//#include <br< th=""><th> Suffix Array </th><th></th><th><pre>#define fri(a,b)</pre></th><th>for(int i=a;i<=b;i++)</th></br<>	 Suffix Array 		<pre>#define fri(a,b)</pre>	for(int i=a;i<=b;i++)
#define fr1(a,b)	j		<pre>#define frj(a,b)</pre>	for(int j=a;j<=b;j++)
ios_base::sync_with_stdio(0); cin.tie(0);				
#include <algorithm></algorithm>	//#define _			
#include #include <ctype></ctype>	<pre>ios_base::sync_with_stdio(0);cin.tie(0);</pre>		<pre>#define frin(a,b)</pre>	for(int i=a;i>=b;i)
#include cctype>	<pre>#include <algorithm></algorithm></pre>		<pre>#define frjn(a,b)</pre>	for(int j=a;j>=b;j)
#include <cmath> #include <cstdio> #define frn(i,a,b) for(i=a;i>=b;i) #include <cstdio> #define fwrite #include <cstdio> #define fwrite #include <cstdio> #define inf #include <cstring> #define inf #include <fstream> #define inpow(a,x,y) int i; a=x;fri(2,y) a*=x #include <cstream> #define makeint(n,s) istringstream(s)>>n #include #define makeint(n,s) istringstream(s)>>n #include <map> #define mod 100000007 #include <queu> #define ISS istringstream #include <set> #define OS</set></queu></map></cstream></fstream></cstring></cstdio></cstdio></cstdio></cstdio></cmath>	<pre>#include <bitset></bitset></pre>			for (int $k=a; k>=b; k$)
#include <cstdio></cstdio>	<pre>#include <cctype></cctype></pre>		<pre>#define frln(a,b)</pre>	
#include <cstdlib> #include <cstring> #include <fstream> #include <fstream> #include <fstream> #include <instream> #include <instream< ii=""> #includ</instream<></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></instream></fstream></fstream></fstream></cstring></cstdlib>				for(i=a;i>=b;i)
#include <cstring> #include <fstream> #include <fstream> #include <instream> #include <iinstream> #include <iinstream #include="" <iinstrea<="" <iinstream="" td=""><td colspan="2"><pre>#include <cstdio></cstdio></pre></td><td colspan="2">#define fwrite</td></iinstream></iinstream></iinstream></iinstream></iinstream></iinstream></iinstream></iinstream></iinstream></iinstream></iinstream></iinstream></instream></fstream></fstream></cstring>	<pre>#include <cstdio></cstdio></pre>		#define fwrite	
#include <fstream></fstream>	<pre>#include <cstdlib></cstdlib></pre>			w",stdout)
#include <iostream> #include #define makeint(n,s)</iostream>	<pre>#include <cstring></cstring></pre>		#define inf	(1e9)
#include #include <map> #include <map> #define mod #define ISS istringstream #define l1 long long #include <set> #define OSS ostringstream #define OSS ostringstream #define OSS ostringstream #include <stack> #define PI 3.141592653589793 #include <string> #define pi #define pi #define pi #define pi #define pp pop_back using namespace std; #define PRE using namespace std; #define print1(a) #define PRE using namespace std; #define BE(a) #define BE(a) #define OSS #define rev(a) #define rev(a) #define chng(a,b) #define chng(a,b) #define CROSS(a,b,c,d) ((b.x-a.x)*(d.y-c.y)-(d.x- (x)*(b.y-a.y)) #define ERR #define ERR #define SIZE(s) #define ERR #define ERR #define SIZE(s) #define SIZE(s) #define SIZE(s) #define SIZE(a) #define SIZE(s) #define ERR #define SIZE(s) #define SIZE(s) #define ERR #define SIZE(s) #define SIZE(s) #define ERR #define SIZE(s) #de</string></stack></set></map></map>	<pre>#include <fstream></fstream></pre>		<pre>#define inpow(a,x,y)</pre>	int i; $a=x$; fri(2,y) $a*=x$
#include <map></map>	<pre>#include <iostream></iostream></pre>		<pre>#define makeint(n,s)</pre>	istringstream(s)>>n
#include <queue> #include <queue> #include <queue> #include <queue> #include <sstream> #include <sstream> #include <sstream> #include <string> #include <string> #include <vetor> #include <ctime> #include <string> #include <string #include="" <string=""> #include <string> #include <string> #include <string> #include <string> #include <string #id<="" #idefine="" #include="" <striing="" <string="" bri="" fore(i,a)="" pri="" sii="" td=""><td><pre>#include <list></list></pre></td><td></td><td>#define mod</td><td>100000007</td></string></string></string></string></string></string></string></string></string></string></string></string></string></string></string></string></string></string></string></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></ctime></vetor></string></string></sstream></sstream></sstream></queue></queue></queue></queue>	<pre>#include <list></list></pre>		#define mod	100000007
