HGNU 11.3结训赛题解

大数据2301周康禧

过题数统计：

A:1/137

B:12/124

C:0/7

D:1/73

E:0/18

F:8/39

G:0/48

H:29/131

I:23/65

J:60/65

K:16/95

L:25/82

本来的预测难度：
Hard: E C G
Mid: D J F K
Easy: B A I L H

实际难度B A偏难 K偏简单

B和A是两个学长出的本来学长的定位是签到题，但是后来由于验题人觉得A题数据太弱，导致数据被加强

H.直觉引导,分析推进

- 当 $n = 2$时：$5^2 \equiv 25\ mod\ 100$
- 假设： $5^k \equiv 25\ mod\ 100$
- $5^{k+1} = 5^k \times 5$
- $5^{k+1} \equiv 25 \times 5\ mod\ 100$
- $5^{k+1} \equiv 125\ mod\ 100$
- $5^{k+1} \equiv 25\ mod\ 100$
  通过数学归纳法，我们证明了对于所有 $n \geq 2$都有：
- $5^n \equiv 25\ mod\ 100$

题目也已经给提示了

输出 5^n 的最后两位数字(十位和个位)

输入格式

输入的唯一一行包含一个整数 n ($2 \leq n \leq 10^9$)-你需要计算的数字5的幂。

Tip: **不要试图枚举，因为会超时，请大胆猜测ヽ(o^▽^o)ノ**

```c
#include<stdio.h>
int main()
{
    printf("25");
    return 0;
}
```

L. 但偏偏,雨渐渐,大到我看你不见

- 定义一个字符串数组
- 直接判断第x-1个即可
- 因为数组下表从一开始

```c
#include<stdio.h>
int main()
{
    char S[100];
    int N,x;
    scanf("%d %d",&N,&x);
    scanf("%s",S);
    if(S[x-1]=='o'){
        printf("YES");
    }
    else{
        printf("NO");
    }
    return 0;
}
```

I. 防AK题？还是签到题？

- 在二维平面上，我们可以向八个方向移动：上、下、左、右以及四个对角方向（左上、右上、左下、右下）。每次移动可以改变 x 或 y 坐标，或者同时改变两者（即对角移动）
- 同时移动的优势：每次对角移动可以同时减少x和y各 1 个单位
- 如果我们能够在每一步中都选择最佳方向（即对角方向），则可以更快地接近目标点。
- 先走对角线，然后走长边多出来的部分即可
- 答案就是$max(|x_2 - x_1|, |y_2 - y_1|)$

代码

```c
#include<stdio.h>
#include<math.h>
int main()
{
    int a,b,c,d,d1,d2,s,t;
    scanf("%d %d\n%d %d",&a,&b,&c,&d);
     d1=abs(c-a);
     d2=abs(d-b);
     s=(d1<d2)?(d1):(d2);
     if(s==d1){
         t=d2-d1;
     }
     else{
         t=d1-d2;
     }
     printf("%d",t+s);
}
```

A. 和谐之元

- 题意为求出$lcm\{a,b,c,d,e,f\}$
- lcm为最小公倍数
- 1. 直接暴力for循环枚举
- 2. 手写gcd函数再写手写lcm函数直接计算
- 注意由于数据加强for循环从1开始每次$+1$, 如果数据很大会TLE(超时)
- 所以考虑加快循环遍历速度, 每次增加$max\{a,b,c,d,e,f\}$即可

```
#include<stdio.h>
int main()
{
long long int a,b,c,d,e,f;
scanf("%lld%lld%lld%lld%lld%lld",&a,&b,&c,&d,&e,&f);
long long int arr[6];
arr[0]=a;
arr[1]=b;
arr[2]=c;
arr[3]=d;
arr[4]=e;
arr[5]=f;

for(int i=0;i<5;i++)
{
    for(int j=i+1;j<6;j++)
    {
        if(arr[i]>arr[j])
        {
            long long int t=arr[i];
            arr[i]=arr[j];
            arr[j]=t;
        }
    }
}
for(long long int i=arr[5];;i=i+arr[5])
{
     if(i%a==0&&i%b==0&&i%c==0&&i%d==0&&i%e==0&&i%f==0)
    {
        printf("%lld",i);
        break;
    }
}

    return 0;
}
```

