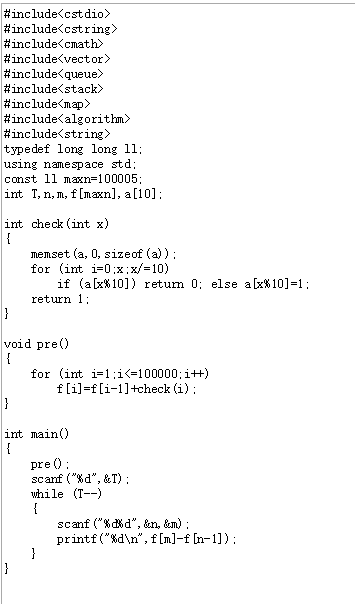
**训练赛**

A:先预处理出100000以内是beautiful number个数的前缀和，然后直接输出。



B：因为不能拼接，所以直接在最大那块的面积与0之间二分求即可

#include <stdio.h>

#include <string.h>

#include <math.h>

#include <algorithm>

using namespace std;

const double pi = acos(-1.0);

double a[10005];

int n,m;

int check(double mid)

{

int i;

int sum = 0;

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

{

sum+=int(a[i]/mid);

if(sum>=m)

return 1;

}

return 0;

}

int cmp(double a,double b)

{

return a>b;

}

int main()

{

int t,i,j;

double l,r,mid;

scanf("%d",&t);

while(t--)

{

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

m++;

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

{

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

a[i] = a[i]\*a[i]\*pi;

}

sort(a,a+n,cmp);

l = 0;

r = a[0];

if(m<n)

n = m;

while(r-l>1e-5)

{

mid = (r+l)/2;

if(check(mid))

l = mid;

else

r = mid;

}

printf("%.4lf\n",l);

}

return 0;

}

C:尺取

D:割绳子，二分

E：暴力枚举



F:二进制枚举（得病处为1，否则为0，比如得了2 1两种病，代号就是011（十进制就是3）），首先枚举出1的个数等于k的二进制数，然后跟所有的牛的代号一一比较，符合的   +1，找出其中和最大的；

#include<stdio.h>

#include<string.h>

#include<stack>

#include<string>

#include<math.h>

#include<queue>

#include<set>

#include<algorithm>

#include<iostream>

#include<vector>

#include<map>

using namespace std;

#define LL \_\_int64

#define inf 1999999999

#define N 1010

#define mod 1000000007

/\*状态压缩就是把所有状态用2进制的01表示，0表示没病，1表示有病

要枚举的所有病的状态就要把每头牛得病的状态表示出来

第一位从右看起，比如牛有第一种病就是000001，也就是1

在比如牛有第一种和第二种病，也就是0000011，也就是3

这样就把每头牛得病的状态表示出来了

一共就15种病，所以用15位就可以表示出所有得病的状态，比如得15种病的

状态就是11111111111111用10进制表示的数就是2^15\*/

int dp[N];

bool judge(int x,int k)

{

int res=0;

while(x)

{

res+=(x&1);

x>>=1;

}

return res<=k;

}

int main()

{

int i,j,m,n,k,num,x;

while(scanf("%d%d%d",&n,&m,&k)!=-1)

{

memset(dp,0,sizeof(dp));//刚开始所有的牛都没有病

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

{

scanf("%d",&num);

while(num--)

{

scanf("%d",&x);//第i个牛得第x种病;

dp[i]|=1<<(x-1);//加进去

}

}

int ans=0;

for(i=0;i<(1<<m);i++)

{

if(judge(i,k)==0)//判断得病的种类不能超过k

continue;

int tmp=0;//在这种状态下牛得数量

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

{

if((dp[j]&i)==dp[j])

tmp++;

}

ans=max(ans,tmp);

