ZJNU 2020-02-11 题解

A - Mental Rotation

简单模拟

B - SpongeBob SquarePants

签到

C - I Don't Want To Pay For The Late Jar!

简单贪心

E - Optimal Slots

背包+想法

题意：

背包容量为t，有n个物品，每个物品有一个体积，怎么放使得空闲的容量最小？如果有多种答案，则让先出现的物品优先放入背包。

数据范围： $t \leq 1000, n \leq 50$

思路：

$dp[i][j]$ 代表的是在从后往前拿到第 i 个物体后，能否有容量为 j 的组合

因为先出现的物体优先放入，所以我们背包的顺序要从后往前，这样我们在判断拿了第 i 个物体后能否有容量为 j 的组合时，只需判断 $dp[i+1][j-w[i]]$ 是否存在就行了。

这样我们先得到了能得到的最大容量 mx 后，从前往后贪心的取，能拿就拿就行了。

代码：

```
#include <bits/stdc++.h>
using namespace std;
#define rep(i,j,k) for(int i = (int)j;i <= (int)k;i ++)
#define debug(x) cerr<<#x<<":"<<x<<endl
#define pb push_back

typedef long long ll;
typedef pair<int,int> pi;
const int MAXN = (int)1e3+7;

int t,n;
int w[MAXN];
bool dp[57][MAXN];

int main()
{
    while (scanf("%d",&t),t) {
        scanf("%d",&n);
        rep(i,1,n) scanf("%d",&w[i]);
        rep(i,0,n+1) rep(j,0,t+1) dp[i][j] = 0;
        dp[n+1][0] = 1;
        int tmpt = 0;
        for(int i = n;i >= 1;i --) {
            rep(j,0,t) dp[i][j] = dp[i+1][j];
            for(int j = t;j >= 0;j --) {
                if (j + w[i] > t) continue;
                if (dp[i+1][j]) {
                    dp[i][j+w[i]] = 1;
                }
            }
        }
        rep(j,0,t) if(dp[1][j]) tmpt = j;
        int ans = tmpt;
        rep(i,1,n) {
            if (tmpt == 0) break;
            if (dp[i+1][tmpt-w[i]]) {
                tmpt -= w[i];
                printf("%d ",w[i]);
            }
        }
        printf("%d\n",ans);
    }
}
```

F - Military Class

动态规划+状压

题意：

有 2 列人，每列都是 n 个人。 我们想要这些人分成 n 组每组两人的组合。但是第一行的第 i 个人能和第二行的人第 j 个人组合的条件是$abs(i-j) \le e$ ，另外，还有 k 个对人之间有矛盾，他们不能组合。问最后能成功组合的方案数是多少?

数据范围：$n \le 2000, e \le 4, k \le 2000$

思路：

从数据范围上可以看到 e 非常小，这肯定就是突破点。

判断第 i 个人在第二行中适合的人选最多也就只有 e*2+1 = 9 个人。这个代表这我们可以直接暴力的枚举每个人之前选择的状态。这样的复杂度就是 O（$n * 2^9 * 9$），可以过了。

$dp[i][j]$ 第i个人选完后，所有能被第 i 个人选择对象的状态 j 下的方案数。

代码：

```
#include <bits/stdc++.h>
using namespace std;
#define rep(i,j,k) for(ll i = (ll)j;i <= (ll)k;i ++)
#define debug(x) cerr<<#x<<":"<<x<<endl
#define pb push_back

typedef long long ll;
typedef pair<ll,ll> pi;
const ll MAXN = (ll)2e3+7;
const ll MOD = (ll)1e9+7;

ll dp[MAXN][517],dp2[MAXN][517];
bool pic[MAXN][MAXN];

int aaaa[10];
```

```cpp
void test(int n,int e,int k) {
    rep(i,1,n) aaaa[i] = i;
    ll ans = 0;
    do {
        bool flag = 0;
        rep(i,1,n) {
            int to = aaaa[i];
            if (abs(to-i) > e) flag = 1;
            if (pic[i][to]) flag =  1;
        }
        if (flag == 0) ans ++;
    }while (next_permutation(aaaa+1,aaaa+n+1));
    debug(ans);
}

signed main()
{
    ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);

    ll n,e,k;
    cin >> n >> e >> k;
    rep(i,1,k) {
        ll u,v;
        cin >> u >> v;
        pic[u][v] = 1;
    }
    dp[0][0] = 1;
    ll ans = 0;
    rep(id,1,n) {
        ll l = max(1LL,id-e);
        ll r = min(n,id+e);
        ll st = r-l+1;

        rep(i,0,(1<<st)-1) {
            rep(j,0,st-1) {
                if ((i>>j)%2 == 0) {
                    ll nowid = r-j;
                    if (pic[id][nowid] == 1) continue;
                    dp[id][i|(1<<j)] = (dp[id-1][i] + dp[id][i|(1<<j)])%MOD;
                }
            }

            ll nxl = max(1LL,id+1-e);
            ll nxr = min(n,id+1+e);
            ll nxst = nxr-nxl+1;

            ll toi = i;
            if (nxst == st-1) {
                if (i&(1<<(st-1))) toi -= (1<<(st-1));
            }else if (nxst == st) {
                if (nxr != r) {
                    if (i&(1<<(st-1))) toi -= (1<<(st-1));
                    toi <<= 1;
                }
            }else {
                toi <<= 1;
            }
            dp2[id][toi] = (dp[id][i] + dp2[id][toi])%MOD;
        }

        if (id == n) {
            ans = dp[id][(1<<st)-1];
        }
        rep(i,0,516) dp[id][i] = dp2[id][i];
    }
    cout << ans << endl;

    //test(n,e,k);
}
```

H - Are You Safe?