#include <set> #define oo (1<30) #include <sstream> #define OS ostringstream #include <stack> #define PI 3.141592653589793 #include <string> #define PI 3.141592653589793 #include <vector> #define pp pop_back using namespace std; #define PRE 1e-8 #define PRE 1e-8 #define pp pop_back #define print1(a) cout<<a<endl cout<<a="" cout<<a<endl=""><a<endl cout<<a=""><a<endl cout<<a=""><a><a><a><a><a><a><a><a><a><a><a><a></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></a<endl></vector></string></stack></sstream></set>	<pre>#include <map></map></pre>		#define ISS	istringstream
#include <sstream> #include <sstream> #include <stack> #define DSS</stack></sstream></sstream>	#include <queue></queue>		#define ll	long long
#include <stack></stack>	#include <set></set>		#define oo	(1<<30)
#include <string> #include <vector> #include <vector> #include <vector> #include <ctime> #define pi</ctime></vector></vector></vector></string>	<pre>#include <sstream></sstream></pre>		#define OSS	ostringstream
#include <vector> #include <ctime> using namespace std; #define pp #define pri</ctime></vector>	<pre>#include <stack></stack></pre>		#define pb	push back
#include <ctime> using namespace std; #define pRE 1e-8 #define print1(a) cout<<a<<ndd style="text-align: left;">#define print1(a) cout<<a<<nd style="text-align: left;">#define print1(a) cout<<a<<nd style="text-align: left;">#define print2(a,b) cout<<a<<nd style="text-align: left;">#define print2(a,b) cout<<a<<nd style="text-align: left;">#define print3(a,b,c) cout<<a<<nt style="text-align: left;">#define print3(a,b,c) cout<<a>#define print3(a,b,c) cout<<a< td=""><td><pre>#include <string></string></pre></td><td></td><td>#define PI</td><td>$3.14\overline{1}592653589793$</td></a<></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nt></a<<nd></a<<nd></a<<nd></a<<nd></a<<nd></a<<nd></a<<nd></a<<nd></a<<ndd></ctime>	<pre>#include <string></string></pre>		#define PI	$3.14\overline{1}592653589793$
<pre>using namespace std; #define PRE #define print1(a) cout<<a<<endl "<<b<<endl="" #define="" ((b.x-a.x)*(d.y-c.y)-(d.x-c.x)*(b.y-a.y))="" -="" 0)="" 0.5="" :="" <="" ?="" a="" all(a,b,c)="" be(a)="" clr(y,z)="" cout<<a<<"="" cross(a,b,c,d)="" err<="" for(int="" i="(a" memset(y,z,sizeof(y))="" print2(a,b)="" rev(a)="" reverse(be(a));="" round(i,a)="" td=""><td><pre>#include <vector></vector></pre></td><td></td><td>#define pi</td><td>(2*acos(0))</td></a<<endl></pre>	<pre>#include <vector></vector></pre>		#define pi	(2*acos(0))
#define print1(a) cout< <a<endl "<<b="" "<<b<endl="" #define="" all(a,b,c)="" cout<<a<"="" cout<<a<<"="" for(int="" i="0;I<b;I++)" print2(a,b)="" print3(a,b,c)="">" "<<b<endl "<<b="" #define="" cout<<a<<"="" print3(a,b,c)="">" "<" " " " " " " " " " " " " " " " " " "</b<endl></a<endl>	<pre>#include <ctime></ctime></pre>		#define pp	pop back
<pre>#define all(a,b,c) for(int I=0;I<b;i++) "<<b<="" #define="" cout<<a<"="" print2(a,b)="">endl a[I] = c #define print3(a,b,c) cout<<a<" "<<b="" "<<b<<endl="" #define="" cout<<a<"="" print3(a,b,c)="">"<"<"< "< "< "< "<<c>endl #define chng(a,b) a^=b^=a^=b; #define rev(a) reverse(BE(a)); #define clr(y,z) memset(y,z,sizeof(y)) #define round(i,a) i = (a < 0) ? a - 0.5 : #define cntbit(mask)builtin_popcount(mask) a + 0.5; #define CROSS(a,b,c,d) ((b.x-a.x)*(d.y-c.y)-(d.x- #define SI set<int>:iterator c.x)*(b.y-a.y)) #define EQ(a,b) (fabs(a-b)<err) #define="" ((int)s.size())="" err<="" size(s)="" td=""><td>using namespace std;</td><td></td><td>#define PRE</td><td>1e-8</td></err)></int></c></a<"></b;i++)></pre>	using namespace std;		#define PRE	1e-8
<pre>a[I] = c</pre>			<pre>#define print1(a)</pre>	cout< <a<<endl< td=""></a<<endl<>
