目录

[一、 提高编写速度和质量](#_Toc29839_WPSOffice_Level1) [1](#_Toc29839_WPSOffice_Level1)

[二、 数据类型](#_Toc7035_WPSOffice_Level1) [2](#_Toc7035_WPSOffice_Level1)

[三、 函数](#_Toc14186_WPSOffice_Level1) [3](#_Toc14186_WPSOffice_Level1)

[四、 字符，字符串题](#_Toc23131_WPSOffice_Level1) [3](#_Toc23131_WPSOffice_Level1)

[五、 排序题](#_Toc31480_WPSOffice_Level1) [6](#_Toc31480_WPSOffice_Level1)

[六、 质数题](#_Toc29630_WPSOffice_Level1) [7](#_Toc29630_WPSOffice_Level1)

[七、 坐标题](#_Toc21701_WPSOffice_Level1) [7](#_Toc21701_WPSOffice_Level1)

[八、 链表题](#_Toc32277_WPSOffice_Level1) [7](#_Toc32277_WPSOffice_Level1)

[九、 二叉树](#_Toc20901_WPSOffice_Level1) [7](#_Toc20901_WPSOffice_Level1)

[十、 DFS/BFS题](#_Toc2663_WPSOffice_Level1) [9](#_Toc2663_WPSOffice_Level1)

[十一、 并查集题](#_Toc8191_WPSOffice_Level1) [10](#_Toc8191_WPSOffice_Level1)

[十二、 最短路题](#_Toc20749_WPSOffice_Level1) [10](#_Toc20749_WPSOffice_Level1)

[十三、 带有编号的题目](#_Toc5133_WPSOffice_Level1) [11](#_Toc5133_WPSOffice_Level1)

[十四、 C++ STL](#_Toc1406_WPSOffice_Level1) [11](#_Toc1406_WPSOffice_Level1)

[十五、 错误类型](#_Toc2826_WPSOffice_Level1) [13](#_Toc2826_WPSOffice_Level1)

[十六、 比赛考点](#_Toc4411_WPSOffice_Level1) [14](#_Toc4411_WPSOffice_Level1)

[十七、 可以选择比赛前巩固的模板题，加深印象](#_Toc3289_WPSOffice_Level1) [15](#_Toc3289_WPSOffice_Level1)

[十八、 易错题，熟知容易错的地方](#_Toc14171_WPSOffice_Level1) [16](#_Toc14171_WPSOffice_Level1)

[十九、 Windows 下的测试](#_Toc18417_WPSOffice_Level1) [18](#_Toc18417_WPSOffice_Level1)

[二十、 题目难度及变化](#_Toc27982_WPSOffice_Level1) [18](#_Toc27982_WPSOffice_Level1)

[二十一、 比赛规则](#_Toc8588_WPSOffice_Level1) [18](#_Toc8588_WPSOffice_Level1)

[二十二、 做题策略](#_Toc23238_WPSOffice_Level1) [18](#_Toc23238_WPSOffice_Level1)

[二十三、 新评测系统](#_Toc9323_WPSOffice_Level1) [19](#_Toc9323_WPSOffice_Level1)

[二十四、 比赛前注意](#_Toc16880_WPSOffice_Level1) [20](#_Toc16880_WPSOffice_Level1)

[二十五、 做题习惯](#_Toc10933_WPSOffice_Level1) [21](#_Toc10933_WPSOffice_Level1)

[二十六、 读题](#_Toc23132_WPSOffice_Level1) [21](#_Toc23132_WPSOffice_Level1)

[二十七、 做题前注意](#_Toc17328_WPSOffice_Level1) [21](#_Toc17328_WPSOffice_Level1)

[二十八、 做题技巧](#_Toc15434_WPSOffice_Level1) [21](#_Toc15434_WPSOffice_Level1)

[二十九、 不敢苟同的题目](#_Toc7273_WPSOffice_Level1) [22](#_Toc7273_WPSOffice_Level1)

1. 提高编写速度和质量

true->1 false->0

min(a,b) max(a,b) \_\_gcd(a,b)

main函数外的变量，初始值为0。

特别是数组，请尽量放在main函数外。

多个语句写在一行，有逗号隔开，这样可以省去加括号和换行的时间。但是break,continue,return等操作，不能写在一行。

Codeblocks 快捷键操作

Ctrl+L 删掉一行

Ctrl+D 复制一行

Ctrl+T 光标所在行与上一行发生交换

Ctrl+R 替换 !!!

