
        int n, l;
        cin >> n >> l;
        deque<pair<int,int> > d;
        vector<int> ans(n);
        int a[n+1];
        for(int i=0; i<n; i++){
                scanf("%d", &a[i]);
        }
        for(int i=0; i<n; i++){
```

```
        int cur = a[i];
        if(!d.empty() && d.front().second<=i-l){
            d.pop_front();
        }
        while(!d.empty()&&d.back().first> cur){
            d.pop_back();
        }
        d.push_back(make_pair(cur, i));
        ans[i] = d.front().first;
        printf("%d ", ans[i]);
    }
    return 0;

}
```

dp+그래프

```
//내리막 길
int dfs(int x, int y)
{
    // 목적지에 도착하면 최초 경우의 수 1 반환
    if(x == m && y == n) return 1;
    // 방문한 적 없다면
    if(dp[x][y] == -1) {
        dp[x][y] = 0; // 방문했다!
        for(int i = 0; i < 4; i++) {
            int nx = x + dx[i];
            int ny = y + dy[i];
            // 인덱스가 범위 안에 있는지 체크
            if(nx >= 1 && nx <= m && ny >= 1
&& ny <= n) {
            if(height[x][y] > height[nx][ny]) {
            // 도착지점에서부터 출발지점까지 역순
으로 경우의 수를 추가하면서 채워 나감
                dp[x][y] += dfs(nx, ny);
                }
            }
        }
    }
    return dp[x][y];
}
```

재귀 + dp

// 카드게임

```
//BOJ 11062 카드게임
// https://www.acmicpc.net/problem/11062
#include <iostream>
#include <algorithm>
using namespace std;
int card[1001];
int dp[1001][1001][2];
int game(int s, int e, int flag){
    int &r = dp[s][e][flag];
    if(r!=-1){
        return r;
    }
    if(s>=e){
        if(flag)    return r = card[s];
        else        return r=0;
    }
    if(flag){
        r=              max(game(s+1,               e,
!flag)+card[s],game(s,e-1,!flag)+card[e]);
    }
    else{
        r=min(game(s+1,e,!flag), game(s,e-1,!flag));
    }
    return r;
}
int main(){
    int n,t;
    cin >> t;
    while(t--){
        int ans = 0;
        scanf("%d", &n);
        for(int i=0; i<n; i++){
            scanf("%d", &card[i]);
        }
        for(int i=0; i<n; i++){
            for(int j=0; j<n; j++){
                for(int k=0; k<2; k++){
                    dp[i][j][k] = -1;
                }
            }
        }
        ans += game(0,n-1,1);
        printf("%d\n", ans);
    }
    return 0;
}
```