<pre>#define BE(a)</pre>	<pre>#define all(a,b,c)</pre>	for(int I=0;I <b;i++)< td=""><td><pre>#define print2(a,b)</pre></td><td>cout<<a<<" "<<b<<endl<="" td=""></a<<"></td></b;i++)<>	<pre>#define print2(a,b)</pre>	cout< <a<<" "<<b<<endl<="" td=""></a<<">
#define chng(a,b)	a[I] = c		<pre>#define print3(a,b,c)</pre>	cout< <a<<" "<<b<<"<="" td=""></a<<">
<pre>#define chng(a,b)</pre>	#define BE(a)	a.begin(),a.end()	"< <c<endl< td=""><td></td></c<endl<>	
#define cntbit(mask)builtin_popcount(mask) a + 0.5; #define CROSS(a,b,c,d) ((b.x-a.x)*(d.y-c.y)-(d.x- c.x)*(b.y-a.y)) #define SI set <int>:iterator #define EQ(a,b) (fabs(a-b)<err) #define="" ((a)*(a))="" ((int)s.size())="" 50005="" err="" for(i="a;i<=b;i++)" for(typeof((a).begin())i="(a).begin();i!=(a).end();" fore(i,a)="" fr(i,a,b)="" i++)="" le-5="" saja(a)="" size(s)="" sort(be(a))="" sqr(a)="" sz="" td="" typeof((b).begin())<="" typing(j,b)=""><td>#define chng(a,b)</td><td>a^=b^=a^=b;</td><td><pre>#define rev(a)</pre></td><td>reverse(BE(a));</td></err)></int>	#define chng(a,b)	a^=b^=a^=b;	<pre>#define rev(a)</pre>	reverse(BE(a));
#define CROSS(a,b,c,d) ((b.x-a.x)*(d.y-c.y)-(d.x- #define SI set <int>:iterator #define EQ(a,b) (fabs(a-b)<err) #define="" ((a)*(a))="" ((int)s.size())="" 1e-5="" 50005="" err="" for(i="a;i<=b;i++)" for(typeof((a).begin())i="(a).begin();i!=(a).end();" fore(i,a)="" fr(i,a,b)="" i++)="" saja(a)="" size(s)="" sort(be(a))="" sqr(a)="" sz="" td="" typeof((b).begin())<="" typing(j,b)=""><td>#define clr(y,z)</td><td><pre>memset(y,z,sizeof(y))</pre></td><td><pre>#define round(i,a)</pre></td><td>i = (a < 0) ? a - 0.5:</td></err)></int>	#define clr(y,z)	<pre>memset(y,z,sizeof(y))</pre>	<pre>#define round(i,a)</pre>	i = (a < 0) ? a - 0.5:
#define EQ(a,b)	#define cntbit(mask)	builtin popcount(mask)	a + 0.5;	
#define EQ(a,b)	<pre>#define CROSS(a,b,c,d)</pre>	((b.x-a.x)*(d.y-c.y)-(d.x-	#define SI	set <int></int>
#define ERR 1e-5 #define saja(a) sort(BE(a)) #define FORE(i,a) #define sqr(a) ((a)*(a)) for(typeof((a).begin())i=(a).begin();i!=(a).end(); #define SZ 50005 i++) #define fr(i,a,b) for(i=a;i<=b;i++) #define typing(j,b) typeof((b).begin())	c.x) * (b.y-a.y))		#define SII	set <int>::iterator</int>
<pre>#define FORE(i,a)</pre>	#define EQ(a,b)	(fabs(a-b) <err)< td=""><td>#define SIZE(s)</td><td>((int)s.size())</td></err)<>	#define SIZE(s)	((int)s.size())
for(typeof((a).begin())i=(a).begin();i!=(a).end(); #define SZ 50005 i++) #define SZ1 55 #define fr(i,a,b) for(i=a;i<=b;i++) #define typing(j,b) typeof((b).begin())	#define ERR	1e-5	#define saja(a)	sort(BE(a))
i++) #define SZ1 55 #define fr(i,a,b) for(i=a;i<=b;i++) #define typing(j,b) typeof((b).begin())	#define FORE(i,a)		<pre>#define sqr(a)</pre>	((a)*(a))
<pre>i++) #define SZ1 55 #define fr(i,a,b) for(i=a;i<=b;i++) #define typing(j,b) typeof((b).begin())</pre>	<pre>for(typeof((a).begin())</pre>	i=(a).begin();i!=(a).end();	#define SZ	50005
	i++)		#define SZ1	55
	<pre>#define fr(i,a,b)</pre>	for(i=a;i<=b;i++)	<pre>#define typing(j,b)</pre>	<pre>typeof((b).begin())</pre>
<pre>#define fread</pre>			j=(b).begin();	
freepen ("input.txt"."r".stdin) #define VD vector <double></double>	freopen("input.txt", "r", stdin)		#define VD	vector <double></double>

```
#define VI
                        vector<int>
#define VLL
                        vector<long long>
                                                               return bucket;
#define VS
                        vector<string>
string str;
                                                           void SetRank(int n)
int revSA[SZ], SA[SZ];
int cnt[SZ] , nxt[SZ];
                                                               for (int i = 0; i < n; i = nxt[i])
bool bh[SZ],b2h[SZ];
int lcp[SZ];
                                                                   cnt[i] = 0;
                                                                   for(int j =i ; j<nxt[i] ; j++)</pre>
bool cmp(int i,int j)
                                                                        revSA[SA[j]] = i;
    return str[i] < str[j];</pre>
                                                               return;
void sortFirstChar(int n)
    /// sort for the first char ...
