



Substring Distribution

TASK | SUBMIT | RESULTS | STATISTICS | HACKING

Submission details

Task:	Substring Distribution
Sender:	Rodry
Submission time:	2021-12-23 09:12:26
Language:	C++17
Status:	READY
Result:	ACCEPTED

Test results ▲

test	verdict	time	
#1	ACCEPTED	0.01 s	<u>>></u>
#2	ACCEPTED	0.01 s	<u>>></u>
#3	ACCEPTED	0.12 s	<u>>></u>
#4	ACCEPTED	0.12 s	<u>>></u>
#5	ACCEPTED	0.10 s	<u>>></u>
#6	ACCEPTED	0.12 s	<u>>></u>
#7	ACCEPTED	0.12 s	<u>>></u>
#8	ACCEPTED	0.13 s	<u>>></u>
#9	ACCEPTED	0.13 s	<u>>></u>

Code ▲

1	<pre>#include<bits stdc++.h=""></bits></pre>
2	
3	<pre>using namespace std;</pre>
4	

String Algorithms

..

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

Your submissions

2021-12-23 09:12:26



```
5 \mid const int mxN = 1e5+5;
 6 int sa[mxN], pos[mxN], tmp[mxN], lcp[mxN];
7 int gap, N;
8 string S;
10 | bool comp(int x, int y) {
       if (pos[x] != pos[y])
11
12
           return pos[x] < pos[y];</pre>
13
       x += gap;
14
       y += gap;
15
       return (x < N && y < N)? pos[x] < pos[y] : x > y;
16 }
17
18 void suffix() {
19
       for (int i = 0; i < N; i++)
20
           sa[i] = i, pos[i] = S[i];
21
22
       for (gap = 1;; gap <<= 1) {</pre>
23
           sort(sa, sa+N, comp);
24
           for (int i = 0; i < N-1; i++)
25
                tmp[i+1] = tmp[i] + comp(sa[i], sa[i+1]);
26
           for (int i = 0; i < N; i++)
27
                pos[sa[i]] = tmp[i];
28
           if (tmp[N - 1] == N - 1)
29
                break;
30
31
32
33
   void build lcp() {
34
       for (int i = 0, k = 0; i < N; i++) if (pos[i] != N-1) {
35
           int j = sa[pos[i] + 1];
36
           while (S[i + k] == S[j + k])
37
                k++;
38
           lcp[pos[i]] = k;
39
           if (k) k--;
40
41 | }
42
43 int pre[mxN];
44
45 int main(){
46
47
       cin>>S; N = S.size();
48
       suffix();
49
       build_lcp();
50
       int prev = 0;
```

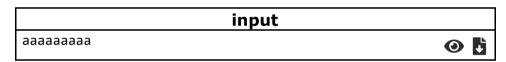
```
51
       for (int i = 0; i < N; i++) {</pre>
52
           pre[prev + 1]++;
           pre[N - sa[i] + 1]--;
53
54
           prev = lcp[i];
55
56
       for (int i = 1; i <= N; i++) {</pre>
           cout << pre[i] << ' ';
57
58
           pre[i+1] += pre[i];
59
60
61
       return 0;
62 }
```

Share code to others

Test details ▲

Test 1

Verdict: ACCEPTED



	correct output
111111111	⊚

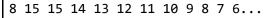
	user output	
1 1 1 1 1 1 1 1 1	• • • • • • • • • • • • • • • • • • •	į

Test 2

Verdict: ACCEPTED

	input	
dfgfglrisgdsdgskd		O

correct output





user output		
8 15 15 14 13 12 11 10 9 8 7 6	O	

Test 3

Verdict: ACCEPTED

input	
aaaaaaaaaaaaaaaaaaaaaa	0

correct output		
2 3 4 5 6 7 8 9 10 11 12 13 14	•	

user output	
2 3 4 5 6 7 8 9 10 11 12 13 14	()

Test 4

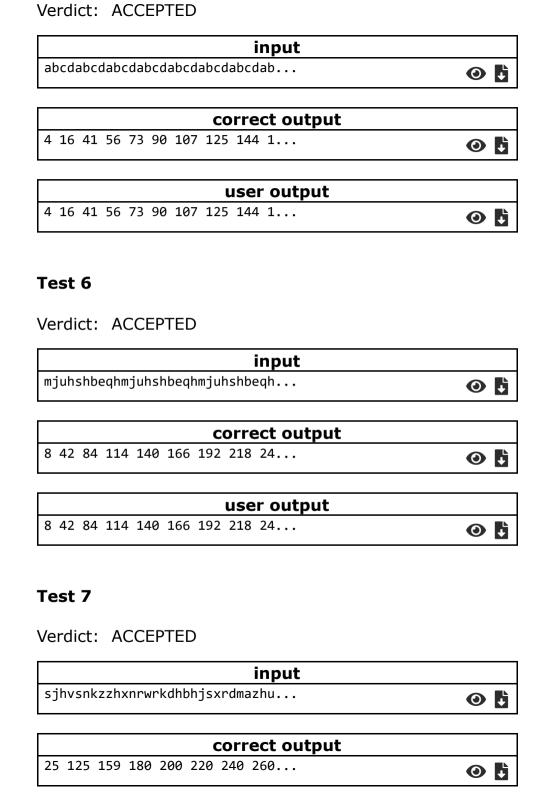
Verdict: ACCEPTED

input	
abababababababababababab	•

correct output		
2 4 8 10 12 14 16 18 20	22 24	©

user output	
2 4 8 10 12 14 16 18 20 22 24	©

Test 5



user output25 125 159 180 200 220 240 260...

Test 8

Verdict: ACCEPTED

input	
wovxwnsrmgvjcvwepxviodztwrkszm	•

correct output		
26 676 7670 10103 10279 10336	Ø	

user output	
26 676 7670 10103 10279 10336	O

Test 9

Verdict: ACCEPTED

input	
aaaaaaaaaaaaaaaaaaaaaa	Ø

correct output		
2 3 4 5 6 7 8 9 10 11 12 13 14	©	

user output	
2 3 4 5 6 7 8 9 10 11 12 13 14	② L