2020牛客暑期多校训练营（第一场）

比赛情况

题号	A	B	C	D	E	F	G	H	I	J
状态	Ø	-	-	-	-	O	-	O	O	O

O 在比赛中通过 Ø 赛后通过 ! 尝试了但是失败了 - 没有尝试

比赛时间

2020-07-12 12:00-17:00

提交记录

题解

A - B-Suffix Array

定义函数 $B(t_1t_2\ldots t_k)=b_1b_2\ldots b_k$ ，b_i 为字符串 t 第 i 位距离前面最近的相同字符的距离，若没有则为 0。给出一个 a,b 组成的串 s ，要回答其所有后缀 B 函数字典序的排列。

似乎是个论文题，结论只在只有两种元素的时候成立。

> Let $C_i = min_{j > i and s_j = s_i} {j - i}$
> The B-Suffix Array is equivalent to the suffix array of $C_1 C_2 \ldots C_n$

与 B 不同这个 $C(t_1t_2\ldots t_k)$ 是可以从后往前扫一遍得到的，如果后面没有相同字符则 $C_i=n$ ，在此之后还要再补一个 $C_{n+1}=n+1$ 保证正确性。对 C 求后缀数组，倒过来即答案。

```
#include<bits/stdc++.h>
#define INF 0x3f3f3f3f
#define ll long long
using namespace std;
const int N=1e5+10;
char str[N];
int s[N],sa[N],rk[N],t[N],c[N],height[N];
void get_sa(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--)sa[--c[x[i]]]=i;
    for(k=1,num=1;num<n;k<<=1,m=num)
    {
        for(num=0,i=n-k;i<n;i++)y[num++]=i;
```

```
        for(i=0;i<n;i++)if(sa[i]>=k)y[num++]=sa[i]-k;
        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--)sa[--c[x[y[i]]]]=y[i];
        for(swap(x,y),i=num=1,x[sa[0]]=0;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 get_height(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)
    for(k=k?k-1:k,j=sa[rk[i]-1];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;
        for(int 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;
        }
        get_sa(n+1,n+2),get_height(n+1);
        for(int i=n;i;i--)printf("%d ",1+sa[i]);
        puts("");
    }
    return 0;
}
```

