Aizu University (team: mo3zimanju)

```
1: ##################################################
2: ################## dijkstra.cpp ##################
3: ##################################################
4:
5: #include<bits/stdc++.h>
6: using namespace std;
7:
8: #define MOD 1000000007
9: //#define MOD 998244353
10: #define INF 1000000010
11: #define EPS 1e-9
12:
13: #define debug(x) cout<<x<<endl;
14: #define repi(i,x,n) for(int i=x;i<n;i++)
15: #define rep(i,n) repi(i,0,n)
16: #define lp(i,n) repi(i,0,n)
17: #define repn(i,n) for(int i=n;i>=0;i--)
18: #define int long long
19: #define endl "\n"
20: #define N 100000
21: #define F first
22: #define S second
23:
24: typedef pair<int,int> PII;
25: typedef pair<int,string> PIS;
26: typedef pair<string,int> PSI;
27:
28: struct Node{
29:   vector<PII> edges;//first=cost,second=index
30:
31:   bool visit;
32:   int cost;
33: };
34:
35: Node mp[N];
36:
37: void dijkstra(int st,int n){
38:   rep(i,n){
39:     mp[i].visit=false;
40:     mp[i].cost= -1;
41:   }
42:   priority_queue<PII>  q;
43:   mp[st].cost=0;
44:   rep(i,mp[st].edges.size() ){
45:     q.push({-mp[st].edges[i].F,mp[st].edges[i].S});
46:   }
47:   mp[st].visit=true;
48:   while(!q.empty() ){
49:     PII p=q.top();
50:     q.pop();
51:     if(! mp[p.S].visit){
52:       mp[p.S].cost= -p.F;
53:       mp[p.S].visit=true;
54:       rep(i,mp[p.S].edges.size() ){
55:         q.push(make_pair(-mp[p.S].edges[i].F-mp[p.S].cost,mp[p.S].edges[i].S));
56:       }
57:     }
58:   }
59: }
60:
61:
62:
63:
64: ##################################################
65: ################ prime_factor.cpp ################
66: ##################################################
67:
68: #include<bits/stdc++.h>
69: using namespace std;
70:
```

```
#define int long long
#define endl "\n"

map<int,int> prime_factor(int n){
  map<int,int> mp;
  for(int i=2;i*i<=n;i+=2){
    while(n%i==0){
      mp[i]++;
      n/=i;
    }
    if(i==2) i--;
  }
  if(n!=1) mp[n]=1;
  return mp;
}




##################################################
################## template.cpp ##################
##################################################

#include<bits/stdc++.h>
using namespace std;
#define MOD 1000000007
//#define MOD 998244353
#define INF 1000000010
#define EPS 1e-9
#define F first
#define S second

#define debug(x) cout<<x<<endl;
#define repi(i,x,n) for(int i=x;i<n;i++)
#define rep(i,n) repi(i,0,n)
#define lp(i,n) repi(i,0,n)
#define repn(i,n) for(int i=n;i>=0;i--)
#define int long long
#define endl "\n"

typedef pair<int,int> PII;
typedef pair<int,string> PIS;
typedef pair<string,int> PSI;


signed main(){
  cin.tie(0);
  ios::sync_with_stdio(false);


  return 0;
}

##################################################
################## divisor.cpp ###################
##################################################

#include<bits/stdc++.h>
using namespace std;
#define MOD 1000000007
#define INF 1000000010
#define EPS 1e-9
#define fst first
#define scd second

#define debug(x) cout<<x<<endl;
#define repi(i,x,n) for(int i=x;i<n;i++)
#define rep(i,n) repi(i,0,n)
#define lp(i,n) repi(i,0,n)
```

```
141: #define repn(i,n) for(int i=n;i>=0;i--)
142: #define int long long
143: #define endl "\n"
144:
145: vector<int> divisor(int n){
146:   vector<int> v;
147:   for(int i=1;i*i<=n;i++){
148:     if(n%i==0){
149:       v.push_back(i);
150:       if(i*i!=n) v.push_back(n/i);
151:     }
152:   }
153:   sort(v.begin(),v.end() );
154:   return v;
155: }
156:
157:
158:
159:
160:
161: ////////////////////////////////////////////////////
162: ////////////////// draftcode.cpp ///////////////////
163: ////////////////////////////////////////////////////
