Teoria de Juegos

# Chess

Maximum rooks in an m\*n chessboard so they are not in position to take any other rook is minimum (m,n).

Maximum queens in an m\*n chessboard (m>=4 and n>=4) so they are not in position to take any other queen is minimum (m,n).

Maximum Kings in an m\*n chessboard so they are not in position to take any other Kings is (m+1) div 2 \* (n+1) div 2

1. Caballos

**private** **static** **int** metodo(**int** m, **int** n) {

**if**(m == 1){

**return** n;

}

**if**(n == 1){

**return** m;

}

**if**(m == 2){

**int** resp = 0;

**for** (**int** i = 0, k = 0; i < n; i++,k++) {

**if**(k == 4){

k = 0;

}

**if**(k == 0 || k == 1){

resp+=2;

}

}

**return** resp;

}

**if**(n == 2){

**int** resp = 0;

**for** (**int** i = 0, k = 0; i < m; i++,k++) {

**if**(k == 4){

k = 0;

}

**if**(k == 0 || k == 1){

resp+=2;

}

}

**return** resp;

}

**if**(m\*n%2==0){

**return** m\*n/2;

}

**else**{

**return** m\*n/2+1;

}

}

private static int alfiles(int n) { //n\*n

if(n == 1){

return 1;

}

else{

return (n\*2)-2;

}

}

**SALTO DE CABALLO**

//cantidad minima necesaria para ir de una posicion a otra

//solo se puede saltar al igual que un caballo de ajedrez

ll SaltoCaballo(ll x1,ll y1,ll x2,ll y2){

ll dx =abs(x2-x1);

ll dy =abs(y2-y1);

ll lb= max(dx+1 , dy + 1)/2;

lb = max(lb, (dx + dy + 2)/3);

while((lb % 2) != (dx+ dy)%2) lb++;

if(abs(dx)==1 && !dy) return 3;

if(abs(dy)==1 && !dx) return 3;

if(abs(dx)==2 && abs(dy)==2) return 4;

return lb;

}

**TORRES DE HANOI**

int n;

void move( int n, char from, char to, char aux ) {

if ( n == 1 )

printf( "Move disk from %c to %c\n", from, to );

else {

move( n - 1, from, aux, to );

printf( "Move disk from %c to %c\n", from, to );

move( n - 1, aux, to, from ); }}

int main() {

scanf( "%d", &n );

move( n, 'A', 'C', 'B' ); }