CSES seleneal1996 —

CSES Problem Set

Substring Order I

TASK | SUBMIT | RESULTS | STATISTICS | HACKING

Submission details

Task:	Substring Order I
Sender:	seleneal1996
Submission time:	2021-12-15 06:04:39
Language:	C++17
Status:	READY
Result:	ACCEPTED

Test results

test	verdict	time	
#1	ACCEPTED	0.01 s	»
#2	ACCEPTED	0.01 s	»
#3	ACCEPTED	0.03 s	»
#4	ACCEPTED	0.03 s	»
#5	ACCEPTED	0.09 s	»
#6	ACCEPTED	0.10 s	»
#7	ACCEPTED	0.03 s	»
#8	ACCEPTED	0.10 s	»
#9	ACCEPTED	0.05 s	»

Compiler report

```
input/code.cpp: In function 'void calc(int)':
input/code.cpp:56:26: warning: unused variable 'c
     for(const auto& [c, v] : node[u].nxt){
                          ^
input/code.cpp: In function 'int main()':
input/code.cpp:75:10: warning: ignoring return va
     scanf(" %s %lld", S, &K);
     ~~~~~^~~~~~~~~~~~~~~~~~~
```

Code

```
#include <bits/stdc++.h>

using namespace std;
typedef long long ll;
const int maxN = 1e5+5;

struct Node {
    ll dp;
```

String Algorithms

...

Counting Patterns	–
Pattern Positions	–
Distinct Substrings	–
Repeating Substring	–
String Functions	–
Substring Order I	✓
Substring Order II	–
Substring Distribution	✓

Your submissions

2021-12-15 06:04:39	✓
2021-12-15 06:00:48	✗

```
    int len, link;
    map<char,int> nxt;
} node[2*maxN];

vector<char> ans;
char S[maxN];
int N, sz, last;
ll K;

void init(){
    node[0].len = 0;
    node[0].link = -1;
    sz = 1;
    last = 0;
}

void extend(char c){
    int cur = sz++;
    node[cur].len = node[last].len + 1;
    int p = last;
    while(p != -1 && !node[p].nxt.count(c)){
        node[p].nxt[c] = cur;
        p = node[p].link;
    }
    if(p == -1){
        node[cur].link = 0;
    } else {
        int q = node[p].nxt[c];
        if(node[p].len + 1 == node[q].len){
            node[cur].link = q;
        } else {
            int clone = sz++;
            node[clone].len = node[p].len + 1
            node[clone].nxt = node[q].nxt;
            node[clone].link = node[q].link;
            while(p != -1 && node[p].nxt[c] =
                node[p].nxt[c] = clone;
                p = node[p].link;
            }
            node[q].link = node[cur].link = c
        }
    }
    last = cur;
}

void calc(int u = 0){
    node[u].dp = 1;
    for(const auto& [c, v] : node[u].nxt){
        if(!node[v].dp) calc(v);
        node[u].dp += node[v].dp;
    }
}

void dfs(int u, ll k){
    if(k < 0)   return;
    for(const auto& [c, v] : node[u].nxt){
        if(node[v].dp <= k) k -= node[v].dp;
        else {
            ans.push_back(c);
            dfs(v, k-1);
            return;
        }
    }
}

```

```
int main(){
    scanf(" %s %lld", S, &K);
    N = (int) strlen(S);

    init();
    for(int i = 0; i < N; i++)
        extend(S[i]);
    calc();

    dfs(0, K-1);
    int M = (int) ans.size();
    for(int i = 0; i < M; i++)
        printf("%c", ans[i]);
    printf("\n");
}
```

Share code to others

Test details

Test 1

Verdict: ACCEPTED

input

```
ababbaaabbabaabaabbaababa
10
```

correct output

```
aaabbabaab
```

user output

```
aaabbabaab
```

Test 2

Verdict: ACCEPTED

input

```
rmnxvouggsdgespsltsldcvkxtg
33
```

correct output

```
esps
```

user output

```
esps
```

Test 3

Verdict: ACCEPTED

input
```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...
```

correct output
```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...
```

user output
```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...
```

Test 4

Verdict: ACCEPTED

input
```
ababababababababababababababab...
```

correct output
```
bababababababababababababababa...
```

user output
```
bababababababababababababababa...
```

Test 5

Verdict: ACCEPTED

input
```
ababbbbaaaabaabbabaaabbaaaaaba...
```

correct output
```
aabbaababaaabbbaaaaaababaaaaba...
```

user output
```
aabbaababaaabbbaaaaaababaaaaba...
```

Test 6

Verdict: ACCEPTED

input
```
bslzdzbpuyxvovpeqxjhpnexwdheng...
```

correct output

```
fdzadhalzyzjstzcplofwhrvgshymp...
```

user output

```
fdzadhalzyzjstzcplofwhrvgshymp...
```

Test 7

Verdict: ACCEPTED

input

```
iybvzbbtkfqevdtjwqhezljzdkkjwi...
```

correct output

```
brdcxlfbsneugpmevkwmehndrzncoh...
```

user output

```
brdcxlfbsneugpmevkwmehndrzncoh...
```

Test 8

Verdict: ACCEPTED

input

```
gawaxmbhgoatjtxywopqckecliivyd...
```

correct output

```
phtjwgbtgbhslxxtprgbyppsnekyoy...
```

user output

```
phtjwgbtgbhslxxtprgbyppsnekyoy...
```

Test 9

Verdict: ACCEPTED

input

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...
```

correct output

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...
```

user output

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...
```