                                                           void findNewRank(int l,int r,int step)
    for(int i =0; i<n; i++)
        SA[i] = i;
                                                               for (int j = l; j < r; j + +)
    sort(SA,SA+n ,cmp);
                                                                    int pre = SA[j] - step;
                                                                    if(pre>=0)
   ///indentify the bucket ......
    for(int i=0; i<n; i++)
                                                                        int head = revSA[pre];
                                                                        revSA[pre] = head+cnt[head]++;
        bh[i] = (i==0 | | str[SA[i]]!=str[SA[i-
                                                                        b2h[revSA[pre]] = true;
1]]);
       b2h[i] = false;
    return;
                                                               return;
int CountBucket(int n)
                                                           void findNewBucket(int l,int r,int step)
    int bucket = 0;
                                                               for (int j = 1; j < r; j + +)
    for(int i =0 , j; i < n ; i = j)
                                                                    int pre = SA[j] - step;
        j = i+1;
                                                                   if(pre>=0 && b2h[revSA[pre]])
        while (j < n \& \& bh[j] == false) j++;
                                                                        for (int k = revSA[pre]+1; b2h[k] &&
        nxt[i] = i;
        bucket++;
                                                           !bh[k]; k++) b2h[k] = false;
```

```
lcp[k] = 0;
                                                                       continue;
    return;
                                                                   int j = SA[k-1];
void buildSA(int n)
                                                                   while(str[i+len]==str[j+len]) len++;
                                                                   lcp[k] = len;
   ///start sorting in logn step ...
                                                                   if(len) len--;
    sortFirstChar(n);
    for(int h =1; h<n; h<<=1)
                                                               return;
        if(CountBucket(n) == n) break;
        SetRank(n);
                                                           void printSA()
        /// cause n-h suffix must be sorted
        b2h[revSA[n-h]] = true;
                                                               for(int i=0;i<SIZE(str);i++) printf("%d</pre>
        cnt[revSA[n-h]]++;
                                                           ", SA[i]);
                                                               puts("");
        for (int i = 0; i < n; i = nxt[i])
                                                               for(int i=1;i<SIZE(str);i++) printf("%d</pre>
                                                           ", lcp[i]);
            findNewRank(i,nxt[i] , h);
                                                               puts("");
            findNewBucket(i , nxt[i] , h);
                                                               return ;
        ///set the new sorted suffix array ...
                                                           int main()
        for(int i =0 ; i<n ; i++)
                                                               int n,p,q;
            SA[revSA[i]] = i;
                                                               int tcase, cas=1;
            bh[i] = b2h[i]; //new bucket ....
                                                               scanf(" %d", &tcase);
                                                               while(tcase--)
                                                               {
                                                                   cin>>str;
    return;
                                                           //
                                                                     cin>>p>>q;
                                                                   buildSA(SIZE(str));
void buildLCP(int n)
                                                                   buildLCP(SIZE(str));
   int len = 0;
                                                                   printSA();
                                                           //
   for (int i = 0; i < n; i + +)
                                                                    int sol = findSol(p,q,SIZE(str));
        revSA[SA[i]] = i;
                                                           //
                                                                     printf("Case %d: %d\n", cas++, sol);
   for(int i =0; i< n; i++)
                                                               return 0;
       int k = revSA[i];
       if(k==0)
```

- Trie tree using array

```
/*
TRIE tree:
>> The complexity of TRIE is: n.
>> It takes a huge amount of words and then it can
search the word with efficient
complexity.
>> Input is: some words which you want to include in
your dictionary then give
words to search.
.. Output is: For every searching word, either YES if
the word exists or NO if
the word does not exist.