Ctrl+Z 撤销 !!!

Dev C++ 不同。

利用maxn等变量修改数组范围，e.g. a[maxn]，这个因人而异。

多个类似的操作，函数替代，以此节省思考量 / 如果写错导致的调试时间，如坐标题。

判断对错，很多个标准

bool judge()

{

......

}

if (judge())

printf(“”);

else

printf(“”);

1. 数据类型

注意( [ ) ]

int [-2^31,2^31)

long long [-2^63,2^63)

short

unsigned int

unsigned short

两数相乘也许大于int范围，

首先结果的数据类型为long long，

其次，两数的数据类型至少有一个为long long，

否则要这样写1ll\*a\*b

--------------------------------------------------

判断两个实数是否相等

相差eps的都可以视为相等

eps可以取1e-8,1e-6,...

e.g.判断

x>y if (x>y+eps)

x<y if (x<y+-eps)

x==y if (y-eps<x && x<y+eps)

--------------------------------------------------

除法

double / double

double / int

int / double

不要

int / int

--------------------------------------------------

判断有没可能出现除0的情况。

--------------------------------------------------

读入：100(保存成实数类型)

scanf("%lf",&a);

保留k位小数

scanf(“%.2f”,a); //2位

scanf(“%.0f”,a); //四舍五入输出整数

--------------------------------------------------

可能出现-0.00，会输出-0.00

If (fabs(a)<eps)

Printf(“0.00”);

虽然出现可能性较低

1. 函数

如果有返回值的最好在最后加上return xxx，即使这句话没有作用

int xxx()

{

if ()

return xxx;

......

return yyy;

}

1. 字符，字符串题

看清

字符串的 输入 输出 格式

--------------------------------------------------

char str[100];

scanf("%s",str);

printf("%s",str);

Scanf读入字符串，以’ ’,’\n’,’\t’结束。

无法读入”abc def”这样的数据。

用fgets()。

以’\n’结束读入。不能读入某些’\n’。

处理方法：

scanf("%d\n",&a);

fgets(str,maxn,stdin);

或

scanf("%d",&a);

fgets(str,maxn,stdin);

fgets(str,maxn,stdin);

天梯赛评测系统，不能使用gets(str)

天梯赛评测系统，一定要去除末尾'\n'。

len=strlen(str)-1;

str[len]=0;

--------------------------------------------------

针对格式设计不同的输入输出

scanf("%c %c",&x,&y); ///跳过的是多个连续的空格

scanf(“%d:%d”,&a,&b); ///跳过一个字符‘:’

printf(“%05d”,a); ///用0补全不够的5位

--------------------------------------------------

熟悉字符串函数

strlen

strcpy

strcmp

strcat

strncpy (末尾加'\0')

不可使用delete。

如果会c++的string，也是挺好的，它比较全面。要用cin,cout读入读出。

.length()

=

.compare()

.append()

str.insert(pos,str1)

str.insert(pos,str1,str1\_begin\_pos,str1\_length)

str.erase(pos,length)

--------------------------------------------------

熟悉转换，如atoi，atof（负数也能处理）

请不要使用itoa，改为使用sprintf

如sprintf(str,”%d”,a);

--------------------------------------------------

读入：字符串长度为n，结束符长度为1，需要的存储大小至少为n+1

char a[10000+1];

char a[10000+10];

--------------------------------------------------

char a[maxn],b[maxn];

swap(a,b);

是两个指针的变换。

--------------------------------------------------

读入n\*m的字符矩阵

Scanf(“%d%d”,&n);

For (i=1;i<=n;i++)

Scanf(“%s”,str[i]+1);

for (i=1;i<=n;i++)

{

scanf("%c",&c);

for (j=1;j<=m;j++)

{

scanf("%c",&c);

}

}

1. 排序题

bool cmp(int a,int b)

{

return a<b;

}

--------------------------------------------------

仅一个关键字，倒序输出

sort(a,a+n);

for (i=n-1;i>=0;i--)

a[1]-a[n]

Sort(a+1,a+n+1)

--------------------------------------------------

两个关键字排序

E.g

struct node

{

int a;

char b[20];

bool operator<(node y)

{

if (a==y.a)

return strcmp(b,y.b)<0;

return a>y.a;

}

}

--------------------------------------------------

但是如果是多个关键字（三个关键字）的排序，这样写可能条例更清晰，不容易错

//第一关键字的排序

if (a.first>b.first)

return ;

else if (a.first<b.first)

return ;

else

{

//第二关键字的排序

...

