NOIP2018初赛总结(提高组)

T1.下列四个不同进制的数中，与其它三项数值上不相等的是

A.(269)16

B.(617)10

C.(1151)8

D.(1001101011)2

答案：D

实际上D比ABC要多2。这种题我一般先算2,8,16进制的，十进制难算，那三个很好互相转。

T2.下列属于解释执行的程序设计语言是

A.C

B.C++

C.Pascal

D.Python

答案：D

常识题。前三项都是编译型语言。

T3.中国计算机学会于?年创办全国青少年计算机程序设计竞赛。

A.1983

B.1984

C.1985

D.1986

答案：B

CCF颂歌题。凭感觉做题即可

upd：附上[官方地址](http://www.noi.cn/newsview.html?id=66&hash=9CB0C5&type=5)

背景：1984年邓小平指出：“计算机的普及要从娃娃做起。”中国计算机学会于1984年创办全国青少年计算机程序设计竞赛（简 称：NOI），当年参加竞赛的有8000多人。这一新的活动形式受到党和政府的关怀，得到社会各界的关注与支持。中央领导王震同志出席了首届竞赛发 奖大会，并对此项活动给予了充分肯定。从此每年一次NOI活动，吸引越来越多的青少年投身其中。十几年来，通过竞赛活动培养和发现了大批计算机爱好者，选 拔出了许多优秀的计算机后备人才。当年的许多选手已成为计算机硕士、博士，有的已经走上计算机科研岗位。

T4.设根节点深度为0，一棵深度为h的满k(k>1)叉树，即除最后一层无任何叶子节点外，每一层上的所有节点都有k个子节点的树，共有?个节点

A.kh+1−1k−1

B.kh−1

C.kh

D.kh−1k−1

答案：A

可以打表可以记结论，甚至你还可以现场推结论：等比数列求和。第0层的节点为1，第1层为k，第二层为k2k2，直到第hh层为khkh，等比数列求和公式为1−kh+11−k1−kh+11−k即可选A。

T5.设某算法的时间复杂度函数的递推方程是T(n)=T(n-1)+n（n为正整数）及T(0)=1，则该算法的时间复杂度为

A.O(log n)

B.O(n log n)

C.O(n)

D.O(n^2)

答案：D

NOIP2015初赛第?题

T6.表达式a∗d−b∗c的前缀形式是

A.ad∗bc∗−

B.−∗ad∗bc

C.a∗d−b∗c

D.−∗∗adbc

答案：B

卧槽慌得一批差点看成后缀表达式。可以直接模拟得到结果。

T7.在一条长度为1的线段上随机取两个点，则以这两个点为端点的线段的期望长度是

A.1/2

B.1/3

C.2/3

D.3/5

答案：B

可以通过几何概型+体积什么什么的求。建立一个三维坐标系Oxyz，x轴代表点A位置，y轴代表点B位置，z轴代表线段长度期望，那么长度的期望就是两个四面体拼起来的图形，顶点为(0,0,0),(1,0,0),(0,1,0),(0,0,1)和(1,1,0),(1,0,0),(0,1,0),(1,1,1)。这个几何体的体积为1/3（锥体体积为底面积乘以高再除以3），由于底面积为1，所以高度平均为1/3，即长度期望为1/3。zcysky的骗分思路也可以参考一下。

T8.关于Catalan数Cn=(2n)!/(n+1)!/n!，下列说法错误的是

A.Cn表示有n+1个节点的不同形态的二叉树的个数。

B.Cn表示含n对括号的合法括号序列的个数。

C.Cn表示长度为n的入栈序列对应的合法的出栈序列个数。

D.Cn表示通过连接顶点而将n+2边的凸多边形分成三角形的方法个数。

答案：A

A选项应该是n而不是n+1。可以把n=1、n=2时候暴力带入。

T9.假设一台抽奖机中有红、蓝两色的球，任意时刻按下抽奖按钮，都会等概率获得红球或蓝球之一。有足够多的人每人都用这台抽奖机抽奖，假如他们的策略均为：抽中蓝球则继续抽球，抽中红球则停止。最后每个人都把自己获得的所有球放到一个大箱子里，最终大箱子里的红球与蓝球的比例接近于

A.1:2

B.2:1

C.1:3

D.1:1

答案：D

考虑如果某个人抽中蓝球，就让他的儿子取抽球，那么每个人抽中红球蓝球的概率相等，那么答案为D。

T10.为了统计一个非负整数的二进制中1的个数，代码如下：

int CountBit(int x){

int ret = 0;

while (x)