A. 和谐之元

- 题意为求出$lcm\{a,b,c,d,e,f\}$
- lcm为最小公倍数
- 1. 直接暴力for循环枚举
- 2. 手写gcd函数再写手写lcm函数直接计算
- 注意由于数据加强for循环从1开始每次$+1$, 如果数据很大会TLE(超时)
- 所以考虑加快循环遍历速度, 每次增加$max\{a,b,c,d,e,f\}$即可

```cpp
#include <bits/stdc++.h>
const int N=2e5+10;
const int inf = 0x3f3f3f3f;
using namespace std;
using ull = unsigned long long int;
using ll = long long int;
ll gcd(ll x,ll y){
    return y?gcd(y,x%y):x;
}
ll lcm(ll x,ll y){
    return y/gcd(x,y)*x;
}
int main() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    ll a[6];
    ll ans=1;
    for(int i=0;i<6;i++)
    {
        scanf("%lld",&a[i]);
        ans=lcm(a[i],ans);
    }
    cout<<ans<<'\n';
    return 0;
}
```

B. $Flysky$逃离博物馆

- 从n层到1层会有$n-1$次选择房间
  每次有两个选择，即会有2^{n-1}种情况而1层只有一个出口的概率就是$\frac{1}{2^{n-1}}$
- 按题意for循环计算取模输出即可

代码

```
#include <stdio.h>
int main()
{
    long long int n,a;a=1;
    scanf("%lld",&n);
    for(int i=1;i<=n-1;i++)
    {
        a=a*2;
        a=a%1000000007;
    }

    printf("%d/%lld",1,a);
    return 0;

}
```

K. 不要你离开

- 记录序列N的最小最大值
- 直接for循环遍历判断哪一个没有出现过即可，需要读入使用另外一个数组记录每一个的出现次数
- 或者排序判断哪一个缺少了也可以
- 用最小最大等差求和减去序列N的Sum也可以

```c
#include<stdio.h>
int main()
{
    int n,a[10000],b[10000],max=0,min=20000
    scanf("%d",&n);
    for(int i=1;i<=n;i++){
        scanf("%d",&a[i]);
        b[a[i]]++;
        if(a[i]>max){
            max=a[i];
        }
        if(a[i]<min){
            min=a[i];
        }
    }
    for(int j=min;j<=max;j++){
        if(b[j]==0){
            printf("%d",j);
        }
    }
}
```

K. 不要你离开

- 记录序列N的最小最大值
- 直接for循环遍历判断哪一个没有出现过即可，需要读入使用另外一个数组记录每一个的出现次数
- 或者排序判断哪一个缺少了也可以
- 用最小最大等差求和减去序列N的Sum也可以

```c
#include<stdio.h>

int main(){
int n;
scanf("%d",&n);
int a[10000],b[10000]={0};
for(int i=1;i<=n;i++)
scanf("%d",&a[i]);
for(int i=1;i<=n;i++)
    b[a[i]]++;

 for(int i=1;i<=n-1;i++)
    for(int j=1;j<=n-i;j++)
    {
        if(a[j]>a[j+1]){
            int  t;
            t=a[j];
            a[j]=a[j+1];
            a[j+1]=t;
        }
    }

    for(int i=a[1];i<=a[n];i++)
    if(b[i]==0)
    printf("%d",i);
return 0;
}
```

K. 不要你离开

- 记录序列N的最小最大值
- 直接for循环遍历判断哪一个没有出现过即可，需要读入使用另外一个数组记录每一个的出现次数
- 或者排序判断哪一个缺少了也可以
- 用最小最大等差求和减去序列N的Sum也可以

```c
#include<stdio.h>
int main()
{
    int n,a[10000],max=0,min=20000;
    int sum=0;
    scanf("%d",&n);
    for(int i=1;i<=n;i++){
        scanf("%d",&a[i]);
        sum+=a[i];
        if(a[i]>max){
            max=a[i];
        }
        if(a[i]<min){
            min=a[i];
        }
    }
    printf("%d",(max+min)*(max-min+1)/2-sum);
}
```

F. 螺旋塔

- 按照题意模拟即可
- 每一次把第一个字符移动到最后一个，其余的依次前移
- 主要考察字符串数组
- 这个题目是被6哥临时换上来的
- 原来的这个位置的题目因为被认为太难被换掉了，觉得这个会简单很多，都是结果也不尽人意，还是难一点了

```c
#include<stdio.h>
int main(){
    int n,k;
    scanf("%d %d",&n,&k);
    char c[n];
    scanf("%s",c);
    printf("%s\n",c);
    for(int i=0;i<k-1;i++){
       int t=c[0];
        for(int j=0;j<n;j++){
            c[j]=c[j+1];
        }
        c[n-1]=t;
        printf("%s\n",c);
    }
    return 0;
}
```

J. 何小猪就要吃零食

牛客上面的原题：链接货仓选址

RandolphJ 发表于 2019-12-08 13:24:15

MarkDown视图 Copy 翻译

题目链接

假设把货仓建在第k个商店的坐标上，那么左边有 k - 1 个商店，右边有 n - k - 1 个商店。

当k<n/2时，向右移一位，因为k-1<n-k-1，所以+(k-1)-(n-k-1)会使总距离减小，因此我们应当把k往右移，直到当 k-1 = n-k-1时，若再往右移，k-1>n-k-1，+(k-1)-(n-k-1)会使总距离变大，所以不能往右移了。 若k-1 = n-k-1, 那么k=n/2, k也就是整个序列中位数的位置。

