# noip 2012 提高组 解题报告

# Vigenère密码

模拟

注意大小写

var

i,l,t:longint;

a:Array[0..1000] of longint;

s:string;

ch:char;

begin

readln(s);

l:=length(s);

for i:=1 to l do

if ord(s[i])>=97 then a[i]:=ord(s[i])-96

else a[i]:=ord(s[i])-64;

a[0]:=a[l];

read(ch);

i:=1;

t:=ord(ch);

while (t>64)and(t<124) do

begin

if t<97 then

begin

t:=t-64;

t:=(t-a[i mod l]+27) mod 26;

if t=0 then t:=26;

t:=t+64;

write(chr(t));

end

else

begin

t:=t-96;

t:=(t-a[i mod l]+27) mod 26;

if t=0 then t:=26;

t:=t+96;

write(chr(t));

end;

read(ch);

inc(i);

t:=ord(ch);

end;

end.

# 国王游戏

快排+高精乘+高精除

可以证明最优序列是按照左手上的数\*右手上的数递增排序，所以

1. 按照左手上的数\*右手上的数递增排序
2. 前n-1个左手上的数累乘（高精乘）
3. 除以n右手上的数（高精除）

var

i,n,l:longint;

a,b,c:array[0..10000] of longint;

g,g2:array[0..100000] of longint;

procedure gj1;

var

j:longint;

begin

for j:=1 to l do g[j]:=g[j]\*b[i];

for j:=1 to l do

begin

g[j+1]:=g[j+1]+g[j] div 10;

g[j]:=g[j] mod 10;

end;

inc(l);

while g[l]>9 do

begin

g[l+1]:=g[l+1]+g[l] div 10;

g[l]:=g[l] mod 10;

inc(l);

end;

if g[l]=0 then dec(l);

end;

procedure gj2;

var

j:longint;

begin

for j:=l downto 1 do

begin

g[j-1]:=g[j-1]+(g[j] mod c[n])\*10;

g[j]:=g[j] div c[n];

end;

while g[l]=0 do dec(l);

end;

procedure sort(l,r:longint);

var

i,j,x,y:longint;

begin

i:=l;

j:=r;

x:=a[(l+r) div 2];

repeat

while a[i]<x do inc(i);

while x<a[j] do dec(j);

if not (i>j) then

begin

y:=a[i];

a[i]:=a[j];

a[j]:=y;

y:=b[i];

b[i]:=b[j];

b[j]:=y;

y:=c[i];

c[i]:=c[j];

c[j]:=y;

inc(i);

dec(j);

end;

until (i>j);

if l<j then sort(l,j);

if i<r then sort(i,r);

end;

begin

readln(n);

readln(b[0],c[0]);

for i:=1 to n do

begin

read(b[i],c[i]);

a[i]:=b[i]\*c[i];

end;

sort(1,n);

l:=1;

g[1]:=b[0];

for i:=1 to n-1 do gj1;

gj2;

for i:=l downto 1 do write(g[i]);

writeln;

end.

# 开车旅行

链表+倍增

1. 按城市高度递增顺序排序，放到链表中
2. 从第一个城市起，找离它最近和次近的城市，记录下来
3. 填每个点向前走2^0步（压缩后的一步，即A走一次B走一次）；
4. 倍增：处理处每个点向前走2^j步能到的城市和到这个城市A,B需要走的距离

方程：f[I,j]=f[f[I,j-1],j-1]]; a[I,j]=a[I,j-1]+a[f[I,j-1],j-1]; b[I,j]=b[I,j-1]+b[f[I,j-1],j-1]

1. 依据s,x判断能走到哪个城市及A,B走过距离

program day13;

label 1;

type

point=^rec;

rec=record

da:longint;

la,ne:point;

end;

var

i,n,m,t,s,top,j,k,ans,dk,lj:longint;

h,a,mi,mmi,b:array[0..100000] of int64;

line:array[0..100000] of point;

fa,na,nb:array[0..100000,-2..17] of int64;

p,q:point;

bo:array[0..100000] of boolean;

re:real;

la,lb,x:int64;

flag:boolean;

procedure sort(l,r:longint);

var

i,j,x,y:longint;

begin

x:=h[(l+r) div 2];

i:=l;

j:=r;

repeat

while h[i]<x do inc(i);

while x<h[j] do dec(j);

if not (i>j) then

begin

y:=h[i];

h[i]:=h[j];

h[j]:=y;

y:=a[i];

a[i]:=a[j];

a[j]:=y;

inc(i);

dec(j);

end;

until i>j;

if l<j then sort(l,j);

if i<r then sort(i,r);

end;

procedure deal;

begin

la:=0;

lb:=0;

k:=s;

flag:=true;

while flag do

begin

j:=0;