164:
165: #include<bits/stdc++.h>
166: using namespace std;
167:
168:
169: typedef string::const_iterator State;
170: class ParseError {};
171:
172: int number(State &begin){
173:   int num=0;
174:   while(isdigit(*begin)){
175:     num*=10;
176:     num+= *begin - '0';
177:     begin++;
178:   }
179:   return num;
180: }
181:
182:
183:
184:
185: ////////////////////////////////////////////////////
186: ////////////////// combinaton.cpp //////////////////
187: ////////////////////////////////////////////////////
188:
189: #include<bits/stdc++.h>
190: using namespace std;
191: #define MOD 1000000007
192: #define BIG 1000000010
193: #define EPS 1e-9
194: #define fst first
195: #define scd second
196:
197: #define debug(x) cout<<x<<endl;
198: #define repi(i,x,n) for(int i=x;i<n;i++)
199: #define rep(i,n) repi(i,0,n)
200: #define repn(i,n) for(int i=n;i>=0;i--)
201: #define int long long
202:
203: const int MAX=500000;
204:
205: int fac[MAX],finv[MAX],inv[MAX];
206:
207: void t(){
208:   fac[0]=fac[1]=1;
209:   finv[0]=finv[1]=1;
210:   inv[1]=1;
```

```
  211:   repi(i,2,MAX){
  212:     fac[i]=fac[i-1]*i%MOD;
  213:     inv[i]=MOD-inv[MOD%i]*(MOD/i)%MOD;
  214:     finv[i]=finv[i-1]*inv[i]%MOD;
  215:   }
  216: }
  217:
  218: int calc(int n,int k){
  219:   if(n<k) return 0;
  220:   if(n<0 || k<0) return 0;
  221:   return fac[n]*(finv[k]*finv[n-k]%MOD)%MOD;
  222: }
  223:
  224:
  225:
  226: signed main(){
  227:   int n,m;
  228:   cin>>n>>m;
  229:   t();
  230:   cout<<calc(n,m)<<endl;
  231:   return 0;
  232: }
  233:
  234:
  235: ##################################################
  236: ################## zalgorithm.cpp ################
  237: ##################################################
  238:
  239: #include<bits/stdc++.h>
  240: using namespace std;
  241: #define MOD 1000000007
  242: #define BIG 1000000010
  243: #define EPS 1e-9
  244: #define fst first
  245: #define scd second
  246:
  247: #define debug(x) cout<<x<<endl;
  248: #define repi(i,x,n) for(int i=x;i<n;i++)
  249: #define rep(i,n) repi(i,0,n)
  250: #define repn(i,n) for(int i=n;i>=0;i--)
  251: #define int long long
  252: #define endl "\n"
  253:
  254: int A[300000],B[300000];
  255:
  256:
  257: //Aã\201«å\205¥ã\202\213ã\201®ã\201¯ã\200\201Sã\201®iæ\226\207å-\227ç\233®ã\201
\213ã\202\211ä½\225æ\226\207å-\227ã\201\214tã\201®æ\216¥é -è¾\236ã\201¨ä¸\200è\207´ã\201
\231ã\202\213ã\201\213
  258:
  259: void zalgorithm(string S,string t){
  260:   int i = 0, j = 0;
  261:   while (i+j < S.size() && j < t.size() && t[j] == S[i+j]) ++j;
  262:   A[i] = j;
  263:   if (j != 0){
  264:     int k = 1;
  265:     while (i+k < S.size() && k+A[k] < j) A[i+k] = A[k], ++k;
  266:   }
  267:   i=1;j=0;
  268:   while (i < S.size()) {
  269:     while (i+j < S.size() && j < t.size() && t[j] == S[i+j]) ++j;
  270:     A[i] = j;
  271:     if (j == 0) { ++i; continue;}
  272:     int k = 1;
  273:     while (i+k < S.size() && k+A[k] < j) A[i+k] = A[k], ++k;
  274:     i += k; j -= k;
  275:   }
  276: }
  277:
  278: void rzalgorithm(string S,string t){
```

```
279:   int i = 0, j = 0;
280:   reverse(t.begin(),t.end() );
281:   reverse(S.begin(),S.end() );
282:   // cout<<t<<endl;
283:   while (i+j < S.size() && j < t.size() && t[j] == S[i+j]) ++j;
284:   B[i] = j;
285:   if (j != 0){
286:     int k = 1;
287:     while (i+k < S.size() && k+B[k] < j) B[i+k] = B[k], ++k;
288:   }
289:   i=1;j=0;
290:   while (i < S.size()) {
291:     while (i+j < S.size() && j < t.size() && t[j] == S[i+j]) ++j;
292:     B[i] = j;
293:     if (j == 0) { ++i; continue;}
294:     int k = 1;
295:     while (i+k < S.size() && k+B[k] < j) B[i+k] = B[k], ++k;
