

# 牛客多校第一场

因为事务繁忙、时间紧迫，这里的题解区仅暂存一下通关代码。

有趣的是，过的几个题都是C语言，暂时还没用到C++的STL

## D

高斯消元板子，含快速幂，没什么内容。

```
#include<stdio.h>
#include<string.h>

#define MOD 998244353

long long n,var,equ,pic[256][256],x[256];
long long ans[256];

long long abs(long long x)
{
    return (x>0)?x:-x;
}

long long QPow(long long bas,long long t)
{
    long long ret=1;
    for(;t;t>>=1,bas=(bas*bas)%MOD)
    {
        if(t&1LL)
        {
            ret=(ret*bas)%MOD;
        }
    }
    return ret;
}

void Gauss()
{
    long long i,j,k,col,maxr;
    for(k=0,col=0;k<=var&&col<var;k++,col++)
    {
        maxr=k;
        for(i=k+1;i<=var;i++)
        {
            if(abs(pic[i][col])>abs(pic[maxr][col]))
            {
                maxr=i;
            }
        }
    }
}
```

```
    }
}
if(k!=maxr)
{
    for(j=col;j<var;j++)
    {
        long long temp=pic[k][j];
        pic[k][j]=pic[maxr][j];
        pic[maxr][j]=temp;
    }
    long long temp=x[k];
    x[k]=x[maxr];
    x[maxr]=temp;
}
x[k]*=QPow(pic[k][col],998244351);
x[k]%=MOD;
for(j=col+1;j<var;j++)
{
    pic[k][j]=(pic[k][j]%MOD*QPow(pic[k][col],998244351))%MOD;
}
pic[k][col]=1;
for(i=0;i<=u;i++)
{
    if(i!=k)
    {
        x[i]-=x[k]*pic[i][col];
        x[i]%=MOD;
        for(j=col+1;j<var;j++)
        {
            pic[i][j]-=pic[k][j]*pic[i][col];
            pic[i][j]%=MOD;
        }
        pic[i][col]=0;
    }
}
}
}

int main()
{
    while(~scanf("%d",&n))
    {
        var=equ=n;
        int i;
        for(i=0;i<n;i++)
        {
            int j;
            for(j=0;j<n;j++)
            {
                scanf("%lld",&pic[i][j]);
            }
        }
    }
}
```

```

    }
}
for(i=0;i<n;i++)
{
    scanf("%lld",&x[i]);
    ans[i]=x[i];
}
Gauss();
long long l=0;
for(i=0;i<n;i++)
{
    l+=(ans[i]*x[i])%MOD;
    l%=MOD;
}
printf("%lld\n",(l>=0?l:l+MOD));
}
return 0;
}

```

## F

本题几乎什么都没用到，却因为max的小问题WA了非常多次。深表惭愧。

```

#include<stdio.h>
#include<string.h>

int main()
{
    int lena,lenb,flag;
    long long max;
    char a[100005],b[100005];
    while(~scanf("%s%s",a,b))
    {
        lena=strlen(a);
        lenb=strlen(b);
        max=2*(lena>lenb?lena:lenb);
        flag=0;
        long long i;
        for(i=0;i<max;i++)
        {
            if(a[i%lena]<b[i%lenb])
            {
                printf("<\n");
                flag=1;
                break;
            }
            if(a[i%lena]>b[i%lenb])
            {

```

```
        printf(">\n");
        flag=1;
        break;
    }
}
if(!flag)
{
    printf("=\n");
}
}
return 0;
}
```

后来得知的一个结论是，对于两个循环字符串字典序比较 $s$ 循环大于 $t$ 循环，等价于 $st$ 大于 $ts$ 。这可真是有趣的结论，大概比这个暴力算法更优吧。