}

字符串排序

c qsort 字符串排序.txt

注意c和c++使用方法的不同，比赛前不看模板，自行打一遍

1. 质数题

求1-n质数的方法

判断是否质数的方法

数字1不是质数

M=sqrt(n+eps); //自动int(xxx)

For (i=2;i<=m;i++)

1. 坐标题

做题前，先明确x,y,z坐标轴的定义，n,m的定义，读入方式。

两数相乘也许大于int范围，初始可使用long long类型。

g++:y1是保留变量，不能用！

改成clang++提交

1. 链表题

链表的大小

结束标志

莫忘记新创建一个链表，使用p=new node();

1. 二叉树

先看求什么，最小/大堆还是二叉搜索树（左边大还是右边大，相等的点怎么处理）。

--------------------------------------------------

编号：

负数 -> +delta 变为正整数

不给范围 -> map

--------------------------------------------------

前中后序遍历：关键是找到子树的根节点，然后得到左右两部分。

左右两部分有可能有空，要判断。

调整左右两部分的遍历顺序和输出的位置，使得一次解决给2输1的问题。

镜像和层次遍历是可以同时解决的，也是使用调整左右两部分的遍历顺序的方法。

--------------------------------------------------

一个点记录左右两部分的根节点，

Struct node

{

Int l,r;

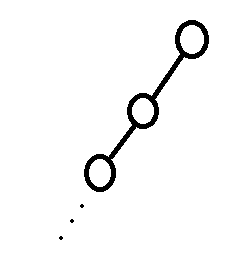
}f[maxn];

Int l[maxn],r[maxn];

不建议用链表的方式存储节点。

--------------------------------------------------

二叉搜索树：不建议用f[index\_of\_tree]存储节点，因为可能出现以下情况。用数组记录每个点的关系，如层数，父节点，左右节点。



堆：如最小堆，点的编号为1-n。每次对两个点进行交换，如果使用上述方式记录层数，父节点，左右节点，处理会比较麻烦。可以选择用编号来判断点与点之间的关系。

--------------------------------------------------

读入很多变量(e.g.L3-016 二叉搜索树的结构)，数组名字请避开这些变量的名字，如我的习惯是用z/tr数组记录堆的信息，单个整形变量的命名分别为a,b,...。

整形变量和字符串变量要区分，我的习惯是整形变量命名为a,b,c,字符串变量命名为d,e,f,g。

--------------------------------------------------

Yes/No 用函数写，如print(boolean);

--------------------------------------------------

scanf,strcmp可能写起来比较麻烦，可以考虑用c++的string，cin,==。

--------------------------------------------------

有可能出现点不在二叉树中的情况。如果代码有错误，请不妨往这点考虑。

1. DFS/BFS题

初始化。

数组的大小。

--------------------------------------------------

不要忘了输出如“No Solution”这样的内容，放在最末尾；若找到解，exit(0)。

--------------------------------------------------

DFS，是否存在爆栈的情况。

\*折半搜索。

\*状态压缩。中间点的状态无影响。

\*迭代搜索。

\*A\*。

--------------------------------------------------

=true

+

Dfs()

-

=false

If (the step satisfies the condition)

{

Return;

}

--------------------------------------------------

BFS内存不足，使用循环队列，较少遇见。

\*双向广度优先搜索。

--------------------------------------------------

如果时间不是特别充裕，优化的写法耗时较长，有些题目直接使用暴力解法即可。

--------------------------------------------------

剪枝：

1. 当前情况已无法使后面的某个地方满足条件，return
2. 当前情况下，即使后面取最优情况，也无法使最终结果好于当前最优解，return

……

一旦层数加深，情况个数会很增长很快（m!），所以剪枝是减少遍历的情况个数的有效手段。

1. 并查集题

首先应该判断这道题是否能用并查集实现(e.g. 相同属性的点在一个集合中)。

初始化 fa[i]=i;

个数 0-n-1 / 1-n / 00..0 - 99..9

最后还要实现一次getfather()。

使用路径压缩，参见代码。

1. 最短路题

首先应该看懂题目再下手！！！

确定输入输出方法。

--------------------------------------------------

编号

若存在编号为0，则存储最小点编号的变量 的初始值不能为0，如可以为n。

For循环从0-n-1，而不是从1-n。

如需要输出编号，对编号进行还原，结束编号不妨设为-1，不能为while (d)。

参见天梯地图该题。

--------------------------------------------------

初始化 dist,vis,初始点,输出的值...

