知识点

基础知识

排序

1. 插入排序（C1-BC）
2. 计数排序（C2-F）
3. 快速排序（E1-A）
4. 归并排序/分治（C1-H）

基础数据结构

1. 堆（C2-C，E5-D）
2. 链表（E1-I）
3. 栈：（C7-C）

- 单调栈（E2-D）

其他

递归（C1-DJ、E1-GJ）

秦九韶算法/honer规则（C1-E）

双指针（C1-F、E1-CH）

模拟（C2-BDE，E5-F）

快速幂（C2-I）

随机化（C2-K）

前缀和（C2-K、E1-B）

概率（E1-E）

动态规划

1. 简单DP及输出方案（C2G，C3ABDEHI，E2-BCEH）

   背包问题（C4-C，E3-BC，）

   - 01背包，滚动数组优化，倒序
   - 完全背包，滚动数组优化，正序
   - 分组背包：C6-G
   - 比较难的是E3-B，背包与贪心等几个算法结合：

     `mjh` 新开了个淘宝店，了解到淘宝最近推出了 `满300减50` 的活动，觉得这个力度太小，决定自己的店铺不参与这个活动，而是制定了属于自己的优惠活动，即：

     - 可选择任意 k 件商品免费，其他商品按原价购买

     `zjx` 刚刚领到了 $W(1 \leq W \leq 10^4)$ 元的低保，打算在 `mjh` 的店铺消费一波，店铺里一共有 n 种商品（每种商品限购一件），第 i 种商品的价格为 $w_i(1 \leq w_i \leq W)$ 元，价值为 $v_i(1 \leq v_i \leq 10^9)$，他想知道在仅花费不超过 W 元的基础上，指定至多 k 件商品免费，可以获得最大价值。

     输入

     第一个数为数据组数 $T(1 \leq T \leq 6)$

     每组数据第一行为三个正整数 $n(1 \leq n \leq 5000), W(1 \leq W \leq 10^4), k(1 \leq k \leq 5000)$，含义如上所述；

     接下来 n 行，每行两个正整数 $w_i(1 \leq w_i \leq W), v_i(1 \leq v_i \leq 10^9)$，含义如上所述。

     特别地，我们保证在同一个测试点中，$\Sigma n \leq 5000$。

   流水线调度问题

   ...

2. 区间DP（C3G，C3J*）

```
for(int l=2;l<=n;l++) {
    for(int i=1;i+l-1<=n;i++) {
        int j=i+l-1;
        for(int k=i;k<j;k++) {
            f[i][j]=min/max(f[i][j],f[i][k]+f[k+1][j]+w(i,j,k));
            ...
        }
    }
}
```

3. LCS：C3F

   $O(n^2)$

4. LIS：E3-H，需要一些转化，$O(n^2)$ 和 $O(n \log n)$ 的做法都要掌握
5. 一些其他DP：E2-GJK，E4-J*

贪心

E2-A、C4-ABD、E3-A

感觉所有题目或许都可以叫做贪心，比较经典的贪心算法有Huffman树、Kruskal求最小生成树等。

但是使用贪心时需要思考正确性或者需要比较强的直觉，否则很有可能是把DP错误的写成了贪心

贪心的证明通常会有些复杂，在写题目的过程中为了节约时间可尝试去思考反例，若短时间内思考不出错解都可以尝试。

图论

0. 存图等可看之前群里发过的pdf
1. 最短路
   - Floyd：C4-E，E4-D，E5-B，E3-J*
   - dijkstra：C4-I，E5-H（主要考察如何建图）
   - BFS求无权图最短路：E2-F
2. 负环：C4-G，E4-D（和二分结合）
3. 拓扑排序：C4-H，E3-E
4. 最小生成树：E3-G，C6-A
5. 网络流：
   - 最大流：E4-A，E4-H*

     最大流重点在建图，注意当前弧优化
   - 二分图最大匹配：E4-B（也可以直接用最大流）
6. 其他：E4-E*

计算几何

corner case特别多，考虑一定要全面，任何一种没想到的情况都有可能是测试点

有板子会用就行，不要自己写

FFT

蝴蝶变换：C6B

DFT：E5A

IDFT：C6C

比较经典的FFT应用：大整数乘法C6-D，都是将问题抽象为一个多项式的形式，稍微复杂一点的还有E6-H

给定一个式子（或者推导一个式子*）形如一个卷积的形式（E5E）

字符串匹配：E6-JK

字符串

1. 哈希

   $O(1)$ 实现比较字符串相等

   结合二分答案可用于比较两个字符串公共前缀长度

   单哈希使用时偶尔会被卡，还是建议使用双哈希

2. KMP

   $O(n)$ 构建 $kmp[i]$ 记录匹配到 i 失配后应跳到哪个位置

   用于字符串匹配，关键在于构建失配指针

数论

1. 组合数（E1-F）
2. 逆元（E2-I）
3. Exgcd（C6-E）

STL

auto

自动获取所需类型

包括基本数据类型，自定义结构体，迭代器等

```
struct Node{
    int x,y;
};
queue<Node> q;
while(!q.empty()) {
    auto u=q.front();
}
```