{

ret++;

\_\_\_\_\_\_\_\_\_;

}

return ret;

}

则空格内要填入的语句是

A.x>>=1

B.x&=x-1

C.x|=x>>1

D.x<<=1

答案：B

假设二进制末尾是100000的形式，-1之后就是011111，与之后就是000000，最后一个1就没了。这个B选项其实就是lowbit操作。

### **不定项题**

T1.NOIP初赛中，选手可以带入考场的有

A.笔

B.橡皮

C.手机（关机）

D.草稿纸

答案：AB

详见CCF官方规则。一般是不允许带手机和草稿纸，各地规定不太一样呢。

upd：附上[官方规则的地址](http://www.noi.cn/newsview.html?id=619&hash=E7405E&type=6)，本题考得是第12条。

选手进入考场时，只许携带笔、橡皮等非电子文具入场。禁止携带任何电子产品或机器设备入场，无存储功能的手表除外；手机（关机）、U盘或移动硬盘、键盘、鼠标、闹钟、计算器、书籍、草稿纸及背包等物品必须存放在考场外。如有违规带入的，一经发现，NOI各省特派员可直接取消违规选手的参赛资格。

T2.2-3树是一种特殊的树，它满足两个条件：

1. 每个内部节点有两个或三个子节点；
2. 所有的叶子节点到根的路径长度相同。

如果一棵2-3树有10个叶节点，那么它可能有?个非叶节点。

A.5

B.6

C.7

D.8

答案：CD

考虑每一层的节点数是1->2->5->10和1->2->4->10的形式。

T3.下列关于最短路算法的说法正确的有

A.当图中不存在负权回路但是存在负权边时，Dijkstra算法不一定能求出源点到所有点的最短路。

B.当图中不存在负权边时，调动多次Dijkstra算法能求出每对顶点间最短路径。

C.图中存在负权回路时，调用一次Dijkstra算法也一定能求出源点到所有点的最短路。

D.当图中不存在负权边时，调用一次Dijkstra算法不能用于每对顶点间的最短路计算。

答案：ABD

Dijkstra：单源最短路，不支持负权边。当然负权回路也是不行的

T4.下列说法中，是树的性质的有

A.无环

B.任意两个节点之间有且只有一条简单路径

C.有且只有一个简单环

D.边的数目恰好是顶点数目-1

答案：ABD

树的性质常识题。

T5.下列关于图灵奖的说法中，正确的有

A.图灵奖是由电气和电子工程师协会（IEEE）设立的。

B.目前获得该奖项的华人学者只有姚期智教授衽。

C.其名称取自计算机科学的先驱、英国科学家艾伦·麦席森·图灵。

D.它是计算机界最负盛名、最崇高的一个奖项，有“计算机界的诺贝尔奖”之称。

答案：BCD

图灵奖是美国的ACM协会创办的。

### **问题求解**

T1.甲乙丙丁四人在考虑周末要不要外出郊游。

已知

1. 如果周末下雨，并且乙不去，则甲一定不去；
2. 如果乙去，则丁一定去；
3. 如果丙去，则丁一定不去；
4. 如果丁不去，而且甲不去，则丙一定不去。

如果周末丙去了，则甲（去了/没去），乙（去了/没去），丁（去了/没去），周末（下雨、没下雨）。

答案：去了，没去，没去，没下雨

根据3，丙去了，那么丁没去

根据2的逆否，丁没去，乙一定没去

根据4，丙去了，丁没去，那么甲一定去了//upd于12.13

根据1，乙没去，甲去了，周末不下雨

T2.方程a∗b=(a or b)∗(a and b)，在a和b都取[0,31]中的整数时，共有?组解。（\*表示乘法；or表示按位或运算；and表示按位与运算）

答案：454

介绍一个结论：a and b≤min(a,b)≤max(a,b)≤a or b<2∗max(a,b)

最后一个小于号可能不太理解，因为乘以一个2一定要进位，a or b一定是不会进位的。前面的不等号很好理解

那么对答案的贡献形式一定是{a and b=min(a,b)a or b=max(a,b)。那么最小值的二进制一定是最大值的二进制的子集。枚举所有二进制可能的位数，由于要删去a=ba=b的重复情况，那么答案为2×(∑i=05C5i×2i)−32=454

### **阅读程序写结果**

程序自己打的，大括号换了行否则太不顺眼了。。。

#include <cstdio>int main(){

int x;

scanf("%d", &x);

int res = 0;

for (int i = 0; i < x; ++i)