while (x>=la+lb+na[k,j]+nb[k,j])and(fa[k,j]<>0) do inc(j);

if (x<la+lb+na[k,j]+nb[k,j])and(j=0) then flag:=false;

if fa[k,0]=0 then flag:=false;

la:=la+na[k,j-1];

lb:=lb+nb[k,j-1];

k:=fa[k,j-1];

end;

if (mmi[k]<>0)and(la+lb+abs(b[mmi[k]]-b[k])<=x) then la:=la+abs(b[mmi[k]]-b[k]);

end;

begin

readln(n);

for i:=1 to n do

begin

read(h[i]);

a[i]:=i;

end;

b:=h;

sort(1,n);

new(line[a[1]]);

line[a[1]]^.la:=nil;

line[a[1]]^.da:=a[1];

for i:=2 to n do

begin

new(line[a[i]]);

line[a[i]]^.la:=line[a[i-1]];

line[a[i]]^.da:=a[i];

line[a[i-1]]^.ne:=line[a[i]];

end;

line[a[n]]^.ne:=nil;

for i:=1 to n do

begin

p:=line[i]^.la;

q:=line[i]^.ne;

if (p=nil)and(q=nil) then goto 1;

if (q<>nil)and((p=nil)or(b[i]-b[p^.da]>b[q^.da]-b[i])) then

begin

mi[i]:=q^.da;

q:=q^.ne;

if (p=nil)and(q=nil) then goto 1;

if (q<>nil)and((p=nil)or(b[i]-b[p^.da]>b[q^.da]-b[i])) then mmi[i]:=q^.da else mmi[i]:=p^.da;

end

else

begin

mi[i]:=p^.da;

p:=p^.la;

if (p=nil)and(q=nil) then goto 1;

if (p<>nil)and((q=nil)or(b[q^.da]-b[i]>=b[i]-b[p^.da])) then mmi[i]:=p^.da else mmi[i]:=q^.da;

end;

1: if line[i]^.la<>nil then line[i]^.la^.ne:=line[i]^.ne;

if line[i]^.ne<>nil then line[i]^.ne^.la:=line[i]^.la;

line[i]:=nil;

end;

for i:=1 to n do if mmi[i]<>0 then

begin

t:=mmi[i];

if mi[t]<>0 then

begin

nb[i,0]:=abs(b[t]-b[mi[t]]);

na[i,0]:=abs(b[i]-b[t]);

end;

fa[i,0]:=mi[t];

end;

for i:=1 to n do fa[i,-1]:=i;

for j:=1 to 17 do

for i:=1 to n do

begin

fa[i,j]:=fa[fa[i,j-1],j-1];

if fa[i,j]<>0 then

begin

na[i,j]:=na[i,j-1]+na[fa[i,j-1],j-1];

nb[i,j]:=nb[i,j-1]+nb[fa[i,j-1],j-1];

end;

end;

read(x);

re:=10000000000;

for s:=1 to n do

begin

deal;

if (lb<>0)and(re>la/lb) then

begin

re:=la/lb;

ans:=s;

end;

end;

writeln(ans);

read(m);

for i:=1 to m do

begin

read(s,x);

deal;

writeln(la,' ',lb);

end;

end.

# 同余方程

扩展欧几里得

依据gcd(a,b)=gcd(b,a mod b);

设ax1+by1=gcd(a,b)=1

则bx2+(a mod b)y2=gcd(b,a mod b);

ax1+by1=bx2+(a mod b)y2;

ax1+by1=bx2+(a-[a/b]\*b)y2=ay2+bx2-(a/b)\*by2;

根据恒等定理得：x1=y2; y1=x2-[a/b]\*y2;

var

a,b,n,m,t:longint;

procedure e(a,b:longint;var x,y:longint);

begin

if b=0 then

begin

x:=1;

y:=0;

exit;

end

else

begin

e(b,a mod b,x,y);

t:=x;

x:=y;

y:=t-(a div b)\*y;

end;

end;

begin

read(a,b);

e(a,b,n,m);

if n<0 then n:=n+b;

writeln(n);

end.

# 借教室

二份答案+前缀和

1. 按订单将答案二分，依此判断区间[l,r]内(l+r)div 2是否满足条件，缩小区间，直至l=r
2. 对于截止到j订单是否满足条件，在sj处+dj，在tj+1处-dj，然后求前缀和。若某一天前缀和大于当天可用教室数，则不能满足条件

var

n,m,i,l,r,now,k,ssum:longint;

room,d,s,t,sum:array[0..1000001]of longint;

flag:boolean;

procedure init;

begin

read(n,m);

for i:=1 to n do

read(room[i]);

for i:=1 to m do

read(d[i],s[i],t[i]);

end;

procedure print;

begin

write(0);

halt;

end;

procedure work;

begin

now:=0;

l:=0;

r:=m+1;

repeat

flag:=true;

k:=(l+r)shr 1;

if k>now then

for i:=now+1 to k do

begin

inc(sum[s[i]],d[i]);

dec(sum[t[i]+1],d[i]);

end

else

for i:=k+1 to now do

begin

dec(sum[s[i]],d[i]);

inc(sum[t[i]+1],d[i]);

end;

now:=k;

ssum:=0;

for i:=1 to n do

begin

inc(ssum,sum[i]);

if ssum>room[i] then begin flag:=false; break; end;

end;

if flag then l:=k+1

else r:=k;

until l=r;

if (k=m)and(flag) then print;

writeln(-1);

write(l);

end;

begin

init;

work;

end.

