

Strings.

Material de referencia.

Índice

1. Arreglo de sufijos.	2
1.1. Implementación	2

1. Arreglo de sufijos.

Consideremos un string s . Un arreglo de sufijos de s es un arreglo de enteros que guarda las posiciones iniciales de los sufijos de s en orden lexicográfico.

1.1. Implementación

Complejidad: $O(|s| \log |s|)$.

```
1  #include <iostream>
2  #include <algorithm>
3  using namespace std;
4
5  #define maxn 100000 //Longitud maxima del string.
6
7  string word;          //String.
8  int n, SuffixArray[maxn]; //Arreglo de sufijos.
9
10 int rnk[maxn][2], bucket[maxn]; //Rango (SuffixArray) y Cubeta (RaxixSort).
11 int tempSA[maxn], tempRA[maxn][2]; //Arreglos temporales.
12
13 //Ordena de acuerdo a los rangos.
14 void RadixSort() {
15     int M = max(n, 256);
16     for (int k = 1; k >= 0; --k) {
17         fill_n(bucket, M, 0);
18
19         for (int i = 0; i < n; ++i)
20             ++bucket[rnk[i][k]];
21         for (int i = 1; i < M; ++i)
22             bucket[i] += bucket[i - 1];
23
24         for (int i = n - 1; i >= 0; --i) {
25             int nxt_id = --bucket[rnk[i][k]];
26             tempSA[nxt_id] = SuffixArray[i];
27             tempRA[nxt_id][0] = rnk[i][0];
28             tempRA[nxt_id][1] = rnk[i][1];
29         }
30         for (int i = 0; i < n; ++i) {
31             SuffixArray[i] = tempSA[i];
32             rnk[i][0] = tempRA[i][0];
33             rnk[i][1] = tempRA[i][1];
34         }
35     }
36 }
37
38 //Construye el arreglo de sufijos.
39 void buildSA() {
40     n = word.size();
41
42     for (int i = 0; i < n; ++i) {
43         SuffixArray[i] = i;
44         rnk[i][0] = word[i];
45     }
46     RadixSort();
47 }
```

```

48     for (int k = 1; k < n; k *= 2) {
49         int curr = 0, prev = rnk[0][0];
50         rnk[0][0] = curr;
51         tempSA[SuffixArray[0]] = 0;
52
53         for (int i = 1; i < n; ++i) {
54             if (rnk[i][0] != prev || rnk[i][1] != rnk[i - 1][1])
55                 ++curr;
56             prev = rnk[i][0];
57             rnk[i][0] = curr;
58             tempSA[SuffixArray[i]] = i;
59         }
60
61         for (int i = 0; i < n; ++i) {
62             int nxt_id = SuffixArray[i] + k;
63             rnk[i][1] = (nxt_id < n) ? rnk[tempSA[nxt_id]][0] : 0;
64         }
65         RadixSort();
66     }
67 }
68
69 int main() {
70     ios_base::sync_with_stdio(0); cin.tie();
71     cin >> word;
72
73     buildSA();
74     for (int i = 0; i < n; ++i) {
75         cout << SuffixArray[i] << ' ';
76         for (int j = SuffixArray[i]; j < n; ++j)
77             cout << word[j];
78         cout << '\n';
79     }
80
81     return 0;
82 }

```

Entrada	Salida
banana	5 a 3 ana 1 anana 0 banana 4 na 2 nana