{

if (i \* i % x == 1)

{

++res;

}

}

printf("%d", res);

return 0;

}

输入：15

输出：4

打表找0到14中所有数平方对15取模判断是否为1即可，最后是1,4,11,14是答案。建议验算几遍，以免出错。

#include <cstdio>int n, d[100];bool v[100];int main(){

scanf("%d", &n);

for (int i = 0; i < n; ++i)

{

scanf("%d", d + i);

v[i] = false;

}

int cnt = 0;

for (int i = 0; i < n; ++i)

{

if (!v[i])

{

for (int j = i; !v[j]; j = d[j])

{

v[j] = true;

}

++cnt;

}

}

printf("%d\n", cnt);

return 0;

}

输入：10 7 1 4 3 2 5 9 8 0 6

输出：6

置换的阶的计数。我tm一开始看成11个元素的置换了，把开头的10也算进去了算成了3。这个动手模拟很简单的。

#include <iostream>using namespace std;string s;long long magic(int l, int r){

long long ans = 0;

for (int i = l; i <= r; ++i)

{

ans = ans \* 4 + s[i] - 'a' + 1;

}

return ans;

}int main(){

cin >> s;

int len = s.length();

int ans = 0;

for (int l1 = 0; l1 < len; ++l1)

{

for (int r1 = l1; r1 < len; ++r1)

{

bool bo = true;

for (int l2 = 0; l2 < len; ++l2)

{

for (int r2 = l2; r2 < len; ++r2)

{

if (magic(l1, r1) == magic(l2, r2) && (l1 != l2 || r1 != r2))

{

bo = false;

}

}

}

if (bo)

{

ans += 1;

}

}

}

cout << ans << endl;

return 0;

}

输入：abacaba

输出：16

寻找字符串中独一无二的子串的个数。magic函数其实是[l,r]区间内的一个BKDRHash。什么鬼名字。话说CCF的程序效率太低了吧

#include <cstdio>using namespace std;const int N = 110;bool isUse[N];int n, t;int a[N], b[N];bool isSmall(){

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

if (a[i] != b[i]) return a[i] < b[i];

return false;

}bool getPermutation(int pos){

if (pos > n)

{

return isSmall();

}

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

{

if (!isUse[i])

{

b[pos] = i; isUse[i] = true;

if (getPermutation(pos + 1))

{

return true;

}

isUse[i] = false;

}

}

return false;

}void getNext(){

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

{

isUse[i] = false;

}

getPermutation(1);

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

{

a[i] = b[i];

}

}int main(){

scanf("%d%d", &n, &t);

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

{

scanf("%d", &a[i]);

}

for (int i = 1; i <= t; ++i)

{

getNext();

}

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

{

printf("%d", a[i]);

if (i == n) putchar('\n'); else putchar(' ');

}

return 0;

}

输入1：6 10 1 6 4 5 3 2

输出1：2 1 3 5 6 4

输入2：6 200 1 5 3 4 2 6

输出2：3 2 5 6 1 4

不难看出本题是求排列的下t个排列，CCF这个程序实现的效率太菜了吧，竟然从初始排列不断枚举，直到排列大于给定排列。

对于输入1，可以直接手推

对于输入2，正解应该是康托展开（了解一下），但是也可以一次跳阶乘次步。例如当后3位是单调递减的时候，可以直接跳3的阶乘次步，直接转移到下一个后3位是单调递减的情况。能节省中间模拟的时间，这种方法对不会康托展开的同学（比如我）适用

### **完善程序**

T1.对于一个1到n的排列P（即1到n中每一个数在P中出现了恰好一次），令qi为第i个位置之后第一个比Pi值更大的位置，如果不存在这样的位置，则qi=n+1。举例来说，如果n=5且P为1 5 4 2 3，则q为2 6 6 5 6。

下列程序读入了排列P，使用双向链表求解了答案。试补全程序。（第二空2分，其余3分）

数据范围1≤n≤105。

#include <iostream>using namespace std;const int N = 100010;int n;int L[N], R[N], a[N];int main(){

cin >> n;

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

{

int x;

cin >> x;

\_\_\_\_(1)\_\_\_\_;//a[x] = i

}

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

{

R[i] = \_\_\_\_(2)\_\_\_\_;//i + 1

L[i] = i - 1;

}

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

{

L[\_\_\_\_(3)\_\_\_\_] = L[a[i]];//R[a[i]]

