Editorial - Partition

Problem summary

Ulug'bek is given an array A of N positive integers and may add at most $K-1$ integers from $[1, M]$. Temur receives only the sorted extended array B and must partition it into K non-empty groups with equal sums. Temur cannot tell which values were added.

Let $L = |B|$. We always have $L \leq N + K - 1$. We also sort the values of A in non-decreasing order $A[0] \leq A[1] \leq \ldots \leq A[N-1]$.

Subtask 1: $N = 3$ (5 points)

We can do a case analysis to solve the case for $N = 3$. Sort the values and write them as $x = A[0], y = A[1], z = A[2]$.

- If $K = 2$, let $d = |x + y - z|$. If $d > 0$, add it to whichever of $\{x, y\}$ and $\{z\}$ has the smaller sum; if $d = 0$, add nothing. If $z \geq x + y$, then $d < z \leq M$; otherwise, $d = x + y - z \leq x \leq M$.
- If $K = 3$, add each positive value among $z - x$ and $z - y$. Then the three groups can all have sum z.

Temur may recover the groups with a similar case analysis. Alternatively, B contains at most five values, so he can enumerate at most $3^5 = 243$ assignments and keep one in which all groups are non-empty and have equal sums.

Subtask 2: $M = 1$ (4 points)

Every value is 1. Add $c = (K - (N \bmod K)) \bmod K$ ones to the array. We have $0 \leq c \leq K - 1$, and the new number of elements is divisible by K. Since $N \geq K$, Temur can assign the same positive number of elements to every group. The complexity is $O(N + K)$.

Subtask 3: $M \leq 2$ (4+7 points)

The case $M = 1$ was handled above. Suppose $M = 2$, and let $S = \sum_{a \in A} a$. Define r as the smallest non-negative integer satisfying $S + r \equiv 0 \pmod{2K}$. If $r \neq 2K - 1$, add $\lfloor r/2 \rfloor$ copies of 2 and, if r is odd, one copy of 1. This uses at most $K - 1$ values and makes the target group sum even.

The only exceptional case is $r = 2K - 1$, where the construction above would use K values. Here $S \equiv 1 \pmod{2K}$; instead, add $K - 1$ ones. The original odd sum contains at least one 1, so there are now at least K ones in total, and the new target is odd.

Temur reconstructs the groups as follows:

- If the target is odd, first put one 1 into every group.
- Pair all remaining ones. A pair of ones and a single two both have value 2, so these value-2 units can

be distributed equally among the groups.

After the optional first step, the number of remaining ones is even and the remaining total is divisible by $2K$. Hence the number of value-2 units is a multiple of K. This takes $O(N + K)$ time.

Why the problem always has a solution

Before designing an algorithm, it is worth checking that the task is always possible. Partition A greedily into groups $G_0, \ldots, G_{K-1}$ with sums $S_0, \ldots, S_{K-1}$:

- process the values in any order;
- always append the current value to a group with the **smallest** current sum.

Claim. At every moment $\max_j S_j - \min_j S_j \le M$.

Suppose it holds, and we add $A[i]$ to a lightest group G_k. For every j, $S_k + A[i] \le S_j + M$, because $S_k \le S_j$ and $A[i] \le M$. The minimum can only grow, so the new range is still at most M. $\square$

Re-index the groups so that $S_0 \le S_1 \le \cdots \le S_{K-1}$ and add the deficits $S_{K-1} - S_0, \quad S_{K-1} - S_1, \quad \ldots, \quad S_{K-1} - S_{K-2}$, omitting the zeroes. These are at most $K - 1$ values, each in $[1, M]$ by the claim, and they raise every group to the common sum S_{K-1}.

This proves that an answer always exists, but it is not an algorithm: after sorting, Temur cannot tell the deficits apart from the original values, and re-running the greedy on the extended multiset generally produces different groups. Making the construction *recoverable* is the actual problem.

Subtask 4: $N \le 10$ (5+10 points)

Ulug'bek uses the greedy construction above, and Temur brute-forces the encoding: he enumerates every subset of positions of B as the proposed set of added values, runs the same deterministic construction on the remaining multiset, and checks whether the multiset of deficits it produces is exactly the proposed subset. The true subset always passes this test, and any subset that passes yields a valid partition of B. Since $|B| \le N + K - 1 \le 2N - 1 \le 19$, the complexity $O\big(2^{|B|}\,|B| \log |B|\big)$ is comfortable.

Subtask 5: $K = 2$ (7 points)

There are multiple approaches to solve the subtask, and we are going to introduce one that is most helpful to get the full solution to the task.

Let's try another way to construct the solution. Let's try to cut the *sorted* array $A[0 \ldots N-1]$ into a prefix and a suffix. For $1 \le t < N$ let $X_t = \sum_{i<t} A[i]$, and $Y_t = \sum_{i \ge t} A[i]$, and choose t minimizing $\max(X_t, Y_t)$.

Write $X = X_t$, $Y = Y_t$. If $X = Y$, add nothing; otherwise add $c = |X - Y|$ to the smaller side. We can note that $c \le M$ always holds.

That's because, assuming $X < Y$, if $t = N - 1$, the suffix is the single value $A[N-1]$ and $c = Y - X < Y = A[N-1] \le M$. Otherwise consider moving $A[t]$ into the prefix. This is another valid cut, so it cannot lower the larger sum below Y. The new suffix sum $Y - A[t]$ is smaller than Y, so necessarily $X + A[t] \ge Y$, that is, $c = Y - X \le A[t] \le M$. The case $X > Y$ is symmetric.

