> 题目需要我们判断一个较长字符串s中长度为i（i = 1, 2 ,...,len(s)）的前后缀是否相同,考虑利用字符串哈希判断字符串是否相同
>
> 可以通过预处理以时间复杂度为O（len（s））的程序完成对长度为i的前后缀对应哈希值的计算``

```
int pre[maxn], suf[maxn];   //前后缀的哈希值
int base = 7;
int len = strlen(s);
pre[1] = hash(s[1]);
suf[1] = hash(s[len]);
int mul = 1;
for (int i = 2; i <= len; i++){
    mul *= base;
    pre[i] = pre[i-1] + hash(s[i]) * mul;
    suf[i] = suf[i-1] * base + hash(s[len+1-i]);
}
```

> 在不考虑数据是否会爆`int`的情况下已经可以得到正确的字符串哈希值,判断后输出即可

```
for (int i = 1; i <= len; i++){
    if (pre[i] == suf[i])
        printf("%d ", i);
}
```

> 然而在题目给出的范围中，`int`甚至`long long`都会爆，所以采用一个与`base`互质的大数作为基数来取余保证数据正确性。

```
typedef unsigned long long ull;
const int base = 13;
const ull mod = 21237123151511;
//计算片段
pre[1] = hsah(s[1]);
suf[1] = hsah(s[len]);
ull mul = 1;
for (int i = 2; i <= len; i++) {
    mul = (mul * base) % mod;
    pre[i] = (pre[i-1] + (hsah(s[i]) * mul)) % mod;
    suf[i] = ((suf[i-1] * base) % mod + hsah(s[len+1-i])) % mod;
}
```

> AC代码：

```
#include <bits/stdc++.h>
using namespace std;
typedef unsigned long long ull;
const int MAXN = 1000010;
const int base = 13;
const ull mod = 21237123151511;

long long hsah(char x){
    if (x >= '0' && x <= '9')
        return x - '0';
```

```
    else if (islower(x))
        return 11 + x - 'a';
    else if (isupper(x))
        return 37 + x - 'A';
}

char s[MAXN];
int len;
ull pre[MAXN], suf[MAXN];
int main() {
    int t;
    scanf("%d", &t);
    while (t--) {
        memset(pre, 0, sizeof(pre));
        memset(suf, 0, sizeof(suf));
        memset(s, 0, sizeof(s));
        scanf("%s", s+1);
        len = strlen(s+1);
        pre[1] = hsah(s[1]);
        suf[1] = hsah(s[len]);
        ull mul = 1;
        for (int i = 2; i <= len; i++) {
            mul = (mul * base) % mod;
            pre[i] = (pre[i-1] + (hsah(s[i]) * mul)) % mod;
            suf[i] = ((suf[i-1] * base) % mod + hsah(s[len+1-i])) % mod;
        }

        for (int i = 1; i <= len; i++) {
            if (pre[i] == suf[i]) {
                printf("%d ", i);
            }
        }
        puts("");
    }

    return 0;
}
```