F - Infinite String Comparision

两个无限循环的字符串给出循环节比较大小。

比较 $a+b-\gcd(a,b)$ 长度即可，也可以将较长字符串复制两遍。

拓展阅读（把我看懵了）循环小数解法 BY KeHe

I - 1 or 2

我居然会过带花树.jpg

题意：给定一个无向图，给定点的度数限制，想要你选择其中一些边使得度数限制被满足。度数只可能是1或2。

如果度数是1的话那么就是普通的一般图匹配，那么自然的想到对度数是2的拆点，普通的拆点可能会导致某条边被重复使用，那么解决方案就是对一条边新建两个新点，然后原来普通的拆点分别向这个边的两个新点连边，就可以保证这条边只被使用一次。

```
#include <stdio.h>
#include <queue>
#include <string.h>
#include <stdlib.h>
#include <algorithm>
using namespace std;
const int N = 305;
const int M = 90005;
struct E
{int next,to;}e[M];
int head[N],tot;
void add(int x,int y)
{
    e[++tot].to = y;e[tot].next = head[x];head[x]=tot;
    e[++tot].to = x;e[tot].next = head[y];head[y]=tot;
}
int nxt[N],pre[N],fa[N],v[N],s[N],n,m;
int getfa(int x)
{
    if(fa[x]==x)return fa[x];
    else return fa[x]=getfa(fa[x]);
}
queue<int>Q;
int getlca(int x,int y)
{
    static int times=0;
    ++times;
    x = getfa(x),y=getfa(y);
    for(;;swap(x,y))
    if(x)
    {
        if(v[x]==times)
            return x;
        v[x] = times;
        x = getfa(pre[nxt[x]]);
    }
}
void blossom(int x,int y,int lca)
{
```

```
    while(getfa(x)!=lca)
    {
        pre[x]=y;
        y=nxt[x];
        if(s[y]==1)
            Q.push(y),s[y]=0;
        if(fa[x]==x)
            fa[x]=lca;
        if(fa[y]==y)
            fa[y]=lca;
        x=pre[y];
    }
}
bool get_partner(int x)
{
    for(int i=0;i<=n;i++)
        fa[i]=i,s[i]=-1;
    while(!Q.empty())
        Q.pop();
    Q.push(x);
    s[x] = 0;
    while(!Q.empty())
    {
        int x= Q.front();
        Q.pop();
        for(int i = head[x];i;i=e[i].next)
        {
            if(s[e[i].to]==-1)
            {
                s[e[i].to]=1;
                pre[e[i].to]=x;
                if(!nxt[e[i].to])
                {
                    for(int u=e[i].to,v = x,last;v;u=last,v=pre[u])
                        last = nxt[v],nxt[v]=u,nxt[u]=v;
                    return true;
                }
                Q.push(nxt[e[i].to]);
                s[nxt[e[i].to]]=0;
            }else if(s[e[i].to]==0&&getfa(e[i].to)!=getfa(x))
            {
                int l = getlca(e[i].to,x);
                blossom(x,e[i].to,l);
                blossom(e[i].to,x,l);
            }
        }
    }
    return false;
}
int d[N];
```

```
int x[N],y[N];
int no[N][3];
int main()
{
    while (scanf("%d%d",&n,&m)!=EOF) {
        for (int i = 1;i<= n;i++)
            scanf("%d",&d[i]);
        for(int i = 1;i<= m;i++)
            scanf("%d%d",&x[i],&y[i]);
        int nn = 0;
        for (int i = 1;i<= n;i++)
            for (int j = 1;j<= d[i];j++)
                no[i][j] = ++nn;
        for (int i = 1;i<= m;i++)
        {
            for (int j = 1;j<= d[x[i]];j++)
                add(nn+1,no[x[i]][j]);
            for (int j = 1;j<= d[y[i]];j++)
                add(nn+2,no[y[i]][j]);
            add(nn+1,nn+2);
            nn+=2;
        }
        n = nn;
        for (int i = 1;i<= n;i++)
            nxt[i] = 0;
        for(int i = n;i;i--)
            if(!nxt[i])
                get_partner(i);
        int cnt = 0;
        for (int i = 1;i<= n;i++)
            if (nxt[i])
                cnt++;
        /*printf("%d\n",ans);
        for(int i = 1;i<= n;i++)
            printf("%d ",next[i]);
        printf("\n");*/
        if (cnt == n)
            printf("Yes\n");
        else printf("No\n");
        for (int i = 0;i< N;i++)
        {
            pre[i] = fa[i] = nxt[i] = head[i] = v[i] = s[i] = 0;
            d[i] = x[i] = y[i] = no[i][1] = no[i][2] = 0;
        }
        tot = 0;
    }
    return 0;
}
```

J - Easy Integration

$\int_0^1(x-x^2)^ndx=\frac{(n!)^2}{(2n+1)!}$ 可以*oeis/wolframalpha/分部积分*

```
#include<bits/stdc++.h>
#define ll long long
using namespace std;
const ll mod=998244353;
const ll N=2e6+10;
int n;
ll fac[N],inv[N];
void init()
{
    fac[0]=1,inv[1]=1;
    for(int i=1;i<N;i++)
    fac[i]=fac[i-1]*i%mod;
    for(int i=2;i<N;i++)
    inv[i]=inv[mod%i]*(mod-mod/i)%mod;
    inv[0]=1;
    for(int i=1;i<N;i++)
    inv[i]=inv[i-1]*inv[i]%mod;
}
int main()
{
    init();
    while(~scanf("%d",&n))
    printf("%lld\n",fac[n]*fac[n]%mod*inv[2*n+1]%mod);
    return 0;
}
```

比赛总结与反思