信奥算法

income

请同学写出题目大意
已知什么求什么

给定n个正整数的序列，
请问平均数恰为A的区间有几个?

识别难点

平均数A如何处理

每个数减去平均数A

如何重述原问题?

每个数减去平均数A后
有多少区间的总和恰为0

区间和恰为0

前缀和辅助计算

```
cin>>n>>A;
for(ll i=1;i<=n;++i){
    cin>>x[i];
    x[i]-=A;
    s[i]=s[i-1]+x[i];
}
```

区间和恰为0

前缀和辅助计算
s[i]表示每个数扣除A后
前i个数的总和

如何重述原问题?

对于s[0],s[1],..,s[n]序列
两两相同的数对有几对?

方法1	map计数器
方法2	排序后“物以类聚”

```
    sort(s,s+n+1);
    ll ans=0;
    ll cnt=1;
    for(ll i=1;i<=n;++i){
        if(
            ans+=
            cnt=0;
        }
        cnt++;
    }
    ans+=cnt*(cnt-1)/2;
```

拓展问题讨论

纯随机数据的序列x[ ]一般输出是什么形态？

如何生成合适的测试数据？

随机生成原序列x[ ]很难形成合法区间

建议生成前缀和序列s[ ]
形成一组组相同数值
再推导还原出x[ ]序列

3080

equipment

请同学写出题目大意
已知什么求什么

n箱导弹可以选:第i箱价格vi,导弹数ci,攻击力fi, m个意向单可以选:第i个付费Vi,要求攻击力不少于Fi的导弹Ci个,求满足要求时最多净赚几元?

请同学阅读[数据规模和约定]
识别部分得分点

【数据规模与约定】
1号数据：n<=15; 2 号数据：m<=15;
3号数据：ci=Ci=1; 4号数据：fi=Fi=1;
5号数据：vi=Vi=1
所有数据：保证n,m<=2000，ci,Ci<=50,
fi,Fi,vi,Vi<=1000000000.

手算

输入	输出?
2 3 100 10 1 200 1 1 2 100 15	5

明显是背包模型的变种

识别要素

2大类信息做搭配：箱子，飞机

导弹个数的约束

攻击力的约束

枚举法

枚举形态是什么?
排列/组合/子集/分组?

枚举子集

2种可能的决策
当前物品取不取
下一件物品取哪个

请设计dfs()参数状态

枚举法带来的思考

识别重叠子问题

箱子和意向单的本质类似
统一处理这两类信息

按照攻击力从大到小排序
可以简化决策难度

统一物品

意向单理解为正收益+消耗导弹

箱子理解为负收益+赢得导弹

按照攻击力从大到小排序

尝试DP

请设计状态定义

f[i][j]表示只考虑前i件物品，
恰好剩j枚导弹,最多能赚多少

```
导弹箱 和 意向单 统一做物品处理，加/减金额，加/减导弹
按照f/F从大到小排序
f[i][j] 只考虑前i件物品,恰好剩j枚导弹,最多能赚多少
n=2
m=1
排序后
箱子：1 200 1
箱子：3 100 10
飞机：2 100 15
     j=0, j=1, j=2, j=3, j=4
i=0  0    -    -    -    -
i=1  0   -1    -    -    -
i=2  0   -1    -   -10  -11
i=3
```

攻击力的约束

排序后保证

若当前考虑的物品是飞机，
其所需的攻击力达标导弹都已查阅

排序规则细节：
若两个物品|金额|相同
箱子靠前放，飞机靠后放

通过DP计算的顺序化解了”攻击力约束”

```
typedef long long ll;
const ll INF=1e18;
const ll N=
const ll M=
```

如何定义f[][]

