code jam
print "hello, world!"

Practice Mode

Round 1B 2010

Contest Analysis

Questions asked 1

Submissions

File Fix-it

12pt	Not attempted **3049/3404 users** correct (90%)
14pt	Not attempted **2909/3047 users** correct (95%)

Picking Up Chicks

13pt	Not attempted **1430/1965 users** correct (73%)
17pt	Not attempted **1393/1424 users** correct (98%)

Your Rank is Pure

14pt	Not attempted **1036/1705 users** correct (61%)
30pt	Not attempted **502/827 users** correct (61%)

Top Scores

Gluk	100
yuhch123	100
Gennady.Korotkevich	100
SergeyRogulenko	100
andrewzta	100
vepifanov	100
burunduk3	100
nika	100
mystic	100

Contest Analysis

Overview | Problem A | Problem B | Problem C

The solution for this problem consists of several steps, each making the problem a bit easier until it becomes simple.

Step 1. Do swaps as fast as possible (or don't do them at all)

Suppose a chick catches another one in front of it. We have the following options:

1. lift the slow one immediately to let the fast one pass;
2. let them run together for some time, and then lift the small one;
3. let them run together all remaining time and not let the fast one pass at all.

The first observation that we need to solve this problem is that option 2 is never necessary. Intuitively: what's the point of holding the fast one if we will still do the same work on swapping them later? Formally: suppose we have a sequence of swaps that forms the solution for the problem, and that swaps chicks **P** and **Q** at time $\mathbf{T_1}$, while they first meet at time $\mathbf{T_0}$, $\mathbf{T_0}<\mathbf{T_1}$. Let's move this swap to time $\mathbf{T_0}$. Note that for each particular chick and each particular moment, the position of this chick at this moment will either stay unchanged, or move closer to the barn. Because of this, the changed solution is also valid.

Step 2. Never swap two chicks that will make it to the barn in time

Suppose that we swap two chicks that would both make it to the barn in time in the end. Then we can avoid this swap and still have both chicks arrive at the barn in time. We will need to do the same number of swaps for the fast chick during the rest of the way than we had to previously (or even fewer, since some chicks might get out of our way), so we'll save at least one swap by keeping the two chicks together until the end.

Step 3. If a chick can't make it to the barn in time, all chicks behind it will have to swap with it

Suppose a chick can't theoretically make it to the barn in time: $\mathbf{X_i}+\mathbf{T}*\mathbf{V_i}<\mathbf{B}$. Then all chicks that start behind it will need to be swapped with this check or else they won't make it to the barn in time either.

Step 4. Split the chicks into two classes to get a lower limit on the number of swaps

Vasyl	100

Let's paint the chicks that can theoretically make it to the barn in time with *red* color, and the ones that can't do that with *blue* color.

For every red chick, the amount of swaps needed to get to the barn on time is at least the number of blue chicks that start closer to the barn. This gives us a lower bound on our answer: if we count this number for each red chick, then the answer is at least the sum of **K** smallest such numbers.

Step 5. The lower bound that we've found is actually the correct answer

At the previous step, we've found some swaps that are necessary in order to solve this problem. Now we note that we don't need any more swaps. More precisely, let's take **K** red chicks that are closest to the barn initially, and swap them with the blue chicks as soon as they reach any. Since we took **K** red chicks that are closest to the barn, the number of blue chicks that will get in their way is minimum possible (remember that we never need to swap two red chicks!).

Step 6. Code!

After making all of the above observations, the actual solution becomes very simple:

```
num_red = 0
num_blue = 0
answer = 0
for i = N - 1 .. 0:
  if num_red == K:
    break
  if X[i] + T * V[i] < B:
    num_blue += 1
  else:
    num_red += 1
    answer += num_blue
if num_red >= K:
  output answer
else
  output "IMPOSSIBLE"
```