Temur's reconstruction. Let $V = \left(\sum_{b \in B} b\right)/2$. Take the largest remaining value as long as it fits into a group of sum V; when it no longer fits, take one value exactly equal to the missing amount. The rest forms the second group.

If the suffix was the larger side, its values are the largest ones and Temur simply collects them until he reaches V. If the prefix was larger, the suffix was completed with $c \le A[t-1] \le A[t]$; Temur first takes all suffix values, and the missing amount is exactly c. The next prefix value is at least c: if it is larger it does not fit, so the exact-complement step picks up c; if it equals c the two copies are interchangeable.

The complexity is $O(N \log N)$.

Subtask 6: $K \le 3$ (7+12 points)

Let's continue the idea from Subtask 5: **cut the sorted array into consecutive blocks and complete each block to a common target**.

Suppose the original values form K consecutive blocks $G_0, \ldots, G_{K-1}$ with sums $S_0, \ldots, S_{K-1}$. Ulug'bek adds to block G_i the complement $C_i = V - S_i$, omitting it when it is zero. We need:

1. $0 \le C_i \le M$;
2. at least one $C_i = 0$, so at most $K - 1$ values are added;
3. Temur can recover the blocks after sorting.

For a fixed capacity V, greedily scan A from left to right and put as many consecutive values as possible into each block while keeping its sum at most V. This uses the minimum possible number of consecutive blocks, so feasibility is monotone in V. Binary search the smallest feasible V. Thus V is feasible using at most K blocks, while $V - 1$ is not.

Now define two greedy partitions.

- **Left greedy:** starting from the left, repeatedly take the longest prefix of the remaining array whose sum is at most V. Let the boundaries be $F_0 = 0, F_1, \ldots, F_K = N$.
- **Right greedy:** starting from the right, repeatedly take the longest suffix of the remaining array whose sum is at most $V - 1$. Let the boundaries be $R_0, \ldots, R_K = N$. Since $V - 1$ is infeasible, $R_0 > 0$.

We have $F_0 < R_0$ and $F_K = R_K = N$. For $K \le 3$, simply try every $t \in 1, \ldots, K$ and find one with $F_t = R_t$. Such a t must exist: let t be the first index with $F_t \ge R_t$. If $F_t > R_t$, then the left block $[F_{t-1}, F_t)$ contains the whole right block $[R_{t-1}, R_t)$, the previous value $A[R_{t-1} - 1]$, and at least one extra positive value. But maximality of the right block implies

$$A[R_{t-1} - 1] + \sum_{j=R_{t-1}}^{R_t - 1} A[j] \ge V,$$

so the left block would have sum $> V$, a contradiction. Hence $F_t = R_t$.

Use the first t left-greedy blocks and the remaining right-greedy blocks. Their common boundary is $F_t = R_t$, so they partition the whole array.

For a left-greedy block of sum $S < V$, maximality gives $S + A[\text{next}] > V$, hence $C = V - S < A[\text{next}] \le M$. For a right-greedy block, maximality under capacity $V - 1$ gives $S + A[\text{previous}] \ge V$, hence $C = V - S \le A[\text{previous}] \le M$.

Moreover, the last left-greedy block has sum exactly V: it contains the right-greedy block immediately before the meeting point together with its previous value, whose sum is already at least V. Therefore one complement is zero, so Ulug'bek adds at most $K-1$ numbers.

Temur's reconstruction. Let $V = (\sum_{b \in B} b)/K$. Repeatedly create a group as follows: take the largest unused value while it still fits into sum V; when the largest unused value no longer fits, take one unused value exactly equal to the missing amount.

The blocks are recovered from right to left. For every right-greedy block, its complement is no larger than the preceding original value; for every left-greedy block, its complement is smaller than the next original value. Thus larger values belonging to the current block are taken first, and when Temur gets stuck, the missing amount is exactly its complement. Equal copies are interchangeable.

With $K \le 3$, Temur can simply scan the whole array whenever he needs the largest unused value or the exact complement. The complexity is $O(N \log(NM) + N)$.

Subtask 7: $K \le 10$ (overall 62 points)

Again find the smallest feasible target V. At each step, while some interval of A remains, choose one of two operations:

- L: take the longest prefix with sum at most V;
- R: take the longest suffix with sum at most $V-1$.

There are only $2^K \le 1024$ sequences of K choices, so Ulug'bek can enumerate all of them.

For each sequence, simulate the K choices. Accept it if all elements are consumed and at least one block has sum exactly V. For each block of sum S, add the positive complement $V - S$. The same maximality argument as above guarantees that every complement is at most M.

Why must one sequence work? From the previous argument, if t is the first index where the left-greedy boundary reaches the corresponding right-greedy boundary, then in fact they are equal. Therefore the sequence consisting of t choices L followed by $K-t$ choices R is valid, and one of its left blocks has sum exactly V.

Using prefix sums, each L or R step can be simulated with binary search in $O(\log N)$ time. Thus Ulug'bek runs in $O(N \log(NM) + 2^K K \log N)$ time.

Temur uses the same reconstruction as in Subtask 6. With $K \le 10$, simply scanning all unused values for every group gives $O(NK)$ time.

Subtask 8: $K \le 100$ (overall 82 points)

A valid sequence always has the form $L, L, \