滚动数组/覆盖填写

原题变量名n,m容易混淆
改用具体信息命名

```
ll nDD,nFJ,sDD=0;
cin>>nDD;
for(ll i=1;i<=nDD;i++){
    cin>>items[i].c>>items[i].f>>items[i].v;
    sDD+=items[i].c;
    
}
cin>>nFJ;
for(ll i=nDD+1;i<=nDD+nFJ;i++){
    cin>>items[i].c>>items[i].f>>items[i].v;
    
}
ll nItems=
sort(items+1,items+1+nItems,cmp);
```

```
    f[0][0]=0;
    for(ll j=1;j<=sDD;j++)f[0][j]=-INF;
    for(ll i=1;i<=nItems;i++){
        for(ll j=0;j<=sDD;j++){
            f[i&1][j]=f[i&1^1][j];
            ll pre=j-items[i].c;
            if(pre>sDD)continue;

            if(f[i&1^1][pre]==-INF)continue;
            f[i&1][j]=max(
                        f[i&1][j],

                    );
        }
    }
```

答案表示

```
ll ans=0;
for(ll j=0;j<=sDD;j++)
    ans=max(ans,
cout<<ans<<endl;
```

太戈编程

2740

three

请同学写出题目大意
已知什么求什么

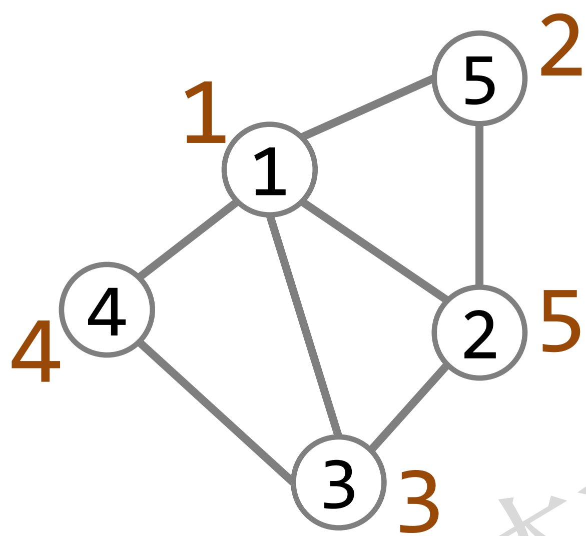
无向图里，所有三元环的三个点权最值求总和

请同学阅读[数据规模和约定]
识别部分得分点

【数据规模与约定】
1号数据：n=3
2,3号数据：3<=n<=200
所有数据：3<=n<=100000，m<=250000

手算

输入	输出?
5 7 1 5 3 4 2 1 2 2 3 5 2 4 3 3 1 1 4 5 1	14



暴力枚举

枚举三个节点u,v,w

判断是否互相为邻居

如何优化?