#include <iostream>
#include <cstdio>
#include <cstring>
#include <algorithm>
#include <string>
#define sz 200005
#define clr(abc,z) memset(abc,z,sizeof(abc))
using namespace std;
// TRIE starts array
#define trie sz 26
struct node{
   bool ending;
   int next[trie sz];
   node()
       ending = false;
        for (int i = 0; i<trie sz; i++) next[i] = 0;
}data[sz];
int counter=1;
void insert in trie(char *str, int len)
```

```
int n = 0;
    for (int i = 0; i < len; i++)
        int now = str[i]-'a';
        if(data[n].next[now] == 0)
            data[n].next[now] = counter++;
        n=data[n].next[now];
    data[n].ending=true;
bool search in trie(char *str, int len)
    int n = 0;
    for (int i = 0; i < len; i++)
        int now = str[i]-'a';
        if(data[n].next[now] == 0) return false;
        n=data[n].next[now];
    return data[n].ending;
bool delete from trie() // this is for memset only
    clr(data,0);
// TRIE ends using array
int main()
    int n;
    char s[sz];
    puts("How many words in dictionary?");
    scanf("%d", &n);
    for (int i = 0; i < n; i + +)
            scanf("%s", s);
```

Trie tree using pointer

```
TRIE tree:
>> The complexity of TRIE is: n.
>> It takes a huge amount of words and then it can
search the word with efficient
complexity.
>> Input is: some words which you want to include in
your dictionary then give
words to search.
.. Output is: For every searching word, either YES if
the word exists or NO if
the word does not exist.
* /
#include <iostream>
#include <cstdio>
#include <cstring>
#include <algorithm>
#include <string>
#define sz 2000005
using namespace std;
```

```
// TRIE starts using pointer
#define trie sz 26
struct node{
    bool ending;
    node *next[trie sz];
    node()
        ending = false;
        for (int i = 0; i<trie sz; i++) next[i] = NULL;
} * root;
void insert in_trie(char *str, int len)
    node *cur = root;
    for (int i = 0; i < len; i++)
        int now = str[i]-'a';
        if(cur->next[now] ==NULL)
            cur->next[now]=new node();
        cur=cur->next[now];
    cur->ending=true;
bool search in trie(char *str, int len)
    node *cur=root;
    for (int i = 0; i < len; i++)
        int now = str[i]-'a';
        if(cur->next[now] == NULL) return false;
        cur=cur->next[now];
    return cur->ending;
bool delete_from_trie(node *cur) // this is for memset,
it should be called by delete from trie(root) from main
    for (int i = 0; i<trie sz; i++)
        if(cur->next[i]!=NULL) delete from trie(cur-
>next[i]);
```

```
delete(cur);
                                                                 which can be both suffix and prefix of that string.
// TRIE ends using pointer
                                                                 >> Input is: a very large size (approx. 100000
                                                                 character) of string.
                                                                 >> It will return the length of largest possible string
int main()
                                                                 which can be both suffix
    root = new node();
                                                                 and prefix.
                                                                 * /
    int n;
                                                                 #include <iostream>
    char s[sz];
    puts ("How many words in dictionary?");
                                                                 #include <cstdio>
    scanf("%d", &n);
                                                                 #include <cstring>
                                                                 #include <algorithm>
    for (int i = 0; i < n; i + +)
                                                                 #include <string>
                                                                 #define sz 2000005
            scanf("%s", s);
                                                                 using namespace std;
            insert in trie(s, strlen(s));
                                                                 //KMP starts
                                                                 char line[sz];
    puts("How many searches from dictionary?");
                                                                 int overlap[sz];
    scanf("%d", &n);
                                                                 int kmp()
    for (int i = 0; i < n; i + +)
                                                                      int len = strlen(line), v;
                                                                      overlap[0] = 0;
            scanf("%s", s);
                                                                      for (int i = 1; i < len; i++)
search in trie(s,strlen(s)) == true?printf("YES\n"):printf
                                                                          v = overlap[i-1];
("NO\n");
                                                                          while(line[v]!=line[i] && v>0)
                                                                              v = overlap[v-1];
    delete from trie(root);
                                                                          if(line[v] == line[i]) overlap[i] = v+1;
    return 0;
                                                                          else overlap[i] = 0;
                                                                      return overlap[len-1];
KMP
                                                                 //KMP ends
/*
                                                                 int main()
Knuth Morris Pattern (KMP):
                                                                      int t, n, m, cas=1;
>> The complexity of KMP is: n+m. where n is the length
                                                                      cout<<"write the string: "<<endl;</pre>
of the string and m
                                                                      cin>>line;
varies string to string and it can be at most n-1.
