想讲的东西

Bug总结

编译能查出来的bug

- 中文符号
- scanf不加取地址符，或者误用取地址符
- 运算符的优先级（可以开-Wall来发现可能的错误）

```c
#include <stdio.h>
int main(){
    int a=1;
    if(a>=1 && a<=9 || a>11){ //&& 和 ||的优先级
        printf("haha");
    }
    if(a<<1+1==0){ //<<和+的优先级
        printf("hello");
    }
    return 0;
}
```

- 返回值相关

```c
#include <stdio.h>
int f(int x);
int main(){
    int x=5;
    printf("%d\n",f(x));
    return 0;
}
int f(int x){
    x=x+1;
    //没有返回值，报错
}
```

列文虎克式bug

- 1和l
- EOF
- %11d
- d和b
- n和m
- 超过两重循环时的循环变量

```
1,l   l,r
%lld %11d
EOF,EOF
d,b
n,m
i,j,k  题目中k为其他意义，但解决问题还需要超过两重循环的情况
```

对C语言某些特点不了解导致的bug

- if-else ,while,for后不正确的使用了分号
- 连续赋值操作

```
#include <stdio.h>
int main()
{
    int x=10;
    if(0<=x<=3){
        printf("haha");
    }
    return 0;
}
```

- 局部变量未初始化就使用。（本地测试即错，或者提交WA）
- 数组开小/数组访问时下标越界（int a[N]后能访问的下标范围是0到$N-1$，我们一般会稍微开大一点）（REG/RE或者WA）
- 浮点数除法与整数除法，浮点数精度，浮点数相等的判断，比如：

```
#include <stdio.h>
int main()
{
    double a=1000000009999999999;
    printf("%lf\n",a/1.0);
    return 0;
}
```

- scanf()，getchar()，fgets()等各种读入。读入带空格的字符串，读入所有字符，读入带有一定结构特征的“字符串”，单字符读入数据的火神读词法，快速读入（字符读入模拟整数的读入）。

```
#include <stdio.h>
int main()
{
    //需求：想读入一个字符串之后，再读入一个非空白字符
    //比如我们想让s里存"haha"，让c里存'a'
    //我们的输入是：
    //haha
    //a
    char s[100];
    char c;
    // fgets(s,100,stdin);
    scanf("%s",s);
    getchar(); //吞回车用，如果是scanf读入字符串则需要吞回车，如果是fgets读入字符串则不需要
    c=getchar();
```

```
    printf("%c",c);
    return 0;
}
```

- 运算符优先级
- 函数的错误调用，看不懂函数的定义（课本附录或者cplusplus.com 或者cppreference.com，看官方文档，举例ceil函数）
- 字符与整数，ASCII码

```
#include <stdio.h>
int main(){
    char c='a';
    printf("%d\n",c); //打印ASCII码
    printf("%c\n",c); //打印字符本身
    return 0;
}
```

- 函数返回值忘了用了

```
#include <stdio.h>
int f(int x);
int main(){
    int x=5;
    f(x); //本来想的是把x改成f(x)，但是忘记修改
    //应该是x=f(x)
    printf("%d\n",x);
    return 0;
}
int f(int x){
    x=x+1;
    return x;
}
```

其他的bug

- while循环时没修改循环变量 (TLE)
- for循环时多修改了循环变量（错误的循环次数）
- 爆int范围（WA，单个数不爆，求和或者乘积爆），声明了错误的数据类型（整数与浮点数，int和ll)
- 忘了取模，取模模错数了（建议复制粘贴）
- 循环次数（0到$n-1$？ 0到n？ 1到n?）
- 无穷大不够大，或者太大

```
#define INF 1000009
int maxv=9999999999;
```

- 输出格式，多组数据的时候的换行，空格导致的PE或者WA
- 多组数据没清空用过的变量或者数组
- UB(未定义行为)

```c
#include <stdio.h>
int a[5];
int main(){
    int i=0;
    a[i]=i++; //编译器不同，可能得到的结果也不一样
    return 0;
}
```

如何高效的写程序（？）

- 想清楚逻辑再写，thinking twice,coding once.
- 多练，提高熟练度
- ？

如何测试自己的程序

- 静态测试：桌面检查（对着表查bug），代码审查（对着表查bug），代码走查（人脑运行程序代码），其中查表找bug多用于软件测试中，规模比较小的程序的测试没必要对着表查，可以等真出了问题再查表。
- printf输出调试，gdb调试
- 对拍（快速随机生成输入数据，让自己写的程序和别人的程序一起输入相同的数据，比较程序输出是否相同，若不同则至少有一个程序存在bug）学习对拍
- 更多的方法……