枚举+优化

枚举u

再枚举u的邻居v

再枚举v的邻居w

判断u和w是否为邻居

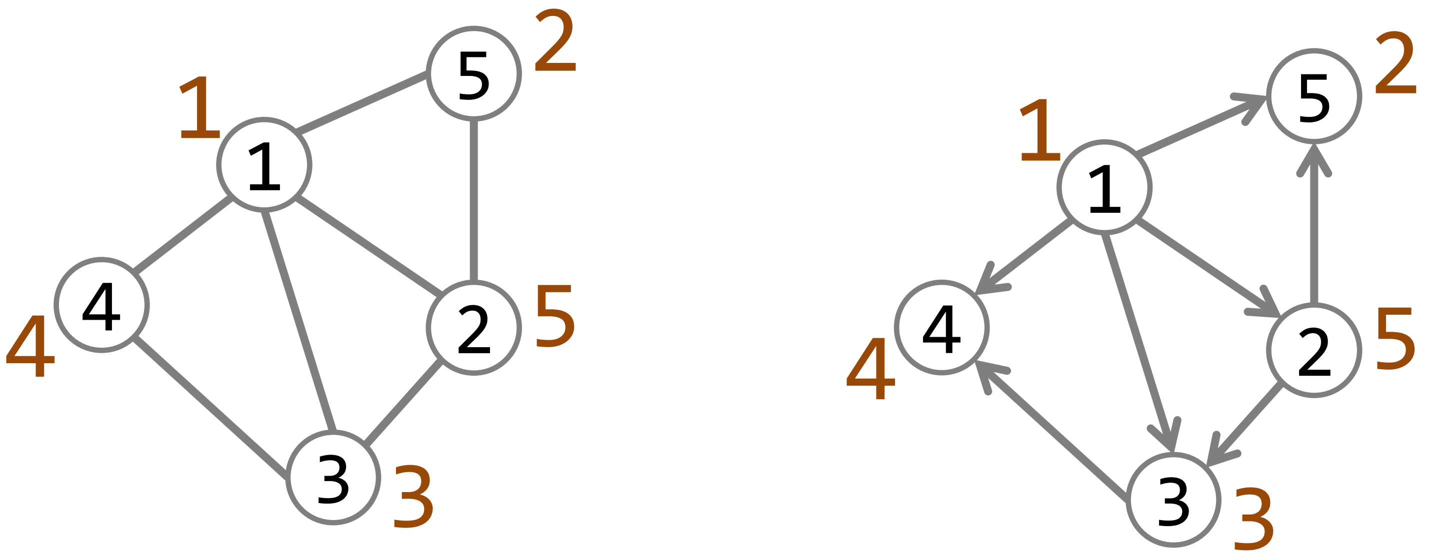
无向边变单向边

原图里对于每条无向边

原图里度数较大的点
向度数较小点连单向边

原图里度数相同时
选编号较小的点
连向编号较大的点

无向边变单向边

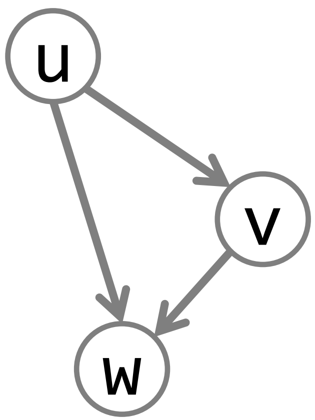


	u=1	u=2	u=3	u=4	u=5
dgr[u]=	4	3	3	2	2

```
void solve(){
	ll ans=0;
	for(int u=1;u<=n;++u){
		for(int i=hd[u];i;i=g[i].nxt)
			ok[g[i].to]=1;
		for(int i=hd[u];i;i=g[i].nxt){
			int v=g[i].to;
			int mZuv=max(z[u],z[v]);
			for(int j=hd[v];j;j=g[j].nxt){
				int w=g[j].to;
				if(!ok[w])continue;
				
			}
		}
		
		
	}
	printf("%lld\n",ans);
}
```

复杂度分析

枚举u,v,w时,计算量来自两方面



枚举u的所有邻居v	总共是O(n + m)
枚举v的所有邻居w	不超过O($m\sqrt{m}$)

v可能被多个不同的u枚举到，(v,w)边可能多次枚举

对于出度小于$\sqrt{m}$的这类v，指向v的(u,v)边最多m，
这类枚举量不超过O($m\sqrt{m}$)

对于出度大于等于$\sqrt{m}$的这类v，对应u的出度只可能更大
这样的u不会超过$\sqrt{m}$个,这类枚举量不超过O($m\sqrt{m}$)

太戈编程

2741

interstellar

请同学写出题目大意
已知什么求什么

动态最近曼哈顿距离三维点

请同学阅读[数据规模和约定]
识别部分得分点

1号数据：n=1，m=1
2号、3号、4号数据：h=1
5号、6号数据：2<=q<=2000
所有数据：n*m*h<=200000，q<=200000

纯暴力

对每个问询查看所有已存在的点

```
struct Node{
    int x,y,z;
    int cost;
};
int dst(Node&a,Node&b){
    return abs(a.x-b.x)+abs(a.y-b.y)+abs(a.z-b.z);
}
```

```
vector<Node> s;
void solveBF(){
	while(q--){
		int type,x,y,z;
		scanf("%d %d %d %d",&type,&x,&y,&z);
		Node u=(Node){x,y,z,0};
		if(type==1){
			s.push_back(u);
		}else{
			int ans=INF;
			for(int i=0;i<s.size();++i)
				ans=min(ans,dst(u,s[i]));
			printf("%d\n",ans);
		}
	}
}
```