}

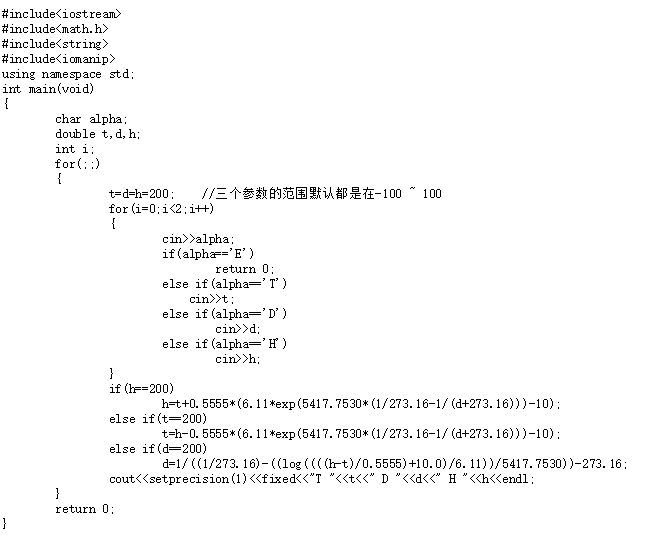
printf("%d\n",ans);

}

return 0;

}

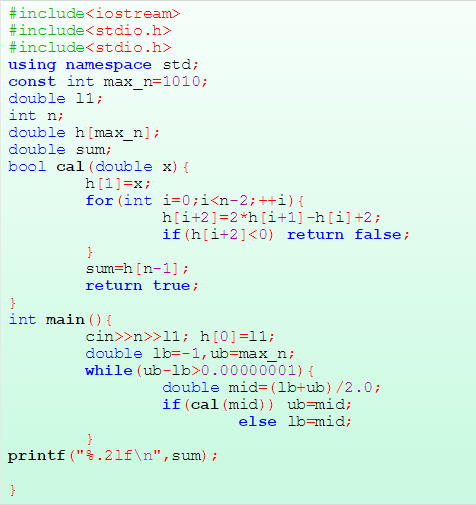
G：



H：N个灯泡离地H\_i，满足H1 = A ，Hi = (Hi-1 + Hi+1)/2 – 1，HN = B ，求最大B。

关键是选取什么值作为mid，已知A，如果知道第二个高度h2的话，就可以推出h3，然后又可以利用h2和h3推出h4。所以h2才是关键，设mid为h2，二分条件为mid可以使得所有灯泡高度大于零。

从严密性的角度讲，h2的大小是跟B正相关的，满足二分单调性的要求。



I：**问题：**

**有**N个0到9的数字，互不重复。用这些数字组成2个整数，每个数字只能用一次，且必须用尽所有数字，且0不能在第一位（除非组成的整数是0）。**求**组成整数的差的最小值。

**思路:**

可以将所有情况分为两类：

（1）N是偶数：可以将这N个数字平分，使每个整数含N/2个数字。要使差最小，只需使两个整数最高有效位的差最小，然后使用贪心算法使次高有效位及以后的位数的差最大。这里使两个整数最高有效位的差最小的做法可能有多种，因此要枚举每种做法，取出最后结果值最小的那一个。

（2）N是奇数：将N个数字中除0外的最小数字分配给其中一个整数，作为最高有效位。然后将剩余的数字平分，从高位到低位贪心取差最大的两个数字。

#include<iostream>

#include<string>

#include<vector>

using namespace std;

int main(){

int N;

string temp;

cin >> N;

getline(cin, temp);

while(N--){

vector<int> digit;

getline(cin, temp);

for(int i = 0; i < temp.size(); i += 2){

digit.push\_back(temp[i] - '0');

}

if(digit.size() == 2){

cout << digit[1] - digit[0] << endl;

}else if(digit.size() % 2 == 0){

int min\_inter = 10000;

for(int i = 0; i + 1 < digit.size(); i++){

if(digit[i] != 0 && digit[i + 1] - digit[i] < min\_inter)

min\_inter = digit[i + 1] - digit[i];

}

int MIN = 10000;

for(int i = 0; i + 1 < digit.size(); i++){

if(digit[i] != 0 && digit[i + 1] - digit[i] == min\_inter){

vector<int> left;

vector<int> right;

left.push\_back(digit[i]);

right.push\_back(digit[i+1]);

for(int j = 0, k = digit.size() - 1; j < k; j++, k--){

if(j == i || j == i + 1){

k++;

continue;

}

if(k == i || k == i + 1){

j--;

continue;

}

left.push\_back(digit[k]);

right.push\_back(digit[j]);

}

int a = 0, b = 0;

for(int j = 0; j < left.size(); j++){

a = (10 \* a + left[j]);

}

for(int j = 0; j < right.size(); j++){

b = (10 \* b + right[j]);

}

if(MIN > b - a)

MIN = b - a;

}

}

cout << MIN << endl;