                                                                      cout<<"Max Length of kmp = "<<kmp()<<endl;</pre>
>> It takes a very large line of input and find the
highest length of a string
                                                                     return 0;
```

- KMP

```
#include <bits/stdc++.h>
#define all(a,b,c)
                          for(int I=0;I<b;I++)</pre>
                                                    a[I] = c
#define BE(a)
                          a.begin(),a.end()
#define chnq(a,b)
                          a^=b^=a^=b;
\#define clr(y,z)
                          memset(y,z,sizeof(y))
#define cntbit(mask)
                             builtin popcount (mask)
                         ((\overline{b}.x-a.x)*(d.y-c.y)-(d.x-
#define CROSS(a,b,c,d)
c.x)*(b.y-a.y))
#define EO(a,b)
                          (fabs(a-b) < ERR)
#define ERR
                          1e-5
#define FORE(i,a)
for (typeof((a).begin())i=(a).begin();i!=(a).end();i++)
#define fr(i,a,b)
                          for(i=a;i<=b;i++)
#define fread
                          freopen("input.txt", "r", stdin)
#define fri(a,b)
                          for (int i=a; i \le b; i++)
#define frj(a,b)
                          for (int j=a; j <=b; j++)
#define frk(a,b)
                          for (int k=a; k \le b; k++)
#define frl(a,b)
                          for (int l=a; 1 <= b; 1++)
                          for(int i=a; i>=b; i--)
#define frin(a,b)
#define frjn(a,b)
                          for (int j=a; j>=b; j--)
#define frkn(a,b)
                          for (int k=a; k>=b; k--)
#define frln(a,b)
                          for (int l=a; l>=b; l--)
#define frn(i,a,b)
                          for(i=a;i>=b;i--)
#define fwrite
                          freopen("output.txt", "w", stdout)
#define inf
#define print1(a)
                          cout<<a<<endl
#define print2(a,b)
                          cout << a << " " << b << endl
                          cout<<a<<" "<<b<<" "<<c<endl
#define print3(a,b,c)
#define rev(a)
                          reverse (BE(a));
#define round(i,a)
                          i = (a < 0) ? a - 0.5 : a +
0.5;
#define SI
                          set<int>
#define SII
                          set<int>::iterator
#define SIZE(s)
                          ((int)s.size())
#define saja(a)
                         sort(BE(a))
#define sqr(a)
                          ((a) * (a))
#define SZ
                          50005
#define SZ1
                          55
```

```
#define typing(j,b)
                         typeof((b).begin())
j=(b).begin();
#define VD
                         vector<double>
#define VI
                         vector<int>
#define VLL
                         vector<long long>
#define VS
                         vector<string>
VI adj[SZ];//only adj should be cleared
int col[SZ],low[SZ],tim[SZ],timer;
int group id[SZ], n, m, components; //components=number of
components group id = which node belongs to which node
stack<int>S;
void scc(int u)
    int i, v, tem;
    col[u]=1;
    low[u]=tim[u]=timer++;
    S.push(u);
    fr(i, 0, SIZE(adj[u])-1)
        v=adj[u][i];
        if(col[v]==1)
            low[u] = min(low[u], tim[v]);
        else if (col[v]==0)
            scc(v);
            low[u] = min(low[u], low[v]);
    //SCC checking...
    if(low[u] == tim[u])
        do
            tem=S.top();S.pop();
            group id[tem]=components;
            col[tem]=2; //Completed...
        }while(tem!=u);
        components++;
}
```

int TarjanSCC() //some change may be required here

```
//this link between groups no.....