时间优化（程设难度？）

以空间换时间

先进行预处理，把一些信息先算出来，存储到某些地方，然后再进行其他事情

- 前缀和

  多用于询问操作很多，但没有修改操作或者修改操作极少的问题中

  如金仙花数，若题目给出更大的询问范围，更多询问次数，则需要用这种方法

- 差分

  多用于修改操作很多，但是查询操作很少的场景中

- 更一般的记忆化

  可以理解为做一张表，把之前算过的结果存在表里，当要用到某个之前算过的结果的时候，直接查表得到结果

一些其他的空间换时间的例子：计数排序

```c
#define N=100009
#define M=10000009
int a[N],cnt[M]={0}; //至多有N个数，且每个数都在0-M-1之间
for(i=0;i<N;i++){
    cnt[a[i]]++;
}
for(i=0;i<M;i++){
    if(cnt[i]!=0){
        for(j=0;j<cnt[i];j++){
            printf("%d ",i);
        }
    }
}
```

数学方法

排列组合相关知识

- 排列组合中的一些基本方法，比如隔板法（球盒），插空法，递推计数（台阶），容斥原理（倍数）等
- 特殊的数列：斐波那契数列，两类斯特林数，卡特兰数（规范01数列）等

小学奥数难度的数论知识，如最大公因数，质数等

线性代数相关知识，如矩阵加速递推

减少枚举，搜索剪枝与记忆化

尽可能地发现提前中止的条件，利用这些条件使得循环及时break，递归及时return

记忆化就是把搜索过程中已经算出来的一些中间问题的结果存起来，当今后又要算这个问题时直接查看结果（台阶问题）

学习并改造经典算法*

一些已经出现过的例子：埃氏筛解a*b+c，冒泡or归并排序求逆序对（基础物理实验）

不久之后还可能会遇到的：欧几里得算法求二元一次方程的特殊整数解，发现单调性使用二分答案，快排求第k小等

空间优化

观察题目特点，使用滚动数组优化空间（如《百团大战》）

代码可读性

多读别人的代码，学习长处

缩进，大括号（可以考虑用IDE或者编辑器的代码格式化功能）

变量命名风格（建议：题目中提到的量，用题目中给的字母，未提到的量可以用简单字母或者有意义的简化的英文单词，比如cnt，temp(tmp)等，实在不济可以稍微用一下拼音）

if else优化

DRY原则（Don't repeat yourself!）

一些“额外”知识以及学习途径

将要学但还没学到的语法知识，可以通过看课本或者课件来学习

你能想到的几乎所有接触过的数学知识，常用的有：高中计数原理知识，小学奥数，数论，高中解析几何里的一些内容，数分（求极限，求导等）高代（高斯消元法，计算行列式，矩阵求逆等）中的一些内容

一些基本算法（不过会学到），比如dfs，bfs，二分查找/二分答案，贪心思想（小狗过河），动态规划，排序，高精度等

可以通过查找教程或者博客来学习，或者去OI-Wiki，再不济可以搜索一些网课或者B站视频

拿到部分分之后的做法

ps:我脑子转得比较慢，场上压力大

如果是因为TLE拿到的部分分，那么一般是因为设计的算法效率不够高，可以考虑上面提到的优化方法。

如果是因为WA拿到的部分分，有一些原因是上面提到的bug，有一些是因为对问题的考虑不够周全，小部分时候是问题考虑错了。这时候需要重新分析问题，以及对程序进行调试（比如输出中间变量）。

部分正确的话，bug不一定会好找，尤其是“万绿丛中一点红”的时候，这个时候参考bug清单会是个好办法。

由于我们的程设课程最后期末考试是看得分而不是过题数的，所以如果你已经拿到了大量部分分的话，可以考虑先做其他题，然后回过头来再拿剩下的分数。

如果是课下做题长时间被卡住，可以考虑点击提交记录，找到和你拿了相同部分分，但是后来拿了满分的同学，在qq或者微信群中找到他/她并加微信询问对方是改了什么错误才过的，也是个不错的方法。

实在找不到问题时，代码重新写一遍可能就过了，或者睡一觉就找到问题了。