

```

1  /*** Z Algo *****/
2  #include <bits/stdc++.h>
3  using namespace std;
4  int z[100002];
5  vector<int> v;
6  struct info {
7      int prop, sum;
8  } tree[100002 * 3]; //sum ????? ????????? ??
9  void update(int node, int b, int e, int i, int j,
10 int x) {
11     if (i > e || j < b)
12         return;
13     if (b >= i && e <= j) //????? ??????
14     {
15         tree[node].sum += ((e - b + 1) * x); //????
16         tree[node].prop += x; //????? ?????????
17         return;
18     }
19     int Left = node * 2;
20     int Right = (node * 2) + 1;
21     int mid = (b + e) / 2;
22     update(Left, b, mid, i, j, x);
23     update(Right, mid + 1, e, i, j, x);
24     tree[node].sum = tree[Left].sum +
25     tree[Right].sum + (e - b + 1) * tree[node].prop;
26     //????? ?????? ?????? ?????? ?????? ??????
27     //????? ?????? ?????? ?????? ?????? ??????
28     int query(int node, int b, int e, int i, int j, int
29     carry = 0) {
30         if (i > e || j < b)
31             return 0;
32         if (b >= i and e <= j)
33             return tree[node].sum + carry * (e - b +
34             1); //????? ?? ?????? ?????? ?????? ?????? ??????
35             //????? ?????? ??????
36         int Left = node << 1;
37         int Right = (node << 1) + 1;
38         int mid = (b + e) >> 1;
39         int p1 = query(Left, b, mid, i, j, carry +
40         tree[node].prop); //????? ?????? ??????
41         int p2 = query(Right, mid + 1, e, i, j, carry +
42         tree[node].prop);
43         return p1 + p2;
44     }
45     int main()
46     {
47         int L = 0, R = 0, a, b, c, d, e, f, j, T, n;
48         char s[100002];
49         cin >> T;
50         for (j = 1; j <= T; j++)
51         {
52             memset(z, 0, sizeof(z));
53             memset(tree, 0, sizeof(tree));
54             v.clear();

```

```

54     L=0;
55     R=0;
56     scanf("%s",&s);
57     n=strlen(s);
58
59
60     for (int i = 1; i < n; i++) {
61         if (i > R) {
62             L = R = i;
63             while (R < n && s[R-L] == s[R]) R++;
64             z[i] = R-L; R--;
65         } else {
66             int k = i-L;
67             if (z[k] < R-i+1) z[i] = z[k];
68             else {
69                 L = i;
70                 while (R < n && s[R-L] == s[R]) R++;
71                 z[i] = R-L; R--;
72             }
73         }
74     }
75     z[0]=n;
76     ff=0;
77     for(int i=0;i<n;i++)
78     {
79         if(z[i]!=0)
80         {
81             update(1,1,100001,1,z[i],1);
82         }
83     }
84
85
86     scanf("%d",&b);
87     for(int i=1;i<=b;i++)
88     {
89         scanf("%d",&c);
90         if(c!=0)
91         {
92
93             d=query(1,1,100001,c,c,0);
94             v.push_back(d);
95         }
96         else
97             v.push_back(n);
98     }
99
100     d=v.size();
101     for(int i=0;i<d;i++)
102     {
103         printf("%d ",v[i]);
104     }
105     printf("\n");
106
107
108 }
109 return 0;
110 }

```