--------------------------------------------------

边是单向还是双向。

是否有重边。是否node \* / Road[x][y]的存储方式。

--------------------------------------------------

一般不用SPFA，怕TLE，但天梯赛一般没问题。

对于Dijkstra，若得到最新的点为重点，即可退出，其它点不会使该点的距离变小，或增加最短路总数。已出现的点不能再出现，使用vis[d]=1，if (!vis[d] && ...)。

能不使用优先队列，就不用，因为前者写起来更复杂。

优先队列，如果使用重载运算符，下方的第二个const 一定要有。

Bool operator(const node &y) const

如果可以使用Floyd，初始化使用0x3f吧。memset(dist,0x3f,sizeof(dist))。写Floyd，减少代码量，但如果不熟悉，还是写Dijkstra好了。

--------------------------------------------------

能不使用结构体存储边，就不用，因为前者写起来更复杂。但注意一定要是在没有重边，题目不要求统计最短路的条数的情况下。

哎，还是写结构体吧，因为天梯赛最短路都不写限制条件，就是等着让你掉坑的。

结构体要写p->to，否则死循环。

代码也许存在多个结构体，注意不要写混了。

--------------------------------------------------

是否有额外的输出，如”No Solution.”。

--------------------------------------------------

也许会出现long long，但可能性较小。

--------------------------------------------------

要有大局观，如果只剩下10分钟，还是骗分或者做其它题更好。

1. 带有编号的题目

是否给出范围，如果没有，可使用map。

输出格式，是否要补全0之类的。

是否要求判断编号是否合法。如不能是负数。

存在编号是0，则需要从0开始，n-1结束。比较的数的下标不能为0。-0的字符串处理。

1. C++ STL

有必要学一下

#include <map>

Key -> Value

是否出现；赋值

unordered\_map某些方面比map快，但天梯赛一般不会出现用map超时，用unordered\_map不超时的情况

map<x,y> ma;

x,y均为数据类型

ma.clear(); //清空

ma[x]=y; //赋值

if (ma.find(x)==ma.end()) //x不在ma中

xxx

printf("%d",ma[x]); //前提是x在ma中

ma.erase(key);

map也可以遍历，方法和vector类似，获得的是key值。

--------------------------------------------------

#include <set>

Set<int> se;

去重；求个数

se.clear(); //清空

se.size(); //统计个数

se.insert(); //添加

--------------------------------------------------

#include <stack>

Stack<int>st;

St.push(x); //加入

St.pop(); //删除

X=St.top(); //顶端

St.empty(); //是否空

//清空 如用于Dijkstra+堆优化

While (!st.empty())

St.pop();

--------------------------------------------------

#include <vector>

Vector<int> vec;

Vector<int>::iterator j;

Vec.clear();

Vec.push\_back()

Vec.insert(pos,vaule);

Vec.erase(pos);

Vec.begin();

Vec.end();

Vec[index]

For (j=vec.begin();j!=vec.end();j++)

Printf(“\*j ”);

for (j=vec.begin();j!=vec.end();j++)

if (\*j==value)

vec.erase(j),j--;

--------------------------------------------------

二分

a[0]-a[n-1] 从小到大

lower\_bound(a,a+n,value); //>=

upper\_bound(a,a+n,value); //>

get pointer

1. 错误类型

|  |  |
| --- | --- |
| **结果** | **说明** |
| 等待评测 | 评测系统还没有评测到这个提交，请稍候 |
| 正在评测 | 评测系统正在评测，稍后会有结果 |
| 编译错误 | 您提交的代码无法完成编译，点击“编译错误”可以看到编译器输出的错误信息 |
| 答案正确 | 恭喜！您通过了这道题 |
| 部分正确 | 您的代码只通过了部分测试点，请继续努力！ |
| 格式错误 | 您的程序输出的格式不符合要求（比如空格和换行与要求不一致） |
| 答案错误 | 您的程序未能对评测系统的数据返回正确的结果 |
| 运行超时 | 您的程序未能在规定时间内运行结束 |
| 内存超限 | 您的程序使用了超过限制的内存 |
| 异常退出 | 您的程序运行时发生了错误 |
| 返回非零 | 您的程序结束时返回值非 0，如果使用 C 或 C++ 语言，要保证 int main 函数最终 return 0 |
| 浮点错误 | 您的程序运行时发生浮点错误，比如遇到了除以 0 的情况 |
| 段错误 | 您的程序发生段错误，可能是数组越界、堆栈溢出（比如递归调用层数太多）等情况引起 |
| 多种错误 | 您的程序对不同的测试点出现不同的错误 |
| 内部错误 | 评测系统发生内部错误，无法评测。工作人员会努力排查此种错误 |

