Watermelons

- Author — Batyr Sardarbekov;
- Developers — Aibar Kuanyshbay, Dinmukhamed Tursynbay, Batyrkhan Altynbekov.

Let's, for convenience, immediately exclude the case for all subtasks when $n + m < K$, since the answer here is -1, because each friend should receive at least one watermelon.

Subtask 1 ($m = 0$, $a_i = a_{i+1}$ $(1 \le i < n)$) — 9 points

Since all a_i are equal, let's define the number $x = a[1]$. In this subtask, it is sufficient to evenly distribute all n watermelons among K friends, then the distribution will be fair, because the difference in watermelons between any two friends is no more than x, and the sum of the watermelons of the remaining friends is at least $2 \cdot x$, due to $n \ge 3$. This can be easily done, for example, as follows: first, distribute $\lfloor \frac{n}{K} \rfloor$ watermelons to all friends, and then distribute one watermelon to friends with numbers from 1 to $n \ mod \ K$.

Subtask 2 ($a_i = a_{i+1}$ $(1 \le i < n)$, $b_j = b_{j+1}$ $(1 \le j < m)$) — 19 points

This subtask can be divided into three cases:

1. If $K \le n$, then the solution is exactly the same as in subtask 1.
2. If $K = n + 1$, then distribute one watermelon a_1 to the first n friends, and one watermelon b_1 to the $(n + 1)$-th friend. If it turns out that $b_1 > n \cdot a_1$, then give one more watermelon b_1 to the 1-st friend (if in this case $m = 1$, then the answer is -1). In both cases, the distribution will be fair.
3. If $K \ge n + 2$, then distribute one watermelon a_1 to the first n friends, and one watermelon b_1 to the remaining $K - n$ friends. Then the distribution will be fair, because there are at least 2 friends with a_1 and at least 2 friends with b_1.

Subtask 3 (It is not necessary to output the distribution of watermelons) — 24 points

The solution to this subtask makes a significant contribution to solving the full problem. Let's say we bought v watermelons from the seller, and received a new array of watermelons c of $n + v$ watermelons. Now we want to know if it is possible to make a fair distribution with the current set of watermelons.

Claim: If $2 \cdot maxC \le \sum_{i=1}^{n+v} c_i$, then the answer exists ($maxC$ – the maximum number in the array c). Let's call this condition the *general* condition.

Proof: If $2 \cdot maxC > \sum_{i=1}^{n+v} c_i$, then there is no answer, because $maxC > \sum_{i=1}^{n+v} c_i - maxC$ contradicts a fair distribution. Otherwise, we will show that it is enough to have 3 friends for a fair distribution, and if there are more friends, it cannot spoil anything. Let's say we have 3 friends. Let's define S_i – the sum of the weights of watermelons of the i-th friend. Let's first sort the watermelons in non-decreasing order. Give the watermelon c_{n+v} (i.e., one watermelon with the maximum weight) to the first friend. Now start distributing the watermelons with numbers from 1 to $n + v - 1$ one by one to the remaining two friends according to the following algorithm: give the i-th watermelon to the 2-nd friend if at the moment $S_2 \le S_3$, otherwise, give it to the 3-rd friend. After such a distribution, it will be true that $|S_2 - S_3| \le maxC$, because when we give the i-th watermelon to the 2-nd friend, then before the transfer, $S_2 \le S_3$, and after the transfer, it cannot be that $S_2 + c_i > S_3 + maxC$, because $c_i \le maxC$ (and vice versa, if we give it to the 3-rd friend). In the end, it will turn out that $S_1 = maxC$, and $|S_2 - S_3| \le maxC$. Now it remains only to prove that $S_1 \le S_2 + S_3$, $S_2 \le S_1 + S_3$, and $S_3 \le S_1 + S_2$. According to the initial condition, we know that $maxC \le \sum_{i=1}^{n+v} c_i - maxC$, therefore $S_1 \le S_2 + S_3$. Since the cases for S_2 and S_3 are analogous, let's assume that $S_3 \ge S_2$, and prove the correctness for S_3. Since $|S_3 - S_2| \le maxC$, then $S_3 \le S_2 + maxC$,

which proves a fair distribution. The remaining $K-3$ friends can be given one watermelon from S_2 or S_3 each, so that S_2 and S_3 do not remain empty. Obviously, if S_2 and S_3 satisfy the condition of a fair distribution, then one watermelon from S_2 or one watermelon from S_3 cannot fail to satisfy the condition.

Now we need to find the minimum number of elements from the array b that, when obtained, will give us an array c that satisfies the general condition. The answer is 0 if the array a already satisfies the general condition, and $n \geq K$. Otherwise, let's sort the array b in non-decreasing order, and iterate through the maximum b_i we want to take. It is obvious that the most optimal next element of the array b that we will take is b_{i-1}, because if the element does not change the maximum, then for the general condition it is better for the sum to be as large as possible. Thus, if we have fixed the maximum b_i, we will go from right to left until the general condition is met, or we have fewer watermelons than K. Thus, we will find the maximum j for which, if we buy watermelons $b_j, b_{j+1}, \ldots, b_i$, we will be able to make a fair distribution. It is not difficult to understand that j can be found by *binary search* or *two-pointer method*, because if j fits, then $j-1$ will also fit. In the end, from all suitable $i-j+1$, we will output the minimum (if none fits, then -1).

Subtask 4 ($m = 0$) — 20 points

From subtask 3, we know the general condition, which determines whether there is an answer for a specific set of watermelons or not. If the answer exists, then it can be restored exactly as written in the Proof of the general condition in subtask 3. That is, first fill in S_1, S_2, S_3 for the first three friends, and then distribute one watermelon from S_2 and S_3 to the remaining friends.

Subtask 5 ($b_i = 1$ $(1 \leq i \leq m)$) — 10 points

Since all elements of the array b are equal, we only need to know the number of watermelons we will buy. It is sufficient to iterate through how many watermelons we will buy, check the general condition from subtask 3, and restore the answer as in subtask 4.

Subtask 6 (No additional constraints) — 18 points

From subtask 3, we know how many watermelons need to be bought and the segment of watermelons that need to be bought in the sorted array b. From subtask 4, we know how to restore the answer if we have a specific set of watermelons. By combining these solutions, we can get a complete solution.