计算几何模板题

题意:

给n个点，给出这n个点的凸包，在给出m个点，判断这m个点是否在之前计算的凸包中

思路:

直接套用求凸包的模板，和多边形的模板即可。

代码:

```
#include <bits/stdc++.h>
using namespace std;
#define rep(i,j,k) for(int i = (int)j;i <= (int)k;i ++)
#define debug(x) cerr<<#x<<":"<<x<<endl
#define pb push_back
```

```cpp
typedef long long ll;
const int MAXN = (int)1e6+7;

const double eps = 1e-6;//eps用于控制精度
const double pi = acos(-1.0);//pi
struct Point//点或向量
{
    double x, y;
    Point() {}
    Point(double x, double y) :x(x), y(y) {}
};
typedef Point Vector;

//基本向量运算
Vector operator + (Vector a, Vector b)//向量加法
{
    return Vector(a.x + b.x, a.y + b.y);
}
Vector operator - (Vector a, Vector b)//向量减法
{
    return Vector(a.x - b.x, a.y - b.y);
}
Vector operator * (Vector a, double p)//向量数乘
{
    return Vector(a.x*p, a.y*p);
}
Vector operator / (Vector a, double p)//向量数除
{
    return Vector(a.x / p, a.y / p);
}
int dcmp(double x)//精度三态函数(>0,<0,=0)
{
    if (fabs(x) < eps)return 0;
    else if (x > 0)return 1;
    return -1;
}
bool operator == (const Point &a, const Point &b)//向量相等
{
    return dcmp(a.x - b.x) == 0 && dcmp(a.y - b.y) == 0;
}
double Dot(Vector a, Vector b)//内积
{
    return a.x*b.x + a.y*b.y;
}
double Length(Vector a)//模
{
    return sqrt(Dot(a, a));
}
```

```
double Angle(Vector a, Vector b)//夹角,弧度制
{
    return acos(Dot(a, b) / Length(a) / Length(b));
}
double Cross(Vector a, Vector b)//外积
{
    return a.x*b.y - a.y*b.x;
}
double Distance(Point a, Point b)//两点间距离
{
    return sqrt((a.x - b.x)*(a.x - b.x) + (a.y - b.y)*(a.y - b.y));
}


Point P_poly[1005];
int n_poly;
bool InsidePolygon (Point A) //判断点是否在凸多边形内（角度和判别法）
{
    double alpha = 0;
    for (int i = 0; i < n_poly; i++)
        alpha += fabs(Angle(P_poly[i] - A, P_poly[(i + 1) % n_poly] - A));
    return dcmp(alpha - 2 * pi) == 0;
}

int n, top;
Point P[1005],sto[1005];
bool cmp(Point A, Point B)
{
    double ans = Cross(A - P[0], B - P[0]);
    if (dcmp(ans) == 0)
        return dcmp(Distance(P[0], A) - Distance(P[0], B)) < 0;
    else
        return ans > 0;
}
void Graham()//Graham凸包扫描算法
{
    for (int i = 1; i < n; i++)//寻找起点
        if (P[i].y < P[0].y || (dcmp(P[i].y - P[0].y) == 0 && P[i].x < P[0].x))
            swap(P[i], P[0]);
    sort(P + 1, P + n, cmp);//极角排序，中心为起点
    P_poly[0] = P[0];
    P_poly[1] = P[1];
    top = 1;
    for (int i = 2; i < n; i++)
    {
        while (Cross(P_poly[top] - P_poly[top - 1], P[i] - P_poly[top - 1]) < 0
&& top >= 1)
            top--;
        P_poly[++top] = P[i];
    }
}

int main()
{
    int T;
    scanf("%d",&T);
    rep(ca,1,T) {
        int n2;
        scanf("%d %d",&n,&n2);
        rep(i,0,n-1) {
            scanf("%lf %lf",&P[i].x,&P[i].y);
        }
        rep(i,0,n2-1) {
            scanf("%lf %lf",&sto[i].x,&sto[i].y);
        }
        Graham();
        printf("Case %d\n",ca);
        int n3 = top*2+1,beg = 0;
        int mnx = 507,mny = 507;
        rep(i,0,top) {
            P_poly[i+top+1] = P_poly[i];
            int tx = P_poly[i].x;
            int ty = P_poly[i].y;
            if (mnx > tx) {
                beg = i;
                mnx = tx;
                mny = ty;
            } else if (mnx == tx && mny > ty) {
                beg = i;
                mnx = tx;
                mny = ty;
            }
        }
        rep(i,beg,beg+top+1) {
            printf("%.0f %.0f\n",P_poly[i].x,P_poly[i].y);
        }
        n_poly = top+1;
        rep(i,0,n2-1) {
            if (InsidePolygon(sto[i])) {
                printf("%.0f %.0f is unsafe!\n",sto[i].x,sto[i].y);
            }else {
                printf("%.0f %.0f is safe!\n",sto[i].x,sto[i].y);
            }
        }
        printf("\n");
    }
}
```

I - To Crash Or Not To Crash

签到题

J - Kitchen Plates

拓扑排序模板题

K - Help The Support Lady

简单贪心