   int i;
                                                                         fr(i,0,components-1)
   timer=components=0;
   clr(col, 0);
                                                                             fr(j,0,SIZE(nadj[i])-1)
   while(!S.empty()) S.pop();
   fr(i,0,n-1) if (col[i]==0) scc(i);
                                                                                 u=i;
   return components;
                                                                                 v=nadj[i][j];
                                                                                 print2(u,v);
VI nadj[SZ];//new adjcency list after SCC(DAG)
void MakeNewDAG Graph()
                                                                     return 0;
   int i,j,u,v;
                                                                 /*
   fr(i,0,components-1) nadj[i].clear();
                                                                Input:
                                                                 8 14
    fr(i,0,n-1)
                                                                0 1
                                                                1 2
        fr(j,0,SIZE(adj[i])-1)
                                                                1 5
                                                                1 4
            u=group id[i];
                                                                 2 6
           v=group id[adj[i][j]];
                                                                 2 3
           if(u!=v)
                                                                 3 2
                nadj[u].pb(v);
                                                                 3 7
                                                                 4 5
                                                                 5 6
                                                                7 6
                                                                7 3
                                                                 6 5
int main()
                                                                4 0
   int i,j,t,cas=0,u,v,ans;
                                                                Output:
                                                                 Total Groups: 3
   while(scanf("%d %d",&n,&m) == 2)
                                                                NewGraphLinkUsingSCC: this graph is directed acyclic
                                                                 graph:
        fr(i,0,n-1) adj[i].clear();
                                                                1 0
                                                                1 0
        fr(i,1,m)
        {
                                                                2 1
            scanf("%d %d",&u,&v);
                                                                2 0
            adj[u].pb(v);
                                                                2 0
        TarjanSCC();
        printf("Total Groups: %d\n",components);
                                                                Another Input:
       MakeNewDAG Graph();
                                                                6 6
       printf("NewGraphLinkUsingSCC: this graph is
                                                                0 1
directed acyclic graph:\n");
                                                                1 2
```

```
2 1
3 4
4 5
5 4
Total Groups: 4
NewGraphLinkUsingSCC: this graph is directed acyclic graph:
1     0
3     2
*/
```

Mat Expo

```
#include <bits/stdc++.h>
```

```
using namespace std;
#define print1(a)
                     cout<<a<<endl
#define print2(a,b) cout<<a<<" "<<b<<endl
#define print3(a,b,c) cout<<a<<" "<<b<<" "<<c<endl
#define oo
                    (1 << 30)
#define PI
                    3.141592653589793
#define pi
                    2*acos(0)
#define ERR
                    1e-5
#define PRE
                    1e-8
#define SZ(a)
                    (int)a.size()
#define LL
                    long long
#define ISS
                    istringstream
#define OSS
                    ostringstream
#define VS
                    vector<string>
#define VI
                    vector<int>
#define VD
                    vector<double>
#define VLL
                    vector<long long>
#define SII
                    set<int>::iterator
#define SI
                    set<int>
#define mem(a,b)
                    memset(a,b,sizeof(a))
#define fr(i,a,b)
                    for(i=a;i<=b;i++)
#define frn(i,a,b) for(i=a;i>=b;i--)
#define fri(a,b)
                    for(i=a;i<=b;i++)
#define frin(a,b) for(i=a;i>=b;i--)
#define frj(a,b)
                    for(j=a;j<=b;j++)
#define frjn(a,b) for(j=a;j>=b;j--)
#define frk(a,b)
                    for (k=a; k \le b; k++)
\#define frkn(a,b) for(k=a;k>=b;k--)
```

```
#define frl(a,b)
                    for(l=a; <=b; l++)
#define frln(a,b)
                    for(l=a; l>=b; l--)
#define EQ(a,b)
                     (fabs(a-b) < ERR)
#define all(a,b,c) for(int I=0;I<b;I++)</pre>
                                             a[I] = c
\#define CROSS(a,b,c,d) ((b.x-a.x)*(d.y-c.y)-(d.x-
c.x)*(b.y-a.y))
#define sqr(a)
                     ((a)*(a))
#define FORE(i,a)
for (typeof((a).begin())i=(a).begin();i!=(a).end();i++)
#define BE(a)
                     a.begin(),a.end()
#define rev(a)
                    reverse (BE(a));
#define sorta(a)
                    sort(BE(a))
#define pb
                    push back
#define popb
                    pop back
#define round(i,a) i = (a < 0)? a - 0.5: a + 0.5;
#define makeint(n,s) istringstream(s)>>n
#define countbit(mask) builtin popcount(musk)
                    1000000007
#define mod
struct matrix{
     LL x[6][6];
matrix base, zero;
matrix matmult(matrix &a, matrix &b, int n)//m*n and n*r
matrix //1 based
    matrix ret;
    int i, j, k;
    fr(i,1,n)
    fr(j,1,n)
        ret.x[i][j]=0;
        fr(k,1,n)
ret.x[i][j]=ret.x[i][j]+(a.x[i][k]*b.x[k][j])%mod; //we
can reduce complexity here
        ret.x[i][j]%=mod;
    return ret;
matrix bigmod(matrix b,long long p,int n) //have to pass
{
```

```
matrix xx=zero;
    int i;
                                                                 >> The complexity of LCS is: n square.
    fr(i,1,n) xx.x[i][i]=1;
                                                                 >> It takes two strings and finds a new string which is
    matrix power=b;
                                                                 the longest common subsequence
                                                                 of the previous two strings. then output the numbering
    while(p)
                                                                 for two string in seperate
        if ((p&1)==1) xx=matmult(xx, power, n);
                                                                 lines.
        power=matmult(power, power, n);
        p/=2;
                                                                 >> Input is: Two strings.