lower/upper_bound

返回a[]中第一个大于等于/（大于）x的下标

```
int pos1=lower_bound(a+1,a+n+1,x)-a;
int pos2=upper_bound(a+1,a+n+1,x)-a;
```

stack

栈，但不推荐使用STL，建议手写

```
stack<int> s;
s.push(x);
s.top();
s.pop();
s.empty();
s.size();
```

queue

（单向）队列，可代替手写循环队列

```
queue<int> q;
q.push(x);
q.front();
q.pop();
q.empty();
q.size();
```

dequeue*

双端队列

priority_queue

优先队列（堆）

注意重载运算符，堆默认为大根堆，但是用小于号实现的，如下定义小根堆

```
priority_queue<int,vector<int>,greater<int>> q;
```

pair

可以用来代替一些便捷的自定义struct

且pair自带小于号，可直接用于排序，第一关键字为第一维升序，第二关键字为第二维升序

```
pair<int,int> p1;
pair<int,string> p2;
pair<double,int> p3;
```

vector

```
vector<int> v;
v.push_back();
v.size();
for(int i=0;i<=v.size()-1;i++) {//错误
    cout<<v[i]<<" ";
}
for(int i=0;i<=(int)v.size()-1;i++) {//(正确)
    cout<<v[i]<<" ";
}
```

set

集合，无重复，有序

```
set<int> s;
//插入
for(int i=1;i<=n;i++)
    s.insert(gint());
//遍历
for(auto x : s)
    cout<<x<<" ";
s.begin():返回第一个元素的迭代器
s.end():返回最后一个元素后的迭代器
//故也可以这样遍历
for(auto it=s.begin();it!=s.end();it++)
    cout<<(*it)<<" ";
s.lower_bound(x):返回第一个大于等于x的值的迭代器
```

map

构建一个映射关系

复杂度为 $O(\log n)$

```
map<T1,T2> mp;
map<int,int> mp1;
map<string,int> mp2;
map<int,set<int> mp3;
map<int,vector<int>> mp4;
```

做题思想/技巧

0. 输入输出优化

   卡常是不合理，但是io优化不可忽略

   一般来说的话 `scanf/printf` 是足够的，如果数据量较大（1e6）建议还是使用快读快输（即转化为字符串输入输出）

1. 随机
   - 给定 I_n^3 中的 n 个互不相同的点 $\{P_i\}$，选取一个点 A 使得 $A, P_i, P_j, (i \neq j)$ 不共线

     C_n^2 条直线每条最多经过 n 个点，占据空间内约 $\frac{1}{2}$ 的点，故随机即可找到 A
   - 多个随机数中选最多的数使其的 $\gcd > 1$
2. 二分
   - 二分查找：找第一个大于等于的数
   - 二分答案：如何写check，注意可二分性，最小的满足条件的

     ```
     int l=1,r=n;
     while(l<=r) {
         int mid=(l+r)>>1;
         if(check(mid)) r=mid-1;
         else l=mid+1;
     }
     cout<<l<<endl;
     ```
3. 差分

   令 $b_i = a_i - a_{i-1}$ ，则 $a_i = \Sigma_{j=1}^{i} b_j$ ，这样区间 $[l, r]$ 修改 d 可以转化为 $b_l += d, b_{r+1} -= d$
4. 打表（C1-I）
   - **题目描述**

     莫卡想要找出一个序列，这个序列由 n 个互不相同的非负整数组成，且这 n 个非负整数的按位异或值为 0。

     可以证明，在 $n \geq 3$ 时，至少存在一个满足以上条件的序列。但莫卡不满足于此，她想找出所有满足以上条件的序列中，序列中最大数的最小值。

注意事项

1. io优化
2. 数组大小

   如无向图双倍空间，FFT四倍空间

   空间计算 `sizeof` ，如下方法计算 a 数组所占空间（MB）：

   ```
   cout<<sizeof a/1024/1024;
   ```

3. corner case
   - $n=0,1$
   - $a_i=0,1e9,-1e9$
   - 特殊的比如几何中斜率为0
4. 初始化
5. 多测清空

   不要滥用 `memset` ，复杂度是 $O(n)$ 的，多测 $T\leq 1e5,\Sigma n\leq 1e5$ 是会被卡到 $O(n^2)$ 的，可这样用：

   ```
   memset(a,0,sizeof(int)*(n+1));//正确
   memset(a,0,sizeof a);//超时
   ```

   以及`memset` 是按字节填充，初始化最大值建议 `0x3f` ，这样相加不会爆 `int` ， `0xff` 意味着全部为1（包括符号位），故对应的是 `-1`
6. 不要卡题
7. 数据范围是否需要long long

   数据类型与输出格式是否匹配（不要使用了%d输出一个long long类型的数）
8. 使用cin,cout时是否关闭同步流（不关闭可能会导致tle）

   使用cin,cout时应用'\n' 代替 endl
9. 数组是否开的够大
10. 是否注意到了浮点数的误差（比如几何题求面积时是否可以直接用整数计算）
11. 本地编译器是否添加了编译命令
12. 是否使用了已经被禁止的函数（如gets）

...