段错误

数据范围写小了。

特殊样例。

代码写错了(下标)。

格式错误

空格改成'\_'进行测试。

造几个有针对性的数据。

不要理所当然，回去再看题意。

运行超时

死循环 可以尝试加return，看是在哪里出现错误。但最好还是把代码重新看一遍，分析错误。

(较少部分可能是数组的范围问题)。

方法不优，看情况选择是否继续优化。

内存超限

要对计算内存大小有个粗略的了解。

有时会出现栈溢出的情况，慎用DFS。

减少内存，卡在最大内存边缘，骗得尽量多的分。

如读入一些数，这些数可能很大，使用数组存储。

编译错误

如gets改为fgets(std,maxn,stdin);

系统：把c改为c++，代码是没有发生改变的，不必再拷贝一次。

浮点错误

/0

多种错误

那就依次考虑有可能出现的错误类型。不过出现可能性较少。

1. 比赛考点

几大考点

排序

DFS/BFS

并查集

最短路

二叉树（前中后序遍历，层次搜索，旋转左右节点，二叉查找树，最小堆）

链表

高精度（作为某些题的最后几分）

几何:

以清华大学邓俊辉教授为首，

出的几何题都比较难，

建议暴力剪枝，骗分为主

包括

斜率DP L3-009 长城

几何+剪枝 / 凸包 L3-012 水果忍者

目前还没有正解 L3-021 神坛

--------------------------------------------------

出现过的其它算法：

大模拟 L3-019 代码排版

dp L3-020 至多删三个字符

基础几何+大模拟 L3-005 垃圾箱分布

基础几何+变形dijkstra 森森美图

树状数组+二分 / 神仙二分堆 L3-002 特殊堆栈

贪心+线段树 L3-017 森森快递

--------------------------------------------------

另外可能还会出现的算法：

最小生成树

更多样的dp

几何创新题（熟悉极角排序，叉积）

更灵活的最短路，二叉树

更隐蔽的并查集，DFS，BFS

不排除给出几道很裸的模板题

1. 可以选择比赛前巩固的模板题，加深印象

L1025

字符串（判断是否符合要求）

L1-032

字符串（输入方式，输出格式）

L1-043 阅览室

字符串（除0，四舍五入输出整数）

L2-022

链表（反转）

L2011

二叉树（前中后序遍历，镜像，层级遍历）

L2-012

最小堆

L3016

二叉排序树

L2017

单变量排序

L2027

多变量排序

L2010

并查集（关系）

L3003

并查集（集合数） / DFS（集合数）

L3011

最短路 Dijkstra,Dijkstra+堆优化

L3-014

最短路（优化）/ BFS（证明）

L3-004

纯BFS

L3-015

DFS（优化，得看数据的情况）

L3009

凸包

L3-012

DFS（优化） / 凸包（证明）

L3020

DP（推导）

1. 易错题，熟知容易错的地方

L1-006 连续因子

最长连续因子个数为1。

整数越界。

L1-025 正整数A+B

回顾考点。

要求输入的是数字，大小在[1,1000]范围内。

1. 字符全是‘0’-‘9’。
2. 字符不能全为‘0’。
3. 字符串长度小于等于3，或字符串为“0000”。

L1-032 Left-pad

输入一行，有空格，fget，去除’\n’。

没说输入字符串的长度。

创建一个足够大的数组 / 逐个字符去除并处理。

L1-039 古风排版

明确坐标。

字符串之外的输出’ ’。

L2-013 红色警报

输出的两个语句仅相差一个字符。

L2-016 愿天下有情人都是失散多年的兄妹

输出 字符串不足k位用’0’补全。

多组输出，也许要初始化。

输入数据存在额外信息，输入“本人ID 性别 父亲ID 母亲ID

”，两个ID要记录性别。

编号的范围00..0-99.9，而不是输入的数据数N。

L2-022 重排链表

链表长度不一定为N。

不超过M，即存在等于M的情况。

L2-023 图着色问题

“问可否用K种颜色为V中的每一个顶点分配一种颜色”，颜色数量要求恰好为K。

L2-028 秀恩爱分得快

-0的字符串处理。

询问的点可能尚未曾出现。

L3-011 直捣黄龙

“第二行顺序输出最快进攻路径的条数、最短进攻距离、歼敌总数”。

输出很不明确。

L3-015 球队“食物链”