    return xx;
                                                                 #include <bits/stdc++.h>
                                                                 #define ios base::sync with stdio(0);cin.tie(0);
                                                                 #define sz 100
int main()
                                                                 #define pb(a) push back(a)
    int t, cas=0;
                                                                 #define pp pop bac\overline{k}()
    cin>>t;
                                                                 #define ll long long
    int k;
                                                                 #define fread freopen("input.txt","r",stdin)
                                                                 #define fwrite freopen("output.txt","w",stdout)
    long long n;
    while(t--)
                                                                 \#define inf (1<<30-1)
                                                                 #define clr(abc,z) memset(abc,z,sizeof(abc))
                                                                 #define PI acos(-1)
        cin>>n>>k;
        printf("Case %d: ",++cas);
                                                                 using namespace std;
        base.x[1][1]=1;
                                                                 int magnitude[sz][sz];
        base.x[1][4]=2;
        base.x[4][1]=3;
                                                                 char direction[sz][sz];
        if(n \le 1)
                                                                 void LCS(string X, string Y)
            print1(n);
            continue;
                                                                     int m = X.size(), n = Y.size();
        matrix ans=bigmod(base, n-1, k+2); //n-number of
                                                                     for (int i = 1; i <= m; i++)
baseconditions+1
        print1(ans.x[1][1]);
                                                                          for (int j = 1; j <= n; j ++ )
    return 0;
                                                                              if(X[i-1]==Y[j-1])
                                                                                  magnitude[i][j] = magnitude[i-1][j-1]+1;
                                                                                  direction[i][j] = 'D'; //'D' denotes its
- LCS
                                                                 came from diagonal
                                                                              else if (magnitude[i-1][j]>=magnitude[i][j-
                                                                 11)
                                                                              {
LCS - Longest Common Subsequence:
```

```
magnitude[i][j] = magnitude[i-1][j];
                                                                          c.pop();
                direction[i][j] = 'U';//'U' denotes its
came from up
                                                                      cout<<endl<<"Positions in first string : ";</pre>
                                                                      while(!p.empty())
            else
                                                                          cout << p. top() << " ";
                magnitude[i][j] = magnitude[i][j-1];
                                                                          p.pop();
                direction[i][j] = 'L';//'D' denotes its
came from left
                                                                      cout<<endl<<"Positions in second string : ";</pre>
                                                                      while(!q.empty())
                                                                          cout<<q.top()<<" ";
                                                                          q.pop();
                                                                      cout << endl;
    return;
                                                                      return 0;
int main()
                                                                  ACCGGTCGAGTGCGCGGAAGCCGGCCGAA
                                                                  GTCGTTCGGAATGCCGTTGCTCTGTAAA
    string a, b;
                                                                  * /
    clr(magnitude,0);
    stack<int>p,q;
                                                                  nCr
    stack<char>c;
    int len;
    cin>>a>>b;
    int m = a.size(), n =b.size();
    LCS(a,b);
                                                                  Finding nCr:
    len = magnitude[m][n];
    while (m&&n)
                                                                  >> The complexity of bubble sort is: unknown.
                                                                  >> It works with two loops.
        if (direction[m][n] == 'D')
                                                                  >> For a given n and r, we can find the value of nCr
                                                                  recursively using the formula
            p.push(m);
                                                                  nCr = (n-1)Cr + (n-1)C(r-1).
            q.push(n);
            c.push(a[m-1]);
                                                                  >> input is: n and r.
            m--, n--;
                                                                  * /
        else if(direction[m][n]=='U') m--;
                                                                  #include <iostream>
        else n--;
                                                                  #include <cstdio>
                                                                  #include <cstring>
    cout << "LCS : ";
                                                                  #include <cmath>
    while(!c.empty())
                                                                  #include <cstdlib>
                                                                  #include <queue>
        cout << c.top();
                                                                  #include <stack>
```

```
#include <vector>
#include <algorithm>
#include <cctype>
#include <fstream>
#include <map>
#include <list>
#include<set>
\#define chng(a,b) a^=b^=a^=b;
#define sz 100
#define pb(a) push back(a)
#define pp pop back()
#define ll long long
#define fread freopen("input.txt","r",stdin)
#define fwrite freopen("output.txt", "w", stdout)
#define inf (1<<30-1)
#define clr(abc,z) memset(abc,z,sizeof(abc))
#define PI acos(-1)
using namespace std;
int dp[sz][sz];
int nCr(int n, int r)
    if(r==1) return n;
    if(n==r) return 1;
    int &ret = dp[n][r];
    if(ret!=-1) return ret;
    ret = nCr(n-1,r) + nCr(n-1,r-1);
    return ret;
int main()
    int data[sz], n,r;
    clr(dp,-1);
    while (cin>>n>>r) cout<<"nCr = "<<nCr(n,r)<<endl;
    return 0;
```