}else{

vector<int> left;

vector<int> right;

int i;

if(digit[0] == 0){

i = 1;

right.push\_back(digit[1]);

}else{

i = 0;

right.push\_back(digit[0]);

}

for(int j = 0, k = digit.size() - 1; j < k; j++, k--){

if(j == i){

k++;

continue;

}

if(k == i){

j--;

continue;

}

left.push\_back(digit[k]);

right.push\_back(digit[j]);

}

int a = 0, b = 0;

for(int j = 0; j < left.size(); j++){

a = (10 \* a + left[j]);

}

for(int j = 0; j < right.size(); j++){

b = (10 \* b + right[j]);

}

cout << b - a << endl;

}

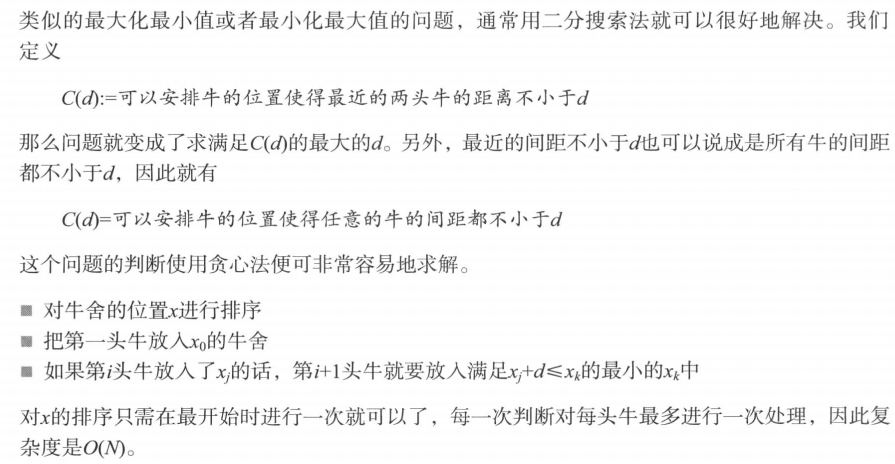
}

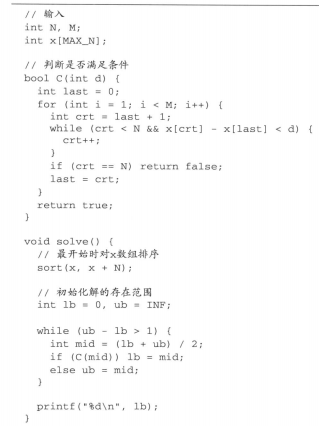
}

**周赛**

1001：二分

自行把牛理解为cxy花





1002：前缀和

对于每个[Li,Ri]，a[Li]++,a[Ri+1]--，再求a数组的前缀和，后面就好处理了。

#include <iostream>

#include <cstdio>

#include <string>

#include <cstring>

#include <cmath>

using namespace std;

const int maxn=1000000+5;

const int maxm =1000000+5;

int a[maxn],b[maxn],sum[maxn];

bool f[maxn];

int main()

{

int n,m;

//freopen("2.in","r",stdin);

//freopen("2.out","w",stdout);

while(scanf("%d%d",&n,&m)!=EOF)

{

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

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

memset(sum,0,sizeof sum);

memset(b,0,sizeof b);

while(m--)

{

int l,r;

scanf("%d%d",&l,&r);

b[l]++;

b[r+1]--;

}

memset(f,0,sizeof f);

int Q=0;

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

{

sum[i]=sum[i-1]+b[i];

if (sum[i]>=a[i])

{

f[i]=true;

Q++;

}

}

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

bool flag=true;

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

{

if (f[i])

{

if (flag)

{

flag=false;

printf("%d",i);

}

else printf(" %d",i);

}

}

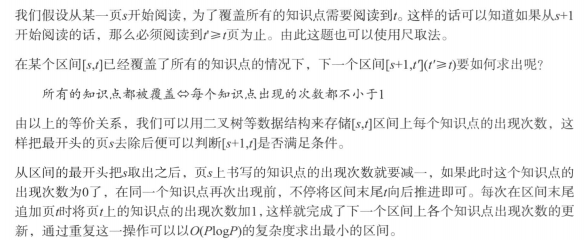
printf("\n");

}

return 0;

