Trie 树

一、 字典树简介

1、概念

字典树，也称 Trie、字母树，指的是某个字符串集合对应的形如下图的有根树。树的每条边上对应有恰好一个字符，每个顶点代表从根到该节点的路径所对应的字符串（将所有经过的边上的字符按顺序连接起来）。有时我们也称 Trie 上的边为转移，顶点为状态。

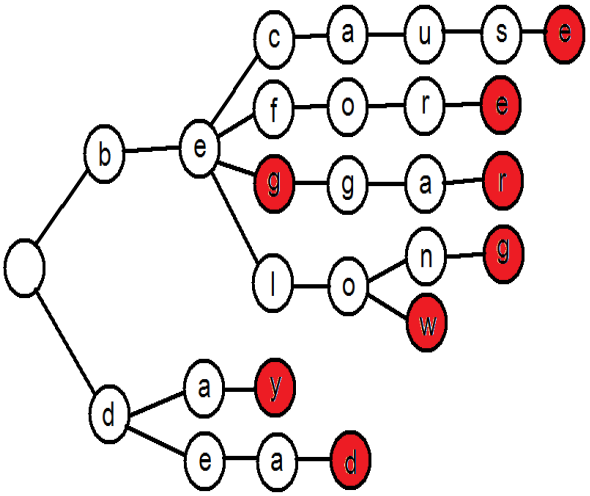
顶点上还能存储额外的信息，例如下图中双圈圈表示顶点所代表的字符串是实际字符串集合中的元素。而实际字符串中的元素都会对应 Trie 中某个顶点代表的串。实际上，任意一个节点所代表的字符串，都是实际字符串集合中某些串的前缀，特别地，根节点表示空串。

此外，对于任意一个节点，它到它儿子节点边上的字符都互不相同。可以发现，Trie 很好的利用了串的公共前缀，节约了存储空间。

若将字符集看做是小写英文字母，则 Trie 也可以看做是一个 26 叉树，在插入询问新字符串时与树一样，找到对应的边往下走。

例如：需要保存下面的 8 个单词

because
before
beg
beggar
belong
below
day
dead



2、trie 树的特点

- 根节点不包含字母，除根节点外每一个节点都仅包含一个英文字母
- 从根节点到某一节点，路径上经过的字母依次连起来所构成的字母序列，称为该节点对应的单词。
- 每个节点的所有儿子包含的字母都不相同。

二、 具体实现

字典树主要有两种操作，插入和查询，这两种操作都非常简单，用一个一重循环均可解决，即第 i 次循环时找到前 i 个字符所对应的节点，然后进行相应的操作。

可以定义一个数组 chr[N][SIZ],SIZ 表示字符集大小（如小写字母构成就是 26），该数组的每一行就代表一个节点。然后再定义一个数组 val[N]记录每个节点的属性,如该节点是否为结束节点或有几个单词在该节点结束等。

1、 插入操作

```
inline int idx(char c){//  计算符号对应的数字
    return c-'a';
}
void insert(char s[],int v){//插入字符串 s，权为 v
    int  len_s = strlen(s);
    int u = 0;
```

```
        for(int i=0;i<len_s;i++){
            int t = idx(s[i]);
            if(chk[u][t] == 0){
                memset(chk[size],0,sizeof(chk[size]));//申请一个新空间
                val[size] = 0; //权为1
                chk[u][t] = size++;//建立儿子
            }
            u = chk[u][t];
        }
        val[u] = v;//这个单词的最后位置
    }
```

2、 查询操作

```
    int find(char s[]){//字典里查找 s
        int u =0;//根节点
        for(i=0;i<strlen(s);i++){
            int t = idx(s[i]);//取出这个字母
            if(!chk[u][t]) return 0;//如果这个字母后没有后缀，返回未找到
            u = chk[u][t];//找儿子
        }
        return val[u];
    }
```

三、 实例

1、 模板 A2401 电话簿

题意：你需要判断其中是否有一个电话号码是另一个电话号码 的前缀。

【分析】模板题

在插入的同时判断是否有前缀，注意事项：多组数据，需要清零。

```
#include<bits/stdc++.h>
const int N=5e5+10;
char str[15];
int n,chr[N][10], tn;
bool val[N];
bool insert(){
    int cur = 0;
    bool tag = true;
    for(int i = 0; str[i] != '\0'; i ++){
        int nxt = str[i] - '0';
        if(!chr[cur][nxt]){
            tag = false;//扩展新节点标记，则无前缀
            chr[cur][nxt] = ++ tn;
        }
        cur = chr[cur][nxt];
        if(val[cur])//如果这个节点是一个单词结束点，返回真
            return true;
    }
```

```
    val[cur] = true;//标记结束标志
    return tag;
}

int main(){
    int t;
    scanf("%d", &t);
    while(t --){
        tn = 0;
        memset(chr, 0, sizeof(chr));
        memset(val, 0, sizeof(val));
        scanf("%d", &n);
        bool flag = false;
        while(n --){
            scanf("%s", str);
            if(!flag)//没有出现前缀才插入
                flag = insert();
        }
        if(flag)
            puts("YES");
        else
            puts("NO");
    }
    return 0;
}
```

