1、

1. A 永远的王 聪明王

这道题目可以看到 科目和数字相对应的

所以首先考虑到结构体数组。

然后简单的使用快排来排序就行了

需要注意的是：关于时间复杂度O(n)的考虑，快速排序的平均复杂度是O(log n)，《啊哈算法》中的快速排序，所采用的 基准数，都是直接找数列的最左边的数字。这样的缺陷是 在某些情况下（例如本来就已经排好序的数、各项都相等的数），时间复杂度达到了和冒泡排序的一样的高复杂度—O(N²),所以采用了杨新宇学长那次给的优化过的算法—直接找到中间的变量（结构体）充当基准数，这样来看，时间将得到很大的优化。

但是有一个需要考虑的点是：这是带上结构体的快排。

平时都是简单的对数字单个数据类型排序。现在使用结构体，所以在之前有交换的部分，都需要转换成 结构体 的交换。

考虑到 结构体可以进行整体赋值运算

非常符合交换操作。

当然 在数值比较的时候 仍然要使用结构体内部 的那个int进行大小比较。

通俗的 说就是: int 为主力进行比较，char（科目）被打包到结构体里面跟着跑。

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

struct S{

char subject[2000];

long int score;

}subject[1001],p,temp;

void Qsort(int left,int right){

int i,j,t;

if(left>right){

return ;

}

temp=subject[left];

i=left;

j=right;

while(i!=j){

while(subject[j].score>=temp.score&&i<j){

j--;

}

while(subject[i].score<=temp.score&&i<j){

i++;

}

if(i<j){

p=subject[i];

subject[i]=subject[j];

subject[j]=p;

}

}

subject[left]=subject[i];

subject[i]=temp;

Qsort(left,i-1);

Qsort(i+1,right);

}

int main(){

long int num=0;

scanf("%ld",&num);

struct S t;

for(long int i=1;i<=num;i++){

scanf("%s %ld",subject[i].subject,&subject[i].score);

}

Qsort(1,num);

for(long int i=1;i<=num;i++){

printf("%s ",subject[i].subject);

}

return 0;

}

8

这道题其实模拟几次就可以发现规律

因为所有正整数都是 1 的整数倍

其实就是从2开始的人开始计算

简单模拟发现 平方数才是符合规律的

如1 4 9…

所以 — 问题本质就是只需要找平方数

（但是无论用啥方法，sqrt函数也好、自己写也好，依然超时了几百毫秒，O（n）优化还在寻找更好的方法。）

三、L-菜学长的糖糖

这道题 糖都有各自的编号，糖是有数量的

那么很明显 这题可以用到 桶排序。

编号为1 的糖 则 a[1]++;

最后只需要比较a[i]哪个是奇数即可。

#include <stdio.h>

#include <math.h>

int Is(int num){//做一个简单的判断是否为平方数的函数

int n=2,rlt=0;

if(sqrt(num)==(int)sqrt(num)){//这里很重要 判断是否为一个平方数

return 1;

}

return 0;

}

int main(){

int m=0;

scanf("%d",&m);

printf("1 ");//无论如何 第一个肯定是站起来的

for(int i=2;i<=m;i++){ //因为所有的数都是 1 的整数倍 所以其实就是默认都站起来了

if(Is(i)){

printf("%d ",i);

}

}

return 0;

}

M-水题

这题画图即可。

拿到题目读题不仔细 导致自己一直以为是两个点到原点的距离差

在数轴上画出来就会有几种分类讨论。

K和N的大小关系分别进行比较。

相等/B再在A 右边的时候，无需移动，直接出0

K>N，A移动到K点即可，出K-A

K<A，B点可能在A点左边找到。

模拟一下即可知道，奇偶性相同的时候不用管，不同的时候，输出1

提示：这里对文章进行总结：

例如：以上就是今天要讲的内容，本文仅仅简单介绍了pandas的使用，而pandas提供了大量能使我们快速便捷地处理数据的函数和方法。

K-小布特烦恼

这道题仔细看描述就知道，很明显的的一个冒泡排序的题目；

然后加上了“不能超过 一定次数”的限定

所以加了个cnt计数，到达规定的次数直接 No并且return。

如果一直没有触发机制 就yes

此外，此题的输出格式我也进行了一番思考

这类题目 有一种在OJ平台上 是可以 输入一组 立马输出 的

但是我写出来这种方法后，发现似乎不行。

于是进行改进，利用二维数组存储每次的数据

输入完全完成之后，再进行遍历输出

（只可惜依然无法成功A题目…）

#include <stdio.h>

#include <stdlib.h>

//K题：如果想要做到完全输入之后才输出，那就需要考虑二维数组存放。

//可考虑先用一边输入一边输出的方法，这样可以保证先做出来，A的过去。此后有时间可以考虑使用二维数组，锻炼能力

void Is(int a[],int x);

int main(){

int n=0;

scanf("%d",&n);

for(int i=0;i<n;i++){//最外围的大循环，用来分别输出每一次的yes no

int \*a=NULL,x;

scanf("%d",&x);

a=(int \*)malloc(x\*sizeof(int));//开辟动态数组，此时a[x]就可以用了。

for(int j=0;j<x;j++){

scanf("%d",&a[j]);//放入数据

}

getchar();//吞字符

Is(a,x);

free(a);//一定记着要释放内存 否则会出现异常

}

}

void Is(int a[],int x){//可以如此表示数组参数

int i=0,j=0,cnt=0,t=0;

int ex=x\*(x-1)/2 -1;

for(i=0;i<x-1;i++){

for(j=0;j<x-i;j++){

if(a[j]<a[j+1]){

t=a[j];

a[j+1]=a[j];

a[j+1]=t;

cnt++;

}

if(cnt>ex){//每次cnt自己加完了后 就检查一次，一旦过线就直接No退出

printf("No\n");

return ;

}

}

}

printf("Yes\n");//如果排完序了后 依然没有进入那个底线，那么就说明可以输出Yes

return ;//要记得void也需要return

}

F-桃花侠大战菊花怪

这道题其实涉及到博弈论的知识

虽然现在没学习过

但是了解过大致知道了 是通过一系列数学知识，（这个游戏必须理论上双方是50％的胜率），找到一种方法保证最后的胜利。

这题输出结果只有Win/Lose

其实就是让我们模拟、判断出到底一直是Win还是Lose

总结

这次新生赛，最大的收获就是自己对于知识理解和题目解读的不足。

从以下几个方面可以看出。

1、做题经验差

做博弈论题目的时候，哪怕自己没学过博弈论相关知识，但是应该仔细思考，结果只有两种“Win” or “Lose”。哪怕自己一窍不通，也应该知道输出两个内容的一个即可。

此外，模拟思想也不具备，做 蹲下起立和博弈论题目的时候，总是妄想推公式一样的把结论推导出来。但是其实自己多试一试几个样例，就可以找到规律（例如平方数才是需要找的起立的人数），从而使得编程的时候，工作量、思维量减少很多

2.纯裸写代码能力不足

做第一题，快速排序和结构体的知识点，自己其实是以及掌握了的，但是代码永远不是很完善，了解算法原理后，不能做到自己从0开始写出一个完整的算法，总是会缺斤少两。

3、读题能力差

做有关于考察冒泡排序题目的时候，心里看到相关规则的确知道在考察冒泡排序。但是却总是觉得，除了冒泡还在考察其他东西，以至于自己把题目想的很复杂，无从下手。如果真正读懂题意，也就不至于在考场上认为这是一道高难度题目了。