## I

最难的一道题。网络流学得不好，唉。

```
#include<stdio.h>
#include<string.h>

char g[1049][1049],inque[1049],inpath[1049];
char inhua[1049];
int st,ed,newbase,ans,ne;
int base[1049],pre[1049],match[1049];
int head,tail,que[1049];
int a[1049],b[1049],d[1049],mp[1049][1049],m,n;

int lca(int u,int v)
{
    memset(inpath,0,sizeof(inpath));
    while(1)
    {
        u=base[u];
        inpath[u]=1;
        if(u==st)
        {
            break;
        }
        u=pre[match[u]];
    }
    while(1)
    {
        v=base[v];
        if(inpath[v])
```

```
        {
            break;
        }
        v=pre[match[v]];
    }
    return v;
}

void reset(int u)
{
    int v;
    while(base[u]!=newbase)
    {
        v=match[u];
        inhua[base[u]]=inhua[base[v]]=1;
        u=pre[v];
        if(base[u]!=newbase)
        {
            pre[u]=v;
        }
    }
}

void contract(int u,int v)
{
    newbase=lca(u,v);
    memset(inhua,0,sizeof(inhua));
    reset(u);
    reset(v);
    if(base[u]!=newbase)
    {
        pre[u]=v;
    }
    if(base[v]!=newbase)
    {
        pre[v]=u;
    }
    int i;
    for(i=1;i<=ne;i++)
    {
        if(inhua[base[i]])
        {
            base[i]=newbase;
            if(!inque[i])
            {
                que[tail]=i;
                tail++;
                inque[i]=1;
            }
        }
    }
}
```

```
}  
  
void findaug()  
{  
    memset(inque,0,sizeof(inque));  
    memset(pre,0,sizeof(pre));  
    int i;  
    for(i=1;i<=ne;i++)  
    {  
        base[i]=i;  
    }  
    head=tail=1;  
    que[tail]=st;  
    tail++;  
    inque[st]=1;  
    ed=0;  
    while(head<tail)  
    {  
        int u=que[head];  
        head++;  
        int v;  
        for(v=1;v<=ne;v++)  
        {  
            if(g[u][v]&&(base[u]!=base[v])&&match[u]!=v)  
            {  
                if(v==st|| (match[v]>0)&&pre[match[v]]>0)  
                {  
                    contract(u,v);  
                }  
                else if(pre[v]==0)  
                {  
                    pre[v]=u;  
                    if(match[v]>0)  
                    {  
                        que[tail]=match[v];  
                        tail++;  
                        inque[match[v]]=1;  
                    }  
                    else  
                    {  
                        ed=v;  
                        return;  
                    }  
                }  
            }  
        }  
    }  
}  
  
void aug()
```

```
{
    int u,v,w;
    u=ed;
    while(u>0)
    {
        v=pre[u];
        w=match[v];
        match[v]=u;
        match[u]=v;
        u=w;
    }
}

void edmonds()
{
    memset(match,0,sizeof(match));
    int u;
    for(u=1;u<=ne;u++)
    {
        if(match[u]==0)
        {
            st=u;
            findaug();
            if(ed>0)
            {
                aug();
            }
        }
    }
}

void create()
{
    ne=0;
    memset(g,0,sizeof(g));
    int i;
    for(i=1;i<=n;i++)
    {
        int j;
        for(j=1;j<=d[i];j++)
        {
            mp[i][j]=++ne;
        }
    }
    for(i=0;i<m;i++)
    {
        int j;
        for(j=1;j<=d[a[i]];j++)
        {
            g[mp[a[i]][j]][ne+1]=g[ne+1][mp[a[i]][j]]=1;
        }
    }
}
```

```
        for(j=1;j<=d[b[i]];j++)
        {
            g[mp[b[i]][j]][ne+2]=g[ne+2][mp[b[i]][j]]=1;
        }
        g[ne+1][ne+2]=g[ne+2][ne+1]=1;
        ne+=2;
    }
}

void print()
{
    ans=0;
    int i;
    for(i=1;i<=ne;i++)
    {
        if(match[i]!=0)
        {
            ans++;
        }
    }
    if(ans==ne)
    {
        printf("Yes\n");
    }
    else
    {
        printf("No\n");
    }
}

int main()
{
    while(~scanf("%d%d",&n,&m))
    {
        int i;
        for(i=1;i<=n;i++)
        {
            scanf("%d",&d[i]);
        }
        for(i=0;i<m;i++)
        {
            scanf("%d%d",&a[i],&b[i]);
        }
        create();
        edmonds();
        print();
    }
    return 0;
}
```

