Here is an explanation of my solution for the problem SHORT.

Denote $x = a - n$ and $y = b - n$. Then the problem is equivalent to finding the number of pairs (x, y) such that $n(x + y + n - 1) \vdots xy$ and $1 \le x, y \le M$ where $M = k - n - 1$.

At first one should note that if $n = 0$ then the answer is M^2 and you need to handle long arithmetic to calculate this. I almost sure that this trap is the main reason why we have so small rate of correct solutions for this problem.

Now let $n \ge 1$. The solution divides into two steps.

1. Since $n(x + y + n - 1) \vdots xy$ then

$$n(x + y + n - 1) = kxy \quad \text{for some} \quad k \ge 1 \tag{1}$$

(this k has nothing common with k from the input).

Note that if n and k have common divisor d then we can divide (1) by d. So let $g = \gcd(n, k)$, $n_1 = n / g$, $k_1 = k / g$. Then (1) is equivalent to $n_1(x + y + n - 1) = k_1 xy$ or

$$(k_1 x - n_1)(k_1 y - n_1) = n_1(k_1 n + n_1 - k_1). \tag{2}$$

Denote $m = k_1 n + n_1 - k_1$ and $w = n_1 m$. Clearly $k_1 x - n_1$ and $k_1 y - n_1$ are divisors of w. So in order to find all x, y that satisfy (2) we can iterate through all divisors of w and for each divisor d we obtain equations for x and y. Namely $k_1 x - n_1 = d$, $k_1 y - n_1 = w / d$.

Finding all divisors of w also requires a bit of thinking. Since $w = n_1 m$ we can find prime factorization of each of the numbers n_1 and m and then merge them to obtain prime factorization of w. After we find the prime factorization of w we can generate all its divisors by simple backtracking procedure. Prime factorization of some number q can be done by usual way. Namely, before handling input we should run Eratosthenes sieve up to some N, calculate for each $n \le N$ the minimal prime divisor minp[n] of n and also find the list of all primes less than N. For this problem the best choice of N is 100000, that is the maximal possible value of n in the input. Now if $q \le N$ then we can factor it by $O(\log q)$ operations using array minp[]. Otherwise we iterate through all primes not greater then $\sqrt{q}$.

Thus factorization of n_1 always requires $O(\log n_1)$ operations and factorization of m requires $O\left(\log m + \left\{m \le N \,?\, 0 : \pi\left(\sqrt{m}\right)\right\}\right)$ operations. Here $\pi(x)$ is the standard notation for the number of primes that is not greater than x. So we see that for each k we can find all pairs (x, y) that satisfy (2) using $O\left(\log w + \left\{m \le N \,?\, 0 : \pi\left(\sqrt{m}\right)\right\} + \tau(w)\right)$ operations. Here $\tau(w)$ stands for the number of divisors of w.

Now the full description of the first step of the solution is the following. Iterate through values of k starting from 1. For each k we calculate w, its prime factorization, generate list of its divisors and find required pairs (x, y). Then we increment the total number of performed operations by $\log w + \left\{m \le N \,?\, 0 : \pi\left(\sqrt{m}\right)\right\} + \tau(w)$. And if it exceeds some predefined number then we increment k by 1, save it for second step and stop the first step of the solution. For the bound of the number of iterations I recommend to use number of the form $c \cdot n$ where c is a small constant ($c = 1,2,3,4$).

2. Due the first step it remains to find only those pairs (x, y) for which

$$n(x + y + n - 1) \vdots xy \text{ and } n(x + y + n - 1) \ge kxy. \tag{3}$$

Since now we are only interested in pairs for which $x \le y$. If we find any such pair then we add 1 to the answer and if $x < y$ add additional 1. In C++ it looks very simple

```
ans += 1 + (x < y);
```

Now we take a closer look at (3). As in the first step we can rewrite the inequality in the form $(kx - n)(ky - n) \le n((k + 1)n - k)$. If $ky \ge kx > n$ then it follows that $n((k + 1)n - k) \ge (kx - n)(ky - n) \ge (kx - n)^2$. Hence

$$x \le \frac{\sqrt{n((k+1)n - k)} + n}{k} < n\frac{\left(1 + \sqrt{k+1}\right)}{k}.$$