R[L[a[i]]] = R[\_\_\_\_(4)\_\_\_\_];//a[i]

}

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

{

cout << \_\_\_\_(5)\_\_\_\_ << " ";//R[i]

}

cout << endl;

return 0;

}

1.a[i]表示大小为i的元素所在位置。

2.对称地写下一行的东西。链表的下一个元素。

3.对称地写下一行的东西。删除链表中本元素，把本元素右边元素的左指针跨过本元素指向左侧元素。

4.还是对称。把左边元素右指针指向右侧元素。

5.输出第i个数的后继元素即第一个比他大的元素，即R[i]。

T2.一只小猪要买N件物品（N不超过1000）。

它要买的所有商品在两家商店里都有卖。第i件物品在第一家商店的价格是a[i]，在第二家商店的价格是b[i]，两个价格都不小于0且不超过10000。如果在第一家商店买的物品的总额不少于50000，那么在第一家店买的物品都可以打95折（价格变为原来的0.95倍）。

求小猪买齐所有物品所需最小的总额。

输入：第一行一个数N。接下来N行，每行两个数。第i行的两个数分别代表a[i]，b[i]。

输出：输出一行一个数，表示最少需要的总额，保留两位小数。

试补全程序。（第一空2分，其余3分）

#include <cstdio>#include <algorithm>using namespace std;

const int Inf = 1000000000;const int threshold = 50000;const int maxn = 1000;

int n, a[maxn], b[maxn];bool put\_a[maxn];int total\_a, total\_b;double ans;int f[threshold];

int main(){

scanf("%d", &n);

total\_a = total\_b = 0;

for (int i = 0; i < n; ++i)

{

scanf("%d%d", a + i, b + i);

if (a[i] <= b[i]) total\_a += a[i];

else total\_b += b[i];

}

ans = total\_a + total\_b;

total\_a = total\_b = 0;

for (int i = 0; i < n; ++i)

{

if (\_\_\_\_(1)\_\_\_\_)//0.95 \* a[i] <= b[i]

{

put\_a[i] = true;

total\_a += a[i];

}

else

{

put\_a[i] = false;

total\_b += b[i];

}

}

if (\_\_\_\_(2)\_\_\_\_)//total\_a >= threshold

{

printf("%.2f", total\_a \* 0.95 + total\_b);

return 0;

}

f[0] = 0;

for (int i = 1; i < threshold; ++i)

f[i] = Inf;

int total\_b\_prefix = 0;

for (int i = 0; i < n; ++i)

{

if (!put\_a[i])

{

total\_b\_prefix += b[i];

for (int j = threshold - 1; j >= 0; --j)

{

if (\_\_\_\_(3)\_\_\_\_ >= threshold && f[j] != Inf)//total\_a + j + a[i]

ans = min(ans, (total\_a + j + a[i]) \* 0.95 + \_\_\_\_(4)\_\_\_\_);//f[j] + total\_b - total\_b\_prefix

f[j] = min(f[j] + b[i], j >= a[i] ? \_\_\_\_(5)\_\_\_\_ : Inf);//f[j - a[i]]

}

}

}

printf("%.2f", ans);

return 0;

}

一个奇怪的DP。

1.如果是a[i]<=b[i]，那么要上面的统计有啥用。这里的循环其实是在假定需要打折的情况下，先挑选已经钦定的a[i]的物品。

2.判断如果打折，直接购买便宜的a是否已经ok了。如果是就不需要再舍弃b购买一个较贵的a了，直接输出即可。

3.注意到下面是total\_a + j + a[i]乘的0.95，那么这个一定是a的总价，那么它一定要大于等于50000。

4.观察下面的min，f[j]+b[i]说明f数组记录的是和b的价格有关的，而且它必须要出现，那么就放到这里。注意到total\_b\_prefix和total\_b没有用过，说明他俩必须要用上，由于prefix是前缀，所以让total\_b减去total\_b\_prefix即可。

5.由于j>=a[i]，那么j-a[i]>=0，考虑我们的背包转移，可以大胆猜想这里填f[j-a[i]]。

本题实际上的状态是f[j]代表在前i个物品中，购买a店物品j元情况下购买b店物品的最小价值。total\_a+j+a[i]代表买所有钦定要买的+额外需要买的+这次买的，f[j]+total\_b-total\_b\_prefix是指i到n中所有选b，前面按照f[j]的状态取，b的价格。最后f[j]和取这次的b[i]与从j-a[i]转移过来去一个min，表示j这个状态取b还是取a。