# 疫情控制

DFS+倍增+二分答案+贪心

1. dfs找出倍增序列
2. 二分答案：在给定时间ans内
3. 对于某个军队：若能到达首都，记下到达首都后的剩余时间

若不能，让它尽可能的向首都走

2）dfs找出来哪个边境城市需要军队驻守，则需要走到从这个边境城市出发向首都走所需经过的最后一个点，按这个点到首都距离从大到小排序，

若这个点有军队通过它进入首都，选择剩余时间不能走到这个点且经过这个点能到首都的军队。否则从到达首都的所有军队中选一个剩余时间最小的军队驻守。

label 1;

var

i,j,k,n,m,f1,t,min,max,top,ans,cans,mi,q,dk,c1,cm1:longint;

f,d,r,e,bo,s,a,c:array[0..100000] of longint;

fa,lx:array[0..50000,0..16] of longint;

cm:array[0..1,0..50000] of longint;

boo,need,had,have:array[0..50000] of boolean;

flag:boolean;

procedure csort(l,r:longint);

var

ii,ij,xx,xy:longint;

begin

xx:=c[(l+r) div 2];

ii:=l;

ij:=r;

repeat

while (c[ii]<xx)and(ii<r) do inc(ii);

while (xx<c[ij])and(ij>l) do dec(ij);

if not (ii>ij) then

begin

xy:=c[ii];

c[ii]:=c[ij];

c[ij]:=xy;

inc(ii);

dec(ij);

end;

until ii>ij;

if l<ij then csort(l,ij);

if ii<r then csort(ii,r);

end;

procedure dfs(i,T1:longint);

var

j,k:longint;

begin

if bo[i]<>i then

begin

j:=1;

i:=c[top];

while top-j>0 do

begin

inc(fa[i,0]);

fa[i,fa[i,0]]:=c[top-j];

lx[i,fa[i,0]]:=d[c[top]]-d[c[top-j]];

j:=j\*2;

end;

exit;

end;

d[i]:=t;

bo[i]:=t1;

inc(top);

c[top]:=i;

for j:=f[i] to f[i+1]-1 do

begin

t:=t+a[j];

dfs(e[j],bo[i]);

t:=t-a[j];

end;

c[top]:=0;

dec(top);

end;

procedure sort(l,r:longint);

var

i,j,x,y:longint;

begin

x:=s[(l+r) div 2];

i:=l;

j:=r;

repeat

while s[i]<x do inc(i);

while x<s[j] do dec(j);

if not (i>j) then

begin

y:=a[i];

a[i]:=a[j];

a[j]:=y;

y:=s[i];

s[i]:=s[j];

s[j]:=y;

y:=e[i];

e[i]:=e[j];

e[j]:=y;

inc(i);

dec(j);

end;

until i>j;

if l<j then sort(l,j);

if i<r then sort(i,r);

end;

procedure ssort(l,r:longint);

var

ii,ij,xx,xy:longint;

begin

xx:=a[(l+r) div 2];

ii:=l;

ij:=r;

repeat

while (a[ii]<xx)and(ii<r) do inc(ii);

while (xx<a[ij])and(ij>l) do dec(ij);

if not (ii>ij) then

begin

xy:=a[ii];

a[ii]:=a[ij];

a[ij]:=xy;

xy:=s[ii];

s[ii]:=s[ij];

s[ij]:=xy;

xy:=e[ii];

e[ii]:=e[ij];

e[ij]:=xy;

inc(ii);

dec(ij);

end;

until ii>ij;

if l<ij then ssort(l,ij);

if ii<r then ssort(ii,r);

end;

procedure init;

begin

read(n);

m:=n-1;

for i:=1 to m do readln(s[i],e[i],a[i]);

for i:=m+1 to 2\*m do

begin

s[i]:=e[i-m];

e[i]:=s[i-m];

a[i]:=a[i-m];

end;

for i:=1 to n do bo[i]:=i;

sort(1,2\*m);

f[1]:=1;

f1:=1;

for i:=1 to 2\*m+1 do

if s[i]<>f1 then

begin

inc(f1);

