Suffix Array

前置知识

系统学习

应用

最长公共子串

LCS

自己造的板子 α 0.0.1

```
    void suffix_array(){
    for (int i=1;i<=n;i++) rank[i]=a[i],sa[i]=i;//初始，未进行排序时,rank为ascii码，sa[i]仅仅是第i名的下标，充当为一个队列的作用。
    for (int k=1;k<n;k<<1){
        memset(cnt,0,sizeof(cnt));
        //对第二关键字进行桶排序
        //cnt 桶， sa[i]<->按照队列一个个调取元素，入栈。 rank[]->元素的排名
        for (int i=1;i<=n;i++) cnt[rank[sa[i]+k]]++;

        //由于将要对sa数组进行新的赋值操作（也就是一次倍增的新的排序，老的sa数组必须被保存下来，不然就像朴素swap(a,b)却没有temp一样
        for (int i=1;i<=n;i++) tmprank[i]=sa[i];
        for (int i=n;i>=1;i--) sa[cnt[tmprank[i]+k]--]=tmprank[i];

        //对第一关键字进行排序
        memset(cnt,0,sizeof(cnt));
        //基数排序，按照原队列（sa[1]->sa[n]（刚刚用第二关键字排好)的☆第一关键字的大小☆进桶再出桶即可
        for (int i=1;i<=n;i++) cnt[rank[sa[i]]]++;
        for (int i=1;i<=n;i++) tmprank[i]=s[i];
        for (int i=n;i>=1;i--) sa[cnt[tmprank[i]]--]=tmprank[i];

        // 把rank更新一下。
        tmprank[1]=1;
        //为什么不是rank[i]而是rank[sa[i]]? 因为rank[]和sa[]互相映射，而倍增时操作的永远是一个“在i--2^k-1条件下有序的序列”不是“下标有序的序列”而“此种条件”的调用方法就是sa[i]
        for (int i=2;i<=n;i++) if (rank[sa[i]]==rank[sa[i-1]] && rank[sa[i+k]]==rank[sa[i-1+k]]) tmprank[sa[i]]=tmprank[sa[i-1]];
        else tmprank[i]=tmprank[i-1]+1;
        for (int i=1;i<=n;i++) rank[sa[i]]=ranktmp[sa[i]];
    }

}
```

以上是我自己对于后缀数组的一些理解。

OJ课程内的新题目

p - 1152 时区转换==bailian - 2966时区转换。
除了输出分钟%02d变成了%d。其他完全一致

即使如此，将bailian2966的%d改成%02d可以AC levoj
而将p1152的%02d改为%d,却过不了bailian-2966
非常非常神奇