So we have a quite good upper bound for x. Let's iterate through all values of x from 1 to this bound. For the fixed value of x we start with proper description of those y for which

$$n(x + y + n - 1) \vdots xy. \tag{4}$$

Let $g = \gcd(n, x)$, $n_1 = n / g$, $x_1 = x / g$ and hence $\gcd(n_1, x_1) = 1$. Then (4) is equivalent to

$$n_1(y+x+n-1)\vdots x_1 y. \tag{5}$$

Let $y=v\cdot z$ where $\gcd(z,x_1)=1$ and z is the maximal divisor of y with this property. It means that v has in its prime factorization only prime divisors of x_1 but z is not divisible by any such prime number. Then (5) is equivalent to the system of divisibilities

$$n_1(v\cdot z+x+n-1)\vdots z \tag{6}$$

$$n_1(v\cdot z+x+n-1)\vdots x_1 v \tag{7}$$

Obviously (6) is equivalent to

$$n_1(x+n-1)\vdots z \tag{8}$$

Since $\gcd(n_1,x_1)=1$ and v has in its prime factorization only prime divisors of x_1 we have $\gcd(n_1,v)=1$ and hence (7) is equivalent to

$$v\cdot z+x+n-1\vdots x_1 v \tag{9}$$

Clearly (9) is equivalent to

$$x+n-1\vdots v$$

$$z+\frac{x+n-1}{v}\vdots x_1 \tag{10}$$

Since $\gcd(z,x_1)=1$ then from (10) follows that $\gcd\left(\frac{x+n-1}{v},x_1\right)=1$. Since v has in its prime factorization only prime divisors of x_1 then $m=\frac{x+n-1}{v}$ is the maximal divisor of $x+n-1$ for which $\gcd(m,x_1)=1$. So we see that v does not depend on y. Namely m can be found by the following procedure. Start with $m=x+n-1$ and then divide m by $\gcd(m,x_1)$ until this gcd is greater then 1. Then v is equal to $\frac{n+x-1}{m}$. Finally since $\gcd(z,v)=1$ and $x+n-1\vdots v$ then (8) is equivalent to $n_1 m\vdots z$.

Summarizing all facts we see that $n(x+y+n-1)\vdots xy$ if and only if $y=v\cdot z$ where

$$n_1 m\vdots z \quad \text{and} \quad z+m\vdots x_1. \tag{11}$$

Here n_1,x_1,v,m can be found explicitly and quickly by n and x.

Now there are two possibilities how to find all z that satisfiy (11).

2.1. Find all divisors of $n_1 m$ like in the first step of the solution, iterate through all of them and check for each such divisor z the conditions $z+m\vdots x_1$, $x\le y$, $n(x+y+n-1)\ge kxy$ and $y\le M$ where $y=v\cdot z$. This requires $O(T_1)$ operations where T_1 is the number of divisors of $n_1 m$.

2.2. Another way is to use inequalities $x\le y$ and $(kx-n)(ky-n)\le n((k+1)n-k)$. It is applicable only when $kx>n$. In this case we obtain the double inequality for y

$$x\le y\le \frac{1}{k}\cdot\left(\frac{n((k+1)n-k)}{kx-n}+n\right).$$

From (11) we have $y=v(x_1 t+r)$ where $t\ge 0$ and $r=(-m)\bmod x_1\ge 0$. Let T_2 be the number of those $t\ge 0$ for which

$$x\le v(x_1 t+r)\le \frac{1}{k}\cdot\left(\frac{n((k+1)n-k)}{kx-n}+n\right) \tag{12}$$

Finding T_2 is an easy exercise on floor function. If T_2 is small enough we can simply iterate through all values of t that satisfy (12) and check the required conditions on x and y. This requires $O(T_2)$ operations.

We can find T_1 in $O(\log(n_1 m))$ time and T_2 in $O(1)$ time. Then we compare T_1 and T_2 and choose the fastest approach among this two.

One should note that for $x>n\left(1+\sqrt[4]{k}\right)/k$ we have $T_2=O\left(\sqrt{k}\right)$ and it is often considerably smaller than T_1 since k will be about 500-1000 after the first step of the solution for the maximal values of n. So for $kx>n$ we choose in general the second approach but for $kx\le n$ only the first approach is applicable.

It is quite complicated to give an adequate estimate for the working time of this solution. But indeed it is fast. I have found answers for 100 largest values of n with $M=10^{18}$ in about 2.2 seconds on my laptop (Celeron Dual-Core T3000 1.80GHz, 3Gb RAM)