2、P10054 秘密消息

题意:M 条信息，n 条密码，想知道每条密码有多少相同前缀的信息。

这个前缀长度必须等于密码和那条信息长度的较小者.（即互为前缀）

【解析】字典树，每个节点记录 2 个值，多少个单词经过了这个节点，多少个单词以这个节点作为结束。

```
#include<bits/stdc++.h>
using namespace std;
const int N=500500;
char c[N];
int chk[N][3],val[N],num[N];
int n,m,ans,size=1;

int fix(char s){
    return s-'0';
}

void insert(char s[]){
    int len=strlen(s);
    int u=0;
```

```
    for(int i=0;i<len;i++){
        int t=fix(s[i]);
        if(chk[u][t]==0){
            memset(chk[size],0,sizeof(chk[size]));
            chk[u][t]=size++;
        }
        u=chk[u][t];
        num[u]++;
    }
    val[u]++;
}

void query(char s[]){
    int len=strlen(s);
    int u=0;
    for(int i=0;i<len;i++){
        int t=fix(s[i]);
        ans+=val[u];
        if(chk[u][t]==0){
            return ;
        }
        u=chk[u][t];
    }
    ans+=num[u];
}

int main(){
    scanf("%d%d",&n,&m);
    int k;
    while(n--){
        scanf("%d",&k);
        for(int i=0;i<k;i++){
            scanf("%s",&c[i]);
        }
        insert(c);
    }
    while(m--){
        ans=0;
        scanf("%d",&k);
        for(int i=0;i<k;i++){
            scanf("%s",&c[i]);
        }
        query(c);
        printf("%d\n",ans);
```

```
    }
    return 0;
}
```

3、H1107 [SCOI2016]背单词

题意：记住一个单词的最小代价，如果他没有后缀代价为 x，否则为 x-y。

X 表示该单词记的序号，y 是最近的是该单词后缀的序号。

第一个条件无效。

【解析】贪心

根据 3 个条件，第一个条件肯定不能去满足，所以安排是其他单词后缀的单词先记则可避免条件一，后缀不好处理，转为前缀，建立前缀数 trie。

Trie 树建好后，按每个节点子树 size 少的优先顺序即可。

【算法过程】

1、建立字典树，记录单词结束标记

2、以字典树建图，以单词关系建图

3、统计每个单词为根的子树节点数量

4、以拓扑序贪心统计答案

4、P10050The XOR Largest Pair

题意：在给定的 N 个整数中 A1,A2,…,An 中选 2 个数异或运算，得到最大值是多少。

【解析】

朴素的做法：N^2 枚举二元组，然后得到答案。

正解：这是一个典型的 01Trie 树问题。我们将每个整数看着 32 位的二进制 01 串，将 N 个数从高位到低位依次插入到一个 01trie 中。

考虑插入第 i 个数，相当于在 trie 中进行依次检索，根据 xor 相同为 0，不同为 1 的特点，贪心的每次走与 Ai 当前位相反的指针，如果没有相反的节点则走相同的，这样就可以得到与 Ai 做 XOR 运算的最大 Aj。

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=3e6+10;
int n,tot=1,ans=0,x;
int val[maxn],ch[2][maxn];
void insert(int tmp){
    int u=1;
    for(int i=30;~i;i--){
        int c=(tmp>>i)&1;
        if(!ch[c][u]) ch[c][u]=++tot;
        u=ch[c][u];
    }
    val[u]=tmp;
}
int query(int tmp){
    int u=1;
    for(int i=30;~i; i--){
        int c=((tmp>>i)&1)==0;
        if(ch[c][u]) u=ch[c][u];
```

```
        else u=ch[!c][u];
    }
    return val[u];
}
int main(){
    scanf("%d",&n);
    tot=1;
    for(int i=1;i<=n;i++){
        scanf("%d",&x);
        ans=max(ans ,x^query(x));
        insert(x);
    }
    cout<<ans;
    return 0;
}
```

5、P10056The Xor-longest Path

题意：一个 N 个节点的树，每条边有比安全，选择 2 个点，使得两点路径的边权异或值最大。

【解析】设 D[x]表示根到 x 的路径边权 xor 值，这个值可以 dfs 得到。

于是发现 x 到 y 的路径异或就是 d[x]^d[y]。

问题变为：d[1]~d[n]选出 2 个异或最大值，与上道题一样。