296:     i += k; j -= k;
297:   }
298: }
299:
300:
301: ####################################################
302: ################### prime.cpp ######################
303: ####################################################
304:
305: #include<bits/stdc++.h>
306: using namespace std;
307: #define MOD 1000000007
308: #define BIG 1000000010
309: #define repi(i,x,n) for(int i=x;i<n;i++)
310: #define rep(i,n) repi(i,0,n)
311: #define repn(i,n) for(int i=n;i>=0;i--)
312: typedef long long int ll;
313:
314: #define N 100000
315:
316: int prime[N];
317:
318: void eratosthenes(){
319:   rep(i,N) prime[i]=1;
320:   prime[0]=prime[1]=0;
321:   rep(i,n){
322:     if(prime[i]){
323:       for(int j=i+i;j<N;j+=i) prime[j]=0;
324:     }
325:   }
326: }
327:
328: bool primeNumber(int n){
329:   if(n < 2) return false;
330:   else{
331:     for(int i = 2; i * i <= n; i++){
332:       if(n % i == 0) return false;
333:     }
334:     return true;
335:   }
336: }
337:
338:
339: ####################################################
340: ################### power.cpp ######################
341: ####################################################
342:
343: #include<bits/stdc++.h>
344: using namespace std;
345: #define MOD 1000000007
346: //#define MOD 998244353
347: #define INF 1000000010
348: #define EPS 1e-9
```

```
349: #define F first
350: #define S second
351:
352: #define debug(x) cout<<x<<endl;
353: #define repi(i,x,n) for(int i=x;i<n;i++)
354: #define rep(i,n) repi(i,0,n)
355: #define lp(i,n) repi(i,0,n)
356: #define repn(i,n) for(int i=n;i>=0;i--)
357: #define int long long
358: #define endl "\n"
359:
360: typedef pair<int,int> PII;
361: typedef pair<int,string> PIS;
362: typedef pair<string,int> PSI;
363:
364: int power(int n,int m){
365:   int now=2;
366:   int num=1;
367:   while(n!=0){
368:     //cout<<n<<" "<<m<<endl;
369:     if(n%2!=0){
370:       num=(num*m)%MOD;
371:
372:     }
373:     n/=2;
374:     m=(m*m)%MOD;
375:   }
376:   return num;
377: }
378:
379:
380: ################################################
381: ################ rollinghash.cpp ################
382: ################################################
383:
384: #include<bits/stdc++.h>
385: using namespace std;
386: using Int = long long;
387: template<typename T1,typename T2> inline void chmin(T1 &a,T2 b){if(a>b) a=b;}
388: template<typename T1,typename T2> inline void chmax(T1 &a,T2 b){if(a<b) a=b;}
389: //BEGIN CUT HERE
390: template<typename T,T MOD,T B>
391: struct RollingHash{
392:   vector<T> hash,p;
393:   RollingHash(){}
394:   RollingHash(const string &s){
395:     int n=s.size();
396:     hash.assign(n+1,0);
397:     p.assign(n+1,1);
398:     for(int i=0;i<n;i++){
399:       hash[i+1]=(hash[i]*B+s[i])%MOD;
400:       p[i+1]=p[i]*B%MOD;
401:     }
402:   }
403:   //S[l, r)
404:   T find(int l,int r){
405:     T res=hash[r]+MOD-hash[l]*p[r-l]%MOD;
406:     return res>=MOD?res-MOD:res;
407:   }
408: };
409: //END CUT HERE
410: //INSERT ABOVE HERE
411: signed main(){
412:   cin.tie(0);
413:   ios::sync_with_stdio(0);
414:   string t,p;
415:   cin>>t>>p;
416:   using ll = long long;
417:   const ll MOD = 1e9+7;
418:   const ll B = 1777771;
```

```
  419:   using RH = RollingHash<ll, MOD, B>;
  420:   RH rt(t),rp(p);
  421:   for(int i=0;i<(int)t.size()-(int)p.size()+1;i++){
  422:     if(rt.find(i,i+p.size())==rp.find(0,p.size())) cout<<i<<"\n";
  423:   }
  424:   cout<<flush;
  425:   return 0;
  426: }
  427:
  428:
  429: ##################################################
  430: ############### unionfindtree.cpp ################
  431: ##################################################
  432:
  433:
  434:
  435: struct UnionFind{
  436:   vector<int> data;
  437:
  438:   UnionFind(int N){
  439:     data.assign(N,-1);
  440:   }
  441:
  442:  bool unite(int x, int y) {
  443:     x = find(x), y = find(y);
  444:     if(x == y) return (false);
  445:     if(data[x] > data[y]) swap(x, y);
  446:     data[x] += data[y];
  447:     data[y] = x;
  448:     return (true);
  449:   }
  450:
  451:   int find(int k) {
  452:     if(data[k] < 0) return (k);
  453:     return (data[k] = find(data[k]));
  454:   }
  455:
  456:   int size(int k) {
  457:     return (-data[find(k)]);
  458:   }
  459: };
  460:
  461:
  462: ##################################################
  463: ############### ivu_Bellmanford.cpp ###############
  464: ##################################################
  465:
  466: #include <bits/stdc++.h>
  467: using namespace std;
  468: #define FOR(i,l,r) for(int i=(l); i<(r); i++)
  469: #define REP(i,n) FOR(i,0,n)
  470: #define endl "\n"
  471: #define debug(x) cout<<x<<endl;
  472: typedef long long ll;
  473: static const int INF = 1e9+7;
  474:
  475: struct Edge {
  476:   int to,cost;  //to æ\216¥ç¶\232å\205\210ã\200\200, cost é\207\215ã\201¿
  477:   Edge(int to, int cost) : to(to), cost(cost) {}  // ã\202³ã\203³ã\202¹ã\203\210ã
\203©ã\202¯ã\202¿
  478: };
  479:
  480: typedef vector<vector<Edge> > Edge_List;
  481: Edge_List graph;
  482:
  483: vector<int> dist; //æ\234\200ç\237-è•\235é\233¢
  484:
  485: // æ\210»ã\202\212å\200¤ã\201\214trueã\201ªã\202\211è² ã\201®é\226\211è•¯ã\202\222
å\220«ã\202\200
  486: bool bellman_ford(int n, int s) {
```

```
  487:     //nã\201¯é \202ç\202¹æ\225°,sã\201¯å§\213ã\201¾ã\202\212ã\201®é \202ç\202¹
  488:
  489:     dist = vector<int>(n,INF);
  490:     dist[s] = 0; // é\226\213å§\213ç\202¹ã\201®è•\235é\233¢ã\201¯0
  491:     for (int i=0; i<n; i++) {
  492:         for (int v=0; v<n; v++) {
  493:             for (int k=0; k<graph[v].size(); k++) {
  494:                 Edge e = graph[v][k];
  495:                 if (dist[v] != INF && dist[e.to] > dist[v] + e.cost) {
  496:                     dist[e.to] = dist[v] + e.cost;
  497:                     if (i == n - 1) return true; // nå\233\236ç\233®ã\201«ã\202
\202æ\233´æ\226°ã\201\214ã\201\202ã\202\213ã\201ªã\202\211ã\201°è² ã\201®é\226\211è•¯ã
\201\214å-\230å\234¨ã\201\227ã\201¦ã\201\204ã\202\213
  498:                 }
  499:             }
  500:         }
  501:     }
  502:   return false;
  503: }
  504:
  505:
  506: /*int main() {
  507:   int n, m;
  508:   cin >> n >> m;
  509:   graph = Edge_List(n);
  510:
  511:   for (int i=0; i<m; i++) {
  512:     int from,to,cost;
  513:     cin >>from>>to>>cost;
  514:     graph[from].push_back(Edge(to, cost));
  515:   }
  516:
  517:   bellman_ford(n,0);
  518:
  519:   for (int i=1; i<n; i++) {
  520:     if (dist[i] != INF)
  521:       cout<<"0ã\201\213ã\202\211"<< i <<"ã\201¸ã\201®ã\202³ã\202¹ã\203\210ã\201¯:
"<<dist[i]<<endl;
  522:   }
  523:
  524: }
  525: */
  526:
  527:
  528: ////////////////////////////////////////////////
  529: ////////////////// suffix_array.cpp ////////////////
  530: ////////////////////////////////////////////////
  531:
  532: #include<bits/stdc++.h>
  533: using namespace std;
  534: #define MOD 1000000007
  535: #define INF 1000000010
  536: #define EPS 1e-9
  537: #define fst first
  538: #define scd second
  539:
  540: #define debug(x) cout<<x<<endl;
  541: #define repi(i,x,n) for(int i=x;i<n;i++)
  542: #define rep(i,n) repi(i,0,n)
  543: #define lp(i,n) repi(i,0,n)
  544: #define repn(i,n) for(int i=n;i>=0;i--)
  545: #define int long long
  546: #define endl "\n"
  547:
  548:
  549: struct SuffixArray {
