

```

1  struct hash_struct{
2
3      int n;
4      string str;
5
6      vi p, m;
7      vector<vi> pre;
8      vector<vi> pow_m;
9
10     hash_struct(){}
11
12     hash_struct(string _str){
13         str = _str;
14         n = str.size();
15         init();
16         build();
17     }
18
19     bool is_prime(int x){
20         for(ll i = 2; i*i <= x; i++) if(x%i==0){
21             return false;
22         }
23         return true;
24     }
25
26     void init(){
27
28         p = {rand(1e5, 2e5), rand(1e9, 2e9)};
29         m = {rand(30, 50), rand(50, 100)};
30
31         for(int j = 0; j < 2; j++){
32             while(!is_prime(p[j])) p[j]++;
33             while(!is_prime(m[j])) m[j]++;
34         }
35
36         pow_m.resize(n, vi(2));
37
38         pow_m[0][0] = pow_m[0][1] = 1;
39         for(int i = 1; i < n; i++)
40             for(int j = 0; j < 2; j++){
41                 pow_m[i][j] = (pow_m[i-1][j] *111* m[j])%p[j];
42             }
43     }
44
45     void build(){
46
47         pre.resize(n);
48
49         vi cval(2);
50         for(int i = 0; i < n; i++){
51             for(int j = 0; j < 2; j++){
52                 cval[j] = ((cval[j] *111* m[j])%p[j] + (str[i]-'a'+1))%p[j];
53             }
54             pre[i] = cval;
55         }
56     }
57
58     vi query(int l, int r){
59         vi ret = pre[r];
60         if(l) for(int j = 0; j < 2; j++){
61             ret[j] = (ret[j] - (pre[l-1][j] *111* pow_m[r-l+1][j])%p[j] + p[j])%p[j];
62         }
63         return ret;
64     }
65
66     };

```