J

我大概是快速幂爱好者，可能不太喜欢递归。这个题要交C++14而不是C++11才不会RE

```
#include<stdio.h>

#define MOD 998244353

long long fact[4000010];

long long init()
{
    fact[0]=1;
    long long i;
    for(i=1;i<4000005;i++)
    {
        fact[i]=(fact[i-1]*i)%MOD;
    }
}

long long QPow(long long bas,long long t)
{
    long long ret=1;
    for(;t;t>>=1,bas=(bas*bas)%MOD)
    {
        if(t&1LL)
        {
            ret=(ret*bas)%MOD;
        }
    }
    return ret;
}

int n;

int main()
{
    init();
    while(~scanf("%d",&n))
    {
        long long a=(fact[n]*fact[n])%MOD;
        long long b=QPow(fact[2*n+1],(long long)998244351);
        printf("%lld\n", (a*b)%MOD);
    }
}
```

## A

从这里以下是补题的部分。

题目：定义字符串到字符串的映射 $F$ 第 $i$ 位为字符串 $t$ 第 $i$ 位 $t_i$ 到前面相同字符的距离，若没有则为0。

给出两元素 $a, b$ 组成的长为 $n$ 的串 $t$ 要回答其所有后缀作用 $F$ 后字典序的排列。

标程定理：对于只有两元的串，这里的 $F$ 和 $G$ 等价。

$G$ 仅将“到前面相同字符的距离，若没有则为0”改为“到后面相同字符的距离，若没有则为 $n$ 并且在末端补充 $n+1$ ”

总之是神奇字符串结论。先记下来。

```
#include<stdio.h>

char str[100010];
int s[100010],sa[100010],rk[100010],t[100010],c[100010],height[100010];

void getsa(int n,int m)
{
    s[n++]=0;
    int *x=rk,*y=t,i,k,num;
    for(i=0;i<m;i++)
    {
        c[i]=0;
    }
    for(i=0;i<n;i++)
    {
        c[x[i]=s[i]]++;
    }
    for(i=0;i<m;i++)
    {
        c[i]+=c[i-1];
    }
    for(i=n-1;i>=0;i--)
    {
        --c[x[i]];
        sa[c[x[i]]]=i;
    }
    for(k=1,num=1;num<n;k<=<=1,m=num)
    {
        num=0;
        for(i=n-k;i<n;i++)
        {
            y[num]=i;
            num++;
        }
    }
}
```

```

    for(i=0;i<n;i++)
    {
        if(sa[i]>=k)
        {
            y[num]=sa[i]-k;
            num++;
        }
    }
    for(i=0;i<m;i++)
    {
        c[i]=0;
    }
    for(i=0;i<n;i++)
    {
        c[x[y[i]]]++;
    }
    for(i=0;i<m;i++)
    {
        c[i]+=c[i-1];
    }
    for(i=n-1;i>=0;i--)
    {
        --c[x[y[i]]];
        sa[c[x[y[i]]]]=y[i];
    }
    int *temp=x;
    x=y;
    y=temp;
    x[sa[0]]=0;
    num=1;
    for(i=1;i<n;i++)
    {
x[sa[i]]=(y[sa[i]]==y[sa[i-1]]&&y[sa[i]+k]==y[sa[i-1]+k])?num-1:num++;
    }
}

void getheight(int n)
{
    int i,j,k=0;
    for(i=1;i<=n;i++)
    {
        rk[sa[i]]=i;
    }
    for(i=0;i<n;height[rk[i++]]=k)
    {
        j=sa[rk[i]-1];
        k=k?k-1:k;
        while(s[i+k]==s[j+k])
        {
            k++;
        }
    }
}

```

```
    }  
  }  
  return;  
}  
  
int main()  
{  
  int n;  
  while(~scanf("%d%s",&n,str))  
  {  
    s[n]=n+1;  
    int a=n+1,b=n+1;  
    int i;  
    for(i=n-1;i>=0;i--)  
    {  
      if(str[i]=='a')  
      {  
        s[i]=(a==n+1)?n:a-i;  
        a=i;  
      }  
      else  
      {  
        s[i]=(b==n+1)?n:b-i;  
        b=i;  
      }  
    }  
    getsa(n+1,n+2);  
    getheight(n+1);  
    for(i=n;i;i--)  
    {  
      printf("%d ",1+sa[i]);  
    }  
    puts("");  
  }  
  return 0;  
}
```

From: <https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link: <https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:namespace:%E7%89%9B%E5%AE%A2%E5%A4%9A%E6%A0%A1%E7%AC%AC%E4%B8%80%E5%9C%BA&rev=1594864510>

Last update: 2020/07/16 09:55