纯暴力

对每个问询查看所有已存在的点

无法提前跳出循环

能否改进暴力
使得跳出循环成为可能
不需要查看所有已存在的点

从问询点bfs扩散找最近目标点

```
void solve(){
    for(int i=1;i<=q;++i){
        int type,x,y,z;
        scanf("%d %d %d %d",&type,&x,&y,&z);
        Node u=(Node){x,y,z,0};
        if(type==1){
            exist[id(x,y,z)]=1;
        }else{
            printf("%d\n",bfs(u,i));
        }
    }
}
```

三维点重新改为一维编号

```
const int N=200009;
int vst[N];
int d[N];
bool exist[N];
#define id(x,y,z) ((x)-1)*m*h+
```

vst数组为什么用int类型?

正解思路的形成

发现：n*m*h<=100000

三维格点总量不多

简化问题

更新前置+问询后置

存在的多个点作为起点
多源跑BFS
对所有格点求最短路

查询O(1)

定期重构算法

设定重构周期T

积累到T个新增点时
再批量加入大部队

跑T个源点的BFS
更新三维格点对应的最短路标记

定期重构算法

```
while(q--){
    int type,x,y,z;
    scanf("%d %d %d %d",&type,&x,&y,&z);
    Node u=(Node){x,y,z,0};
    if(type==1){
        
        if(s.size()>T)
            bfs();
    }else{
        int ans=d[id(x,y,z)];
        for(int i=0;i<s.size();++i)
            ans=min(ans,dst(u,s[i]));
        printf("%d\n",ans);
    }
}
```

```
void bfs(){
    queue<Node> q;
    for(int i=0;i<s.size();++i){
        q.push(s[i]);
        
    }
    s.clear();
    Node u,v;
    while(!q.empty()){
        u=q.front(), q.pop();
        for(int k=0;k<6;++k){
            v.x=u.x+dx[k];
            v.y=u.y+dy[k];
            v.z=u.z+dz[k];
            
            if(v.x<1||v.y<1||v.z<1) continue;
            if(v.x>n||v.y>m||v.z>h) continue;
            if(d[id(v.x,v.y,v.z)]<=v.cost) continue;
            d[id(v.x,v.y,v.z)]=v.cost;
            q.push(v);
        }
    }
}
```

讨论

距离数组d何时初始化

数组d只初始化1次
之后被新的多源bfs
不断覆盖更新

思路1

精算重构周期T

猜测T和q的大致关系是什么?

T=sqrt(q)?

这个式子的理由是什么?

思路2

精算重构周期T

重构次数q/T

每次重构：最差情况计算量n*m*h*6

不重构时的新增操作：计算量1

不重构时的问询操作：平均计算量T/2

假设新增q/2次，问询q/2次

总计数量估计
n*m*h*6*q/T+q/2*T/2

最优T^* = $(n*m*h*24)^{1/2}$

思路3

精算重构周期T

重构次数q/T

每次重构时空间点均匀分布：计算量 n*m*h*6/(q/2)

不重构时的新增操作：计算量1

不重构时的问询操作：平均计算量T/2

假设新增q/2次，问询q/2次

总计数量估计 n*m*h*6*2/T+q/2*T/2

最优 $T^* = (n*m*h*12/q)^{1/2}$

精算重构周期T

```
T=sqrt(q);
```

```
T=sqrt(n*m*h);
```

```
T=sqrt(n*m*h*24);
```

```
T=sqrt(n*m*h*12/q)+1;
```

精算重构周期T

随机数据 平均情况	每次重构时空间点均匀分布：计算量 n*m*h*6/(q/2)
最大值同阶	所有数据：n*m*h<=100000，q<=200000

T=sqrt(n*m*h*12/q)+1;

对于不是随机数据的情况非常危险

精算重构周期T

最大值同阶	所有数据：n*m*h<=100000，q<=200000
回归初心	T=sqrt(q)

总计数量最差情况估计
n*m*h*6*sqrt(q)+q*sqrt(q)/2

对于各类数据
整体速度都能接受

3091