如果商店数为偶数的话，中点有两个商店，此时在这两个商店之间任选一点都可，即a[n/2] <=k<=a[n/2+1]，因为除这两个商店外左右两边的商店数是相等的（即k向右移总距离加减数相等），而这两个商店到k的距离和又是定值，所以不影响答案，我们可以直接选择a[n/2]。

```
#include<algorithm>
#include<cstdio>
using namespace std;
int n,ans,mid,a[100005];
int main() {
    scanf("%d",&n);
    for (int i=1; i<=n; i++) scanf("%d",&a[i]);
    sort(a+1,a+n+1),mid=a[n/2];
    for (int i=1; i<=n; i++) ans+=abs(a[i]-mid);
    printf("%d",ans);
}
```

收起

应该很好想到，排序后选中间的就可以了

```c
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int main(void){
    int n;
    scanf("%d",&n);
    int num[n];
    int s=0;
    for(int i=0;i<n;i++){
        scanf("%d",&num[i]);
        s=s+num[i];
    }

    for(int i=0;i<n;i++){
    for(int j=i+1;j<n;j++){//枚举后面比我小的
        if(num[j]<num[i]){
            int t=num[j];//交换位置
            num[j]=num[i];
            num[i]=t;
        }
    }
    }
    int x=num[n/2];
    int sum=0;
    for(int i=0;i<n;i++){
        sum=sum+abs(x-num[i]);
    }
    printf("%d",sum);
    return 0;

}
```

D. 简单数学题（middle version）

- 很可惜这题本场比赛只有一个人完全算对了

- 这题只是在初选赛的$easy$版本上加了一个条件而言，本来觉得应该可以过好几个人的，结果都没有完全算对
- 这题就是在原来easy版本上简单改一下需要计算C_{m+n-1}^{n-1}而不是原来$easy$版本的C_{m-1}^{n-1}，因为需要每一个x不为零，就相当是先让每一个x都先分1然后再按每一个x不会为0计算即可，所以比$easy$的版本要多加一个n，为高中简单组合数学

代码

```
#include<stdio.h>
int main()
{
    unsigned long long n,m,a=1,b=1,c=1;
    unsigned long long  s;
    scanf("%llu %llu",&n,&m);
    for(int i=1;i<=m+n-1;i++){
        a=a*i;
    }
    for(int j=1;j<=n-1;j++){
        b=b*j;
    }
    for(int v=1;v<=m;v++){
        c=c*v;
    }
    s=a/(b*c);
    printf("%llu",s);
}
```

G. 有手就行

- 如果每一个T都要两个for循环计算再求和的话就只能有45分，其他的都TLE（超时）了

解释：

$$\sum_{i=1}^{5}\left\lfloor\frac{5}{i}\right\rfloor=\left\lfloor\frac{5}{1}\right\rfloor+\left\lfloor\frac{5}{2}\right\rfloor+\left\lfloor\frac{5}{3}\right\rfloor+\left\lfloor\frac{5}{4}\right\rfloor+\left\lfloor\frac{5}{5}\right\rfloor=5+3+2+2+1=13$$

$$\sum_{i=1}^{5}\left\lfloor\frac{5}{i}\right\rfloor=\left\lfloor\frac{5}{1}\right\rfloor+\left\lfloor\frac{5}{2}\right\rfloor+\left\lfloor\frac{5}{3}\right\rfloor+\left\lfloor\frac{5}{4}\right\rfloor+\left\lfloor\frac{5}{5}\right\rfloor=5+2+1+1+1=10$$

Tip:如果你超出时间限制的话，试着在T次询问之前就将所有可能取值的$f(x),g(x)$算出来然后在T的循环内部直接输出，这样试一下？(◠‿◠)

- 但是我已经给出提示了，可惜没人能体会理解，因为题目a,b的范围最多1000，你只需要一次性把所有的$f(x)$和$g(x)$计算完然后全部存在数组中，然后T的循环直接输出就可以了

× Time Exceeded

```
#include<stdio.h>
int main()
{
    int n,cnt=1,a,b,i=1,sum=0,quan=0;
    int w,q;
    scanf("%d",&n);
    for(cnt=1;cnt<=n;cnt++){
        scanf("%d %d",&a,&b);
        sum=0,quan=0;
        for(i=1;i<=a;i++){
          w=a/i;
          if(a%i!=0){
            w=w+1;
          }
          sum+=w;
        }
        for(i=1;i<=b;i++){
            q=b/i;
            quan+=q;
        }
        printf("%d %d\n",sum,quan);

    }
    return 0;
}
```