}

1003：尺取



#include<cstdio>

#include<map>

#include<set>

#include<algorithm>

#include <iostream>

using namespace std;

const int MAXN=1000000+10;

const int INF=0x3fffffff;

int a[MAXN];

int main()

{

//freopen("4.in","r",stdin);

//freopen("4.out","w",stdout);

int p;

while(~scanf("%d",&p))

{

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

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

set<int> S;

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

S.insert(a[i]);

int n=S.size();

map<int,int> x;

int s=0,t=0,ans=p,cnt=0;

while(true)

{

while(cnt<n && t<p)

{

if(x[ a[t++] ]++==0)

cnt++;

}

if(cnt<n) break;

ans=min(ans,t-s);

if(--x[a[s++]]==0)

cnt--;

}

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

}

return 0;

}

1004:枚举啊啊啊

#include <iostream>

#include <cstdio>

#include <cstring>

#include <string>

using namespace std;

int main()

{

//freopen("3.in","r",stdin);

//freopen("3.out","w",stdout);

long long ans;

int m,x,y,z;

while(scanf("%d%d%d%d",&m,&x,&y,&z)!=EOF)

{

ans=0;

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

for (int j=0;j<=y;j++)

{

if (((m-5\*j-25\*i)<=x)&&((m-5\*j-25\*i)>=0))

ans++;

}

printf("%I64d\n",ans);

}

return 0;

}

1005：二分

#include<cstdio>

#include<cstring>

#include<algorithm>

using namespace std;

int n, m, w;

int a[100010], b[100010];

bool ok(int x){

memset(b, 0, sizeof(b));

int k = m;

int c = 0;

int p;

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

c += b[i];

if(a[i]+c<x){

p = x-a[i]-c;

if(p>k) return 0;//天数不足，不可行

k-=p;

c+=p;

b[(i+w)>=n?n:(i+w)]-=p;

}

}

return 1;

}

int main(){

// freopen("5.in","r",stdin);

//freopen("5.out","w",stdout);

while(~scanf("%d %d %d", &n, &m, &w)){

int low, top;

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

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

if(i){

low = min(low, a[i]);

top = max(top, a[i]);

}

else{

low=top=a[i];

}

}

top+=m;

int ans = low;

while(low<=top){

int mid = (low+top)>>1;

if(ok(mid)){

ans = max(ans, mid);

low = mid+1;

}

else{

top = mid-1;

}

}

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

}

return 0;

}

1006：贪心

#include<iostream>

#include<vector>

#include<algorithm>

#include<functional>

#include<iterator>

#include<iterator>

#include<cstdio>

#include<queue>

using namespace std;

int main()

{

//freopen("7.in", "r", stdin);

//freopen("7.out", "w", stdout);

priority\_queue<int,vector<int>,greater<int> > pq;

int n(0);

while(scanf("%d",&n)!=EOF)

{

while(!pq.empty())

pq.pop();

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

{

int b;

scanf("%d",&b);

pq.push(b);

}

long long sum(0);

int l1(0), l2(0);

while (pq.size() > 1)

{

l1= pq.top();

pq.pop();

l2= pq.top();

pq.pop();

sum = sum+l1 + l2;

pq.push(l1 + l2);

}

printf("%I64d\n",sum);

}

return 0;

}

1007:尺取

#include<stdlib.h>

#include<stdio.h>

#include<math.h>

bool isPrime(int num){

int m;

if(num==2) return true;

//注意，搜索是否可以被整除的范围只在小于根号num内即可

//不会有比根号num更大的约数，另外，一旦发现可以被整除

//立即跳出循环，这样可以大幅度减少时间消耗，要不然会超时

for(m=2;m<(int)sqrt((double)num);m++){

if(num%m==0)

break;

}

if(num%m==0)

return false;

else

return true;

}

int main(){

freopen("6.in","r",stdin);

freopen("6.out","w",stdout);

int prime[1229],i,j=0,len=1229,in,out,sum;

//计算10000以内的所有素数

for(i=2;i<10000;i++){

if(isPrime(i))

prime[j++]=i;

}

while(~scanf("%d",&in)){

out=0;

for(i=0;i<len;i++){

sum=0;

for(j=i;j<len;j++){

sum+=prime[j];

if(sum>in)

break;

else if(sum==in){

out++;

break;

}

}

}

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

}

return 0;

}