1. Programer Vasa se zaigrao i uziva u sledecoj igri. Na početku igre na sva polja kvadratne N×N table postave se čokolade u omotu raznih boja. Kad je na potezu, igrač mora zameniti neka dva susedna (gore, dole, levo ili desno) omota različite boje te nakon toga odabrati neki niz od jednog ili više uzastopnih (u vrsti ili koloni) čokolada u omotu iste boje koje će uzeti i pojesti. Za dato početno stanje table, pomozite Vasi i napišite program koji će ispisati koliko najviše cokolda može osvojiti u prvom potezu.

ULAZNI PODACI

U prvoj vrsti nalazi se prirodni broj N (3 ≤ N ≤ 50), dimenzije table. U sledećih N crsta nalazi se stanje table, j-ti znak u i-tom redu označava boju čokolada na polju (i, j) : C (crvena), P (plava), Z (zelena), Y (žuta). Prvi potez je uvek moguće napraviti.

IZLAZNI PODACI

U prvom i jedinom redu ispišite najveći broj cokolada koje mali Vasa može osvojiti u prvom potezu.



Objašnjenje drugog test primera: u prvom redu uočavamo niz PPPP, pa zamenom bilo

koja dva omota u ostalim redovima niz ostaje očuvan i Vasa ga može uzeti.

Objašnjenje trećeg test primera: zamenom čokolada Y i C u 4.redu dobija se niz CCCC u prvoj koloni.

1. Majka je malom Vasi upravo kupila tablu čokolade od N redova sa M kockica u svakom redu. Vasa nije sebičan, te je odlučio podeliti svoju čokoladu sa prijateljima. On će podeliti čokoladu tako da radi proizvoljne rezove između kockica čokolade, i to tako da kad završi s rezanjem svaki deo ima oblik kvadrata. Uz to, on želi podeliti čokoladu samo s najboljim prijateljima zbog čega ga zanima koji je najmanji broj kvadrata na koji može podeliti celu čokoladu.



Na gornjim slikama prikazane su dve moguće podele čokolade - N je 3, a M je 4. Prva

podela čini 6 kvadrata, a druga 4 kvadrata i to je ujedno i podela s najmanje kvadrata.

ULAZNI PODACI

U prvom i jedinom redu nalaze se dva prirodna broja N i M (1 ≤ N, M ≤ 1 000), broj redova i

kolona čokolade.

IZLAZNI PODACI

U prvom i jedinom redu ispišite najmanji broj kvadrata na koji se čokolada može podeliti.



Resenja

#include <cstdio>

#include <algorithm>

 using namespace std;

const int MAXN = 105;

char a[MAXN][MAXN];

int n;

int maxniz() {

 int best = 0;

 for( int i = 0; i < n; ++i )

 for( int j = 0; j < n; ++j ) {

 int c1 = 0, c2 = 0, x = i, y = j;

 while( x < n && a[x][j] == a[i][j] ) x++, c1++;

 while( y < n && a[i][y] == a[i][j] ) y++, c2++;

 best = max( best, max(c1,c2) );

 }

 return best;

}

int main( void ) {

 scanf( "%d", &n );

 for( int i = 0; i < n; ++i )

 scanf( "%s", a[i] );

 int rj = 0;

 for( int i = 0; i < n; ++i )

 for( int j = 0; j < n; ++j ) {

 if( i+1 < n && a[i][j] != a[i+1][j] ) {

 swap( a[i][j], a[i+1][j] );

 rj = max( rj, maxniz() );

 swap( a[i][j], a[i+1][j] );

 }

 if( j+1 < n && a[i][j] != a[i][j+1] ) {

 swap( a[i][j], a[i][j+1] );

 rj = max( rj, maxniz() );

 swap( a[i][j], a[i][j+1] );

 }

 }

 printf( "%d\n", rj);

 return 0;

}

2.

#include <cstdio>

#include <algorithm>

#include <cstring>

using namespace std;

int dp[1001][1001];

int greedy(int n, int m) {

 int c = 1;

 while (n != m) {

 if (n > m) n -= m;

 else m -= n;

 ++c;

 }

 return c;

}

int main() {

 int n, m;

 scanf("%d%d", &n,&m);

 if (m < n) swap(n, m);

 for (int j = 1; j <= m; ++j) {

 dp[1][j] = dp[j][1] = j;

 for (int i = 2; i <= n; ++i) {

 dp[i][j] = 1000000000;

 }

 }

 for (int i = 2; i <= n; ++i) {

 dp[i][i] = 1;

 for(int k = 1; k\*2 <= i; ++k) {

 for (int j = i+1; j <= m; ++j)

 dp[i][j] = min(dp[i][j], dp[i-k][j] + dp[k][j]);

 }

 for (int j = i+1; j <= m; ++j) {

 for(int k = 1; k\*2 <= j; ++k)

 dp[i][j] = min(dp[i][j], dp[i][j-k] + dp[i][k]);

 }

 for (int j = i+1; j <= m; ++j) {

 dp[j][i] = dp[i][j];

 }

 }

 printf("%d\n", dp[n][m]);

 return 0;

}