G. 有手就行

- 如果每一个T都要两个for循环计算再求和的话就只能有45分，其他的都TLE（超时）了

解释：

$$\sum_{i=1}^{5}\left\lfloor\frac{5}{i}\right\rfloor=\left\lfloor\frac{5}{1}\right\rfloor+\left\lfloor\frac{5}{2}\right\rfloor+\left\lfloor\frac{5}{3}\right\rfloor+\left\lfloor\frac{5}{4}\right\rfloor+\left\lfloor\frac{5}{5}\right\rfloor=5+3+2+2+1=13$$

$$\sum_{i=1}^{5}\left\lfloor\frac{5}{i}\right\rfloor=\left\lfloor\frac{5}{1}\right\rfloor+\left\lfloor\frac{5}{2}\right\rfloor+\left\lfloor\frac{5}{3}\right\rfloor+\left\lfloor\frac{5}{4}\right\rfloor+\left\lfloor\frac{5}{5}\right\rfloor=5+2+1+1+1=10$$

Tip:如果你超出时间限制的话，试着在T次询问之前就将所有可能取值的$f(x),g(x)$算出来然后在T的循环内部直接输出，这样试一下？(◠‿◠)

- 但是我已经给出提示了，可惜没人能体会理解，因为题目a,b的范围最多1000，你只需要一次性把所有的$f(x)$和$g(x)$计算完然后全部存在数组中，然后T的循环直接输出就可以了

```cpp
#include <bits/stdc++.h>
const int N=2e5+10;
const int inf = 0x3f3f3f3f;
using namespace std;
using ull = unsigned long long int;
using ll = long long int;

int main() {
    ios::sync_with_stdio(false);
    cin.tie(0);
    vector<int>f(N,0),g(N,0);
    for(int i=1;i<=1e3;i++){
        for(int j=1;j<=i;j++)
        {
            f[i]+=(i+j-1)/j;
            g[i]+=i/j;
        }
    }
    int _;
    cin>>_;
    while (_--)
    {
        int a,b;
        cin>>a>>b;
        cout<<f[a]<<" "<<g[b]<<'\n';
    }
    return 0;
}
```

C. 这不易如反掌吗？

- 根据题目我们需要对32768取模变成零，可以每次乘以2或者加1，那么考虑每一个数字二进制位的最低位的1位是第k位，然后一直选择操作2，那么最快就是$16-k-1$次，但是观察样例发现有的情况可以通过+1可能让二进制最低位的1的位置前移动，
- 那么就直接枚举加0~15次1然后一直取最小的情况就可以了，因为如果1的次数大于等于16，就一定可以一直操作2然后至少15次完成，不会一直加1超过15次
- 然后需要有一些二进制的位运算的基础知识，乘以2就是二进制左移1
  反之右移就是除以二(向下取整)
- $x *= 2$ ⟹ $x <<= 1$
- ((t>>k)&1)就是枚举t的二进制第k位的值

```
#include <stdio.h>
int main() {
    int n;
    scanf("%d",&n);
    int a[n+1];
    for(int i=1;i<=n;i++){
        scanf("%d",&a[i]);
        int ans=16;
        for(int j=0;j<=15;j++){
            int t=a[i]+j;
            for(int k=0;k<=15;k++){
                if((t>>k)&1){
                    if(ans>16-k-1+j){
                        ans=16-k-1+j;
                    }
                    break;
                }
            }
        }
        if(a[i]==0)ans=0;
        printf("%d ",ans);
    }
    return 0;
}
```

E. 简单数学题（hard version）

- Hard版本就还是算C_{m+n-1}^{n-1}但是需要取模计算
- 如果你使用for循环算出每一部分的阶乘然后再除的话就错完了，因为除法取模需要用逆元计算(可以自己了解一下)
- 所以这个时候就需要使用二维数组来计算
- 使用那个杨辉三角的递推公式相加就可以然后直接取模就好了
- 这个题目一样要全部算好然后存数组，之后直接输出即可

```
#include <stdio.h>
const int N = 4010;
const int mod = 998244353;
int c[4010][4010];
int main(){
    for(int i = 0; i <=4000; i ++){
        c[i][0]=1;
        c[i][i]=1;
        for(int j = 0; j <i; j ++){
            c[i][j] = (c[i - 1][j - 1] + c[i - 1][j]) % mod;
        }
    }
    int _;
    scanf("%d",&_);
    while(_--){
        int a,b;
        scanf("%d%d",&a,&b);
        printf("%d\n",c[a + b - 1][a - 1]);
    }
    return 0;
}
```