“战胜过”：包括主场和客场。

L3-016 二叉搜索树的结构

点的编号很大。

点有可能未曾出现在树中。

1. Windows 下的测试

windows

比对输出

fc 1.txt 2.txt

运行

x.exe

x.exe <y.txt

x.exe <y.txt >z.txt

1. 题目难度及变化

仅供参考

第一部分和第二部分的超纲题较少。仍然保持出基础题，模板题的传统。

第三部分的题目越来越灵活，越来越难。但骗分是毫无问题的。

仍然保持着出神仙数据的习惯。一些特殊样例很难想到，需要非常细心才行，请自行取舍一道题拿下剩余的分数是否在时间上划算。

1. 比赛规则

再强调一下

竞赛的预定时长为3小时

参赛队员仅可以携带无计算功能的铅笔或水笔入场

不能携带任何可用计算机处理的软件或数据

程序须经过若干测试用例的测试，每个测试用例分配一定分数

可以反复提交，取最高分，提交错误不扣分

1. 做题策略

1.

参赛队员可以在比赛中的任何时刻尝试解决任何梯级的题目。但只有当一支队伍的基础题总分超过800分时，该队进阶部分的题目分数才被判为有效；只有当其进阶题总分超过400分时，该队登顶部分的题目分数才被判为有效。

基础欠缺的同学努力练习基础题。

最好规定个时间，做8道第一部分的题或8道第一部分的题和4道第二部分的题。

以最高分作为目的。要对时间有把握。

可以针对自己容易错误的内容进行总结，并进行针对训练。

比赛剩下1个小时/0.5个小时时看一下团队榜，如果真的遇到第一部分不足800分，第二部分不足400分的时候，部分同学可能需要回头拿下第一部分和第二部分的分数。

2.

仅供参考。

选择第一部分八道题 5分两道，10分两道，15分两道，20分两道的顺序做题，分四个阶段。

第二部分的题目，如果遇到短时间无法解决的题目，跳过先，把有把握的题目的先做完，再把剩下的题目做完。当然如果过了不少时间仍然没有思路，跳过先。

看一下时间。如果有模板题，先把模板题做了。如果遇到很难或者短时间拿满分的题目，果断骗分，然后再也不看这道题目。

如果时间只剩下10分钟以内(或之类的)，短时间无法解决第三部分的某道题，即使你会做这道题，也要果断放弃这道题。选择 这道题骗分 / 回到第一部分和第二部分把剩下的分数拿回来(主要是检查错误，发现特殊情况，再仔细看一下题目和自己的代码)。

当然，如果还有大片的题目没有完成，然后时间所剩不多，如1个小时，还有第二部分两道题和第三部分的题目没有完成，看队伍情况。如果队伍第二部分离400分还有很远，那么以做第二部分为主，否则以尽量拿到最多分为主，第二部分做把握的题目+第三部分骗分。估计这时心情也是紧张的，所以就不要做那些你认为短时间无法完成编写和调试的题目了。

3.

在进阶和登顶阶段最先完整获得其中任一题分数的前5位队员，分别依次获得50、40、30、20、10分“先锋奖励”。该奖励分数计入个人总分。当团队进阶或登顶成功后，也计入团队的有效分。