  550:   vector< int > SA;
  551:   const string s;
  552:
  553:   SuffixArray(const string &str) : s(str) {
```

```
  554:     SA.resize(s.size());
  555:     iota(begin(SA), end(SA), 0);
  556:     sort(begin(SA), end(SA), [&](int a, int b) {
  557:       return s[a] == s[b] ? a > b : s[a] < s[b];
  558:     });
  559:     vector< int > classes(s.size()), c(s.begin(), s.end()), cnt(s.size());
  560:     for(int len = 1; len < s.size(); len <<= 1) {
  561:       for(int i = 0; i < s.size(); i++) {
  562:         if(i > 0 && c[SA[i - 1]] == c[SA[i]] && SA[i - 1] + len < s.size() && c[SA
[i - 1] + len / 2] == c[SA[i] + len / 2]) {
  563:           classes[SA[i]] = classes[SA[i - 1]];
  564:         } else {
  565:           classes[SA[i]] = i;
  566:         }
  567:       }
  568:       iota(begin(cnt), end(cnt), 0);
  569:       copy(begin(SA), end(SA), begin(c));
  570:       for(int i = 0; i < s.size(); i++) {
  571:         int s1 = c[i] - len;
  572:         if(s1 >= 0) SA[cnt[classes[s1]]++] = s1;
  573:       }
  574:       classes.swap(c);
  575:     }
  576:   }
  577:
  578:   int operator[](int k) const {
  579:     return SA[k];
  580:   }
  581:
  582:   size_t size() const {
  583:     return s.size();
  584:   }
  585:
  586:   bool lt_substr(const string &t, int si = 0, int ti = 0) {
  587:     int sn = (int) s.size(), tn = (int) t.size();
  588:     while(si < sn && ti < tn) {
  589:       if(s[si] < t[ti]) return true;
  590:       if(s[si] > t[ti]) return false;
  591:       ++si, ++ti;
  592:     }
  593:     return si >= sn && ti < tn;
  594:   }
  595:
  596:   int lower_bound(const string &t) {
  597:     int low = -1, high = (int) SA.size();
  598:     while(high - low > 1) {
  599:       int mid = (low + high) / 2;
  600:       if(lt_substr(t, SA[mid])) low = mid;
  601:       else high = mid;
  602:     }
  603:     return high;
  604:   }
  605:
  606:   pair< int, int > lower_upper_bound(string &t) {
  607:     int idx = lower_bound(t);
  608:     int low = idx - 1, high = (int) SA.size();
  609:     t.back()++;
  610:     while(high - low > 1) {
  611:       int mid = (low + high) / 2;
  612:       if(lt_substr(t, SA[mid])) low = mid;
  613:       else high = mid;
  614:     }
  615:     t.back()--;
  616:     return {idx, high};
  617:   }
  618:
  619:   void output() {
  620:     for(int i = 0; i < size(); i++) {
  621:       cout << i << ": " << s.substr(SA[i]) << endl;
  622:     }
```

```
623:   }
624: };
625:
626: struct LongestCommonPrefixArray {
627:   const SuffixArray &SA;
628:   vector< int > LCP, rank;
629:
630:   LongestCommonPrefixArray(const SuffixArray &SA) : SA(SA), LCP(SA.size()) {
631:     rank.resize(SA.size());
632:     for(int i = 0; i < SA.size(); i++) {
633:       rank[SA[i]] = i;
634:     }
635:     for(int i = 0, h = 0; i < SA.size(); i++) {
636:       if(rank[i] + 1 < SA.size()) {
637:         for(int j = SA[rank[i] + 1]; max(i, j) + h < SA.size() && SA.s[i + h] == S
A.s[j + h]; ++h);
638:         LCP[rank[i] + 1] = h;
639:         if(h > 0) --h;
640:       }
641:     }
642:   }
643:
644:   int operator[](int k) const {
645:     return LCP[k];
646:   }
647:
648:   size_t size() const {
649:     return LCP.size();
650:   }
651:
652:   void output() {
653:     for(int i = 0; i < size(); i++) {
654:       cout << i << ": " << LCP[i] << " " << SA.s.substr(SA[i]) << endl;
655:     }
656:   }
657: };
658:
659:
660:
661:
662: signed main(){
663:   cin.tie(0);
664:   ios::sync_with_stdio(false);
665:   string s;
666:   cin>>s;
667:   SuffixArray sa(s);
668:   int n;
669:   cin>>n;
670:   sa.output();
671:   return 0;
672: }
673:
674:
```