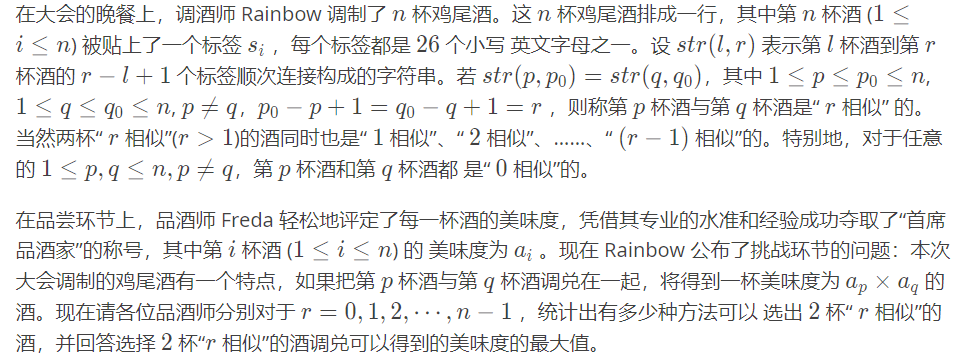
NOI品酒大会



某些题目求解时要求你将后缀数组划分成若干个连续 LCP 长度大于等于某一值的段，亦即将  数组划分成若干个连续最小值大于等于某一值的段并统计每一段的答案。如果有多次询问，我们可以将询问离线。观察到当给定值单调递减的时候，满足条件的区间个数总是越来越少，而新区间都是两个或多个原区间相连所得，且新区间中不包含在原区间内的部分的  值都为减少到的这个值。我们只需要维护一个并查集，每次合并相邻的两个区间，并维护统计信息即可。

首先很容易发现，对于任意一对r相似，它一定是k(0<k<r)相似的

所以求出height数组后按其中的值排序，然后从大到小做

当前需要处理的串为i和i-1，设前缀长度为k

易知若将两个并查集合并，则当前的前缀在并查集中一定是最小的，所以Ans[k][0]+=两棵树size的乘积（因为任意两两前缀都是k相似的，可以配对）

除了并查集的size，还维护一下并查集的max和min值，

则Ans[k][1]=max(Ans[k][1],Max1Max2,Min1Min2)

维护min值是为了防止有很小的负数这种情况（负负得正）

最后因为Ans[i]也是满足Ans[i+1]的，所以做个前缀和合并一下答案就好

using namespace std;

#define ll long long

#define MAX 320000

inline int read()

{

int x=0,t=1;char ch=getchar();

while((ch<'0'||ch>'9')&&ch!='-')ch=getchar();

if(ch=='-')t=-1,ch=getchar();

while(ch<='9'&&ch>='0')x=x\*10+ch-48,ch=getchar();

return x\*t;

}

int n,v[MAX];

char s[MAX];

ll ans1[MAX],ans2[MAX];

struct SA

{

int a[MAX];

int x[MAX],y[MAX],t[MAX];

int SA[MAX],height[MAX],rk[MAX];

bool cmp(int i,int j,int k){return y[i]==y[j]&&y[i+k]==y[j+k];}

void GetSA()

{

int m=50;

for(int i=1;i<=n;++i)t[x[i]=a[i]]++;

for(int i=1;i<=m;++i)t[i]+=t[i-1];

for(int i=n;i>=1;--i)SA[t[x[i]]--]=i;

for(int k=1;k<=n;k<<=1)

{

int p=0;

for(int i=n-k+1;i<=n;++i)y[++p]=i;

for(int i=1;i<=n;++i)if(SA[i]>k)y[++p]=SA[i]-k;

for(int i=0;i<=m;++i)t[i]=0;

for(int i=1;i<=n;++i)t[x[y[i]]]++;

for(int i=1;i<=m;++i)t[i]+=t[i-1];

for(int i=n;i>=1;--i)SA[t[x[y[i]]]--]=y[i];

swap(x,y);

x[SA[1]]=p=1;

for(int i=2;i<=n;++i)

x[SA[i]]=cmp(SA[i],SA[i-1],k)?p:++p;

if(p>=n)break;

m=p;

}

for(int i=1;i<=n;++i)rk[SA[i]]=i;

for(int i=1,j=0;i<=n;++i)

{

if(j)--j;

while(a[i+j]==a[SA[rk[i]-1]+j])++j;

height[rk[i]]=j;

}

}

}SA;

bool cmp(int a,int b){return SA.height[a]>SA.height[b];}

int id[MAX];

int f[MAX],mm[MAX],mi[MAX],size[MAX];

ll ans[MAX];

int getf(int x){return x==f[x]?x:f[x]=getf(f[x]);}

void Merge(int x,int y,int len)

{

x=getf(x);y=getf(y);

f[y]=x;

ans1[len]+=1ll\*size[x]\*size[y];

size[x]+=size[y];

ans[x]=max(ans[x],ans[y]);

ans[x]=max(ans[x],max(1ll\*mm[x]\*mm[y],1ll\*mi[x]\*mi[y]));

ans[x]=max(ans[x],max(1ll\*mm[x]\*mi[y],1ll\*mi[x]\*mm[y]));

mm[x]=max(mm[x],mm[y]);

mi[x]=min(mi[x],mi[y]);

ans2[len]=max(ans2[len],ans[x]);

}

int main()

{

n=read();

scanf("%s",s+1);

for(int i=1;i<=n;++i)SA.a[i]=s[i]-96;

SA.GetSA();

for(int i=1;i<=n;++i)v[i]=read(),id[i]=i;;

for(int i=1;i<=n;++i)f[i]=i,size[i]=1,mm[i]=mi[i]=v[i],ans[i]=-1e18;

memset(ans2,-63,sizeof(ans2));

sort(&id[2],&id[n+1],cmp);

for(int i=2;i<=n;++i)

Merge(SA.SA[id[i]],SA.SA[id[i]-1],SA.height[id[i]]);

for(int i=n;i>=0;--i)ans1[i]+=ans1[i+1];

for(int i=n;i>=0;--i)ans2[i]=max(ans2[i],ans2[i+1]);

for(int i=0;i<n;++i)printf("%lld %lld\n",ans1[i],!ans1[i]?0:ans2[i]);

return 0;

}