Codeforces Round 511 Div.A B

Description

Given a $n \times m$ chessboard, every time put two chessman with Manhattan distance 3 between them.

Calculate the maximum number of chessmen you can put on it.

$n, m \leq 10^9$.

Solution

All possible pairs of position is a biparite graph, but n, m is too large, we cannot run hungry algorithm.

Try some simple case first.

In a line ($n = 1$), the answer is obviously $\lfloor \frac{n}{6} \rfloor \times 6 + 2 \times max(n\%6 - 3, 0)$, that every 6 blocks can be filled and the rest can be filled if and only if they are more than 4.

Thus we've dealt the case of $n == 1$.

Now take $n == 2$ into consideration. We know an 2×2 is 0, an 2×3 is 2, and an 2×7 is 12, others can be full.

Why?

For $m \leq 7$, only $4, 5, 6$ can be full, but we can use them to generate all numbers larger than 7. For an even number, it can be split into more than four 2s. And four is two 2s, six is three 2s, so we just need to prove 2 and 3 can generate all numbers larger than or equals to 4.

The same, for an even number, it can be split into 2s, that's what the TWO can deal, as for odds, we can take a 3 from it, to convert it into even.

So 4 and 6 can generate even numbers larger than 7. And minus 5 will deal the odds.

Also we can find that a 4×3 is full.

Use what we've proved above, we can generate all $4 \times x$ rectangles.

One more step, we have $4 \times x$ and $6 \times x$ (made of 6×1) now, that means we can deal all even numbers!

The last problem is n, m are both odd.

We've discussed above that use 3 or 5 can deal odd cases, and we can find that 3×3, 3×5 and 5×5 are all missing only one blank, and the rest of the rectangle will be full. The ans is $\lfloor \frac{n \times m}{2} \rfloor \times 2$.

```
#include <iostream>
#include <cstring>
#include <cstdio>
#include <algorithm>

using namespace std;

typedef long long ll;

ll n, m;

ll solve2(ll x) {
  if (x == 2) return 0;
  else if (x == 3) return 4;
  else if (x == 7) return 12;
  else return x << 1;
}

int main() {
  scanf("%lld%lld", &n, &m);
  if (n > m) swap(n, m);
  if (n == 1) printf("%lld\n", m / 6ll * 6ll + 2ll * max(m%6 - 3, 0ll));
  else if (n == 2) printf("%lld\n", solve2(m));
  else printf("%lld\n", n * m / 2 * 2);
  return 0;
}
```