f[f1]:=i;

end;

q:=f[2]-f[1];

ssort(1,q);

bo[1]:=-1;

c[1]:=1;

top:=1;

for k:=f[1] to f[2]-1 do

begin

d[e[k]]:=t+a[k];

t:=t+a[k];

dfs(e[k],-e[k]);

t:=t-a[k];

end;

read(m);

for i:=1 to m do read(r[i]);

if m<q then writeln(-1);

for i:=1 to n do

for j:=1 to fa[i,0] do if max<lx[i,j] then max:=lx[i,j];

max:=max\*2;

end;

procedure ddfs(ii:longint);

var

kk:longint;

begin

if boo[ii] then exit;

boo[ii]:=true;

for kk:=f[ii] to f[ii+1]-1 do if not have[e[kk]] then ddfs(e[kk]);

if (f[ii+1]-f[ii]=1) then need[abs(bo[ii])]:=true;

end;

procedure check;

begin

ans:=(max+min) div 2;

fillchar(boo,sizeof(boo),false);

fillchar(had,sizeof(had),false);

fillchar(need,sizeof(need),false);

fillchar(have,sizeof(have),false);

fillchar(c,sizeof(c),0);

c1:=0;

for i:=1 to n do cm[0,i]:=maxlongint;

for i:=1 to n do cm[1,i]:=0;

for i:=1 to m do

if ans>=d[r[i]] then

begin

if (cm[0,-bo[r[i]]]>=ans-d[r[i]])and(ans-d[r[i]]<=d[r[i]]) then

begin

cm[1,-bo[r[i]]]:=i;

cm[0,-bo[r[i]]]:=ans-d[r[i]];

end

end

else

begin

k:=r[i];

dk:=k;

cans:=ans;

flag:=true;

had[i]:=true;

while flag do

begin

j:=1;

while (cans>=lx[k,j])and(lx[k,j+1]<>0) do

begin

dk:=fa[k,j];

inc(j);

end;

if (j=1)and(cans<lx[k,j]) then

begin

have[dk]:=true;

flag:=false;

end

else

begin

cans:=cans-(d[k]-d[dk]);

k:=dk;

end;

end;

end;

boo[1]:=true;

for k:=f[1] to f[2]-1 do

if not have[e[k]]then ddfs(e[k]);

for i:=1 to q do

if (need[e[i]])and(cm[1,e[i]]<>0) then

begin

had[cm[1,e[i]]]:=true;

need[e[i]]:=false;

end;

for i:=1 to m do if not had[i] then

begin

inc(c1);

c[c1]:=ans-d[r[i]];

end;

csort(1,c1);

for i:=q downto 1 do if need[e[i]] then

begin

if c[c1]<d[e[i]] then

begin

ans:=max;

exit

end;

dec(c1);

end;

ans:=min;

end;

begin

init;

repeat

ans:=(max+min) div 2;

fillchar(boo,sizeof(boo),false);

fillchar(had,sizeof(had),false);

fillchar(need,sizeof(need),false);

fillchar(have,sizeof(have),false);

fillchar(c,sizeof(c),0);

c1:=0;

for i:=1 to n do cm[0,i]:=maxlongint;

for i:=1 to n do cm[1,i]:=0;

for i:=1 to m do

if ans>=d[r[i]] then

begin

if (cm[0,-bo[r[i]]]>=ans-d[r[i]])and(ans-d[r[i]]<=d[r[i]]) then

begin

cm[1,-bo[r[i]]]:=i;

cm[0,-bo[r[i]]]:=ans-d[r[i]];

end

end

else

begin

k:=r[i];

dk:=k;

cans:=ans;

flag:=true;

had[i]:=true;

while flag do

begin

j:=1;

while (cans>=lx[k,j])and(lx[k,j+1]<>0) do

begin

dk:=fa[k,j];

inc(j);

end;

if (j=1)and(cans<lx[k,j]) then

begin

have[dk]:=true;

flag:=false;

end

else

begin

cans:=cans-(d[k]-d[dk]);

k:=dk;

end;

end;

end;

boo[1]:=true;

for k:=f[1] to f[2]-1 do

if not have[e[k]] then ddfs(e[k]);

for i:=1 to q do if (need[e[i]])and(cm[1,e[i]]<>0) then

begin

had[cm[1,e[i]]]:=true;

need[e[i]]:=false;

end;

for i:=1 to m do if not had[i] then

begin

inc(c1);

c[c1]:=ans-d[r[i]];

end;

csort(1,c1);

for i:=q downto 1 do if need[e[i]] then

begin

if c[c1]<d[e[i]] then

begin

min:=ans;

goto 1;

end;

dec(c1);

end;

max:=ans;

1:

until (max=min)or(max-min=1);

check;

writeln(ans);

end.