```
#include <iostream>

#include <cstdio>

#include <cstring>

#include <algorithm>

using namespace std;

#define mod 1000000007

#define base 233


long long hs1[100005];

long long hs2[100005];

long long pw[100005];

char a[100005];

char b[100005];


long long que_1(int l, int r)

{

    return hs1[r] - hs1[l - 1] * pw[r - l + 1];

}


long long que_2(int l, int r)

{

    return hs2[r] - hs2[l - 1] * pw[r - l + 1];
```

```
}
bool cmp(long long a, long long b)
{
    return a < b;
}

int main()
{
    while (scanf("%s%s", a + 1, b + 1) != EOF) {
        int len_a = strlen(a + 1);
        int len_b = strlen(b + 1);
        int len1, len2;

        pw[0] = 1;
        if (len_a >= len_b) {
            len1 = len_a, len2 = len_b;
            for (int i = 1; i <= len1; i++) {
                hs1[i] = (hs1[i - 1] * base + (a[i] - 'a'));
                pw[i] = (pw[i - 1] * base);
            }
            for (int i = 1; i <= len2; i++) {
                hs2[i] = (hs2[i - 1] * base + (b[i] - 'a'));
            }

        }
        else {
            len1 = len_b, len2 = len_a;
            for (int i = 1; i <= len1; i++) {
                hs1[i] = (hs1[i - 1] * base + (b[i] - ’a’));
                pw[i] = (pw[i - 1] * base);
            }
            for (int i = 1; i <= len2; i++) {
                hs2[i] = (hs2[i - 1] * base + (a[i] - ’a’));
            }
        }

        int l = 1, r = len2;   // 1长   2短
        int mid, cnt = 0, ans = 0;
        long long key[100005];
        long long judge;
        bool flag;
        while (l <= r) {
            mid = (l + r) >> 1;
            cnt = 0;
```

```
flag = 0;
for (int i = 1; i + mid - 1 <= len2; i++) {
   key[cnt] = que_2(i, i + mid - 1);
   cnt++;
}
sort(key, key + cnt, cmp);
for (int i = 1; i + mid - 1 <= len1; i++) {
   judge = que_1(i, i + mid - 1);
   if (binary_search(key, key + cnt, judge)) {
      flag = 1;
      break;
   }
   if (flag) {
      break;
   }
}

if (flag) {
   l = mid + 1;
   ans = mid;
}
else {
```

```
                r = mid - 1;
            }
        }
        printf("%d\n", ans);
    }

    return 0;
}
```