就算刀架脖子上，也不能作弊！！！

考试策略方面

> - 3.5h，先通读三道题，每道题读三遍！不要死磕一道题，想不到正解直接写暴力，之后想到正解写暴力对拍，确保每道题有暴力上交！
> - 不熟悉的算法不要写，不要在考场肝没写过的知识点！
> - 能拍一定要拍，尤其T3，除非写定暴力了不然一定要留够时间！对拍注意边界情况(防止无良出题人)，难题可以手造小样例。
> - 要把可能有用的代码版本全部存下来！
> - 数据多猜多测，说不定就对了(面向数据的编程警告)，推完式子一定要代值/复推验证。
> - 保持良好心态，不要管旁边的人，旁边键盘噼里啪啦，下来查分他有可能是0分。不要让旁边的键盘声带乱你的节奏。考完一场直接准备下一场，不要讨论影响第二场发挥。

良好习惯

> - 文件名、文件操作一定要写对，文件名、模数最好直接从题面复制！模数定义MOD！
> - 避免“以为简单的题没对拍”、“没看数据范围瞎写TLE”、“没判特殊情况导致正解爆零”。
> - 看题之后代入样例验证理解，看清输入/输出格式，多组数据对应好输入输出

细节错误

变量之类

> - 别用什么`next`,`time`,`y1`,`y2`,`pipe`,`log`,`pow10`作变量！一个方法保证不会出锅：使用英文单词的辅音字母或拼音作为变量名。
> - 看清题目要求，记得取模，看准模数是不是质数。记得可能减到负数的时候要`(x % mod + mod) % mod`。
> - 定义了long long，格式化输出时一定要用`%lld`，如：`printf("%lld",a)`;输出数值少，用`cout<<a;`

STL

> - 定义 $1e6$ 个`STL`会占很大空间。
> - 使用`map`存 $1e6$ 个数左右就会巨慢，用`hash`要注意。
> - 别认为`vector`和`queue`很快。
> - 然而`priority_queue`在不想手写或者手写可能写挂的情况下才用。
> - `bitset`在时限较紧的时候尽量手写。
> - **log(a)**函数不要频繁调用，尽量**预处理。** `for(int i=a;i<n;++i) la[i]=la[i/a]+1;`

空间问题

> - 如果出现本来不该修改的数组被修改了，可能不是写挂了而是数组开小了。
> - `SPFA`如果手写队列一定要开循环队列
> - `>>`或`<<`超过`31`(`long long`是`63`)位是未定义操作！
> - ST表空间要开够，欧拉序列要开双倍空间，线段树开4倍空间
> - RE的一些原因：爆栈、模（除）0，数组越界

读字符串

```
string S;
char C[N];

cin >> S;
cin >> C;
scanf("%s", C);
getline(cin, S);
```

- `cin`很慢！如果要读空格必须用`getline`。

其他易错点

- 输入输出尽量少用cin、cout，看清输出规模
- dijkstra注意看有没有负环,有负权边换SPFA判
- 找欧拉路径的时候，注意（1）特判欧拉回路（2）起点（出度=入度+1）（3）要写充要条件来判断是否有欧拉路径/回路
- set中删除数操作前先判断数是否存在
- 看清题意，是有根数还是无根树，建图是双向边还是单向边，双向边记得开够数组
- strlen()记得先求出值再循环，不要$O(n)$变$O(n^2)$
- **离散化之后要注意新的值域**,主要是指用树状数组维护信息的时候
- 在多组数据的时候，一定要读入完成之后再特判，中途 continue 、break
- **define 最好加括号**

```
#define cal(x, y) x * (m + 2) + y//cal(i - 1, j)返回负数
#define cal(x, y) (x) * (m + 2) + (y)//这样就没有问题了
```

- 如果要进行**1e7**次以上的乘除运算，开**long long 100%**会TLE，`a[i]=1ll*b[i]*c[i]%mod;`
- “子串”、“子序列”、“连续的子序列”、“不连续的子串”...（还有可能出题人重新定义）
- memset使用头文件#include< cstring >，freopen使用头文件#include< cstdio >，
- 不要abs(long long)，会出错，最好手写。`return x>0 ? x : -x;`