这个看自己的能力了。一来高手如云，二来我们赛区的网络不是很好，开始时间也可能会稍微慢点，估计拿到先锋奖励较难。以我们的现实情况，我不建议这样做。

当然，如果你平时的做题策略就是先第二部分和第三部分，那么这没有问题。但如果平时第二部分和第三部分的题的完成率就不是太高，那我就强烈不建议这样做了。

如果你要这样做，建议平时多训练几次这样的做题技巧，比赛时，果断选择裸题做。

1. 新评测系统

不小心按了右键：按Close Menu

再一次复制：直接按Ctrl+C即可，不需要再次选中

不能同时打开多个窗口，多个界面。

修改未完全正确的代码：从下方框中拷贝。

提前打开榜单，搜索“xxx”，得到唯一的队伍信息。

榜单不会自动刷新，要重新点击才能看到最新情况。

正确看待榜单有可能出现问题的情况。

正式比赛是1:00-4:00。

1. 比赛前注意

对编译环境进行测试

要能实现代码运行、调试

比赛前打好基本模板（应该允许的，见muban.txt），保存成txt放在桌面，需要的时候拷贝之。

如果允许打其它模板，把BFS、并查集、最短路、搜索二叉树、字符串排序等的模板打一下，保存成txt放在桌面，注意命名。前提是真的可以，最好问清楚！

我忘记了去年是否有杀毒软件，如果有，我强烈建议关掉它们！

去年好像电脑有多个输入法，我的建议是去掉只剩一个。

Codeblocks和Dev c++ 字体的大小，最好调一下。

去年遇到某些浏览器不能使用的情况，注意老师的最新通知！

去年好像是不禁外网的（神坑），我用360浏览器，然后一不小心就拖动了网页内容，然后进行百度搜索了。。。

当然，利用网页搜索题目，想也知道，这是绝对不行的！

当然这要是不小心碰上了，请镇定自若的关闭之。

去年还遇到一个比较坑的事情，就是我提交代码时，发现网页框里有别人的代码！！！

遇到这样的事情，镇定自若，不要慌张，这样的事情其它同学大概率也会遇到！不要贪小便宜，遇到这样的事情，肯定有比较完善的补救措施，不要脑袋一热，做出错误的事情！

机子，键盘不行的话，请及时联系老师更换

运到编译器的问题，尽快联系老师解决

去年我的电脑就是有点小问题，碰一下屏幕，屏幕就没有内容，还有同学反应有些键盘比较难用，像这样的问题，尽管问老师，看能否换机，换键盘！

1. 做题习惯

因人而异

模板保存成txt放在桌面并打开，当做对一道题，拷贝之，继续做下一题。比赛前应该允许打模板，打的尽量详细一点。

一道题做错，创建新的工程（比赛前提前创建），做下一题。

初始代码的maxn写大一点没关系（如1e5），一般都不会内存超限。

一般人名的长度都不会大于50/100，直接写就是。

很多个整数读进来，cin>>a>>b>>...可能比scanf好。如果有多种不同类型，scanf可一一比对输入数据的类型，全加上&比较好（字符串用&，结果相同）。

多组数据，注意初始化，天梯赛中，建议是用while (q--)，避免重复使用变量。

比赛也许很多人提交，如果评测很慢，可以打开其它题目后再看。

如果通过代码找不到错误，请不要忘了造数据测试！

1. 读题

特别是对于难理解题意、输出要求复杂的题目，先认真题目，才开始做题。

对样例实现一遍：手算、画图。

遇到特殊题目，或完全不理解题意的题目，用一个特别暴力、简短的代码实现一下。

1. 做题前注意

求的是啥 (特别是天梯赛的最短路题，谨记！)

数据范围

输入输出格式

是否多组数据（注意初始化）

特殊数据

1. 做题技巧

比赛如果拿高分：

1.平时刻苦的训练

2.比赛时良好的心态

3.骗分大法

......

4.要有大局观。剩下的时间，如果选择题目和对应的做法拿到更高的分数。

如果时间不多，对于某些题目要果断舍弃。如果有些错误难以发现，先跳过，有时候，天梯赛的特殊样例太坑了。思考错误有可能发生在哪里，花时间检查错误是否合算，不要慌张，不要盲目的检查错误。

如舍弃第一部分和第二部分无论如何都发现不了错误的2,3分，舍弃第三部分的大模拟题，拿下第三部分的其中一道用暴力的算法可以获得很多分数的题目。

时间把控

Part1:

25min-30min

30min-40min

40min-60min

60min-90min

90min-120min

Part2:

30min-40min

40min-55min

55min-80min

80min-120min

120min-150min

大致分割，请对号入座，平时加以训练，要对大致时间有把握。

赛前把易错题和薄弱点再看一遍。

1. 不敢苟同的题目

L3-021 神坛

目前网上的方法都是错误，能举出反例。

这么简洁的题目，个人认为无法找到一个非常漂亮的解法[ O(n^2logn) ]。

L3-015 球队“食物链”

可能是因为数据量太少了，错误很明显的代码交上去都对了。建议用正确代码对拍一下，造多点数据。

剪枝要看数据，俗话说，靠脸。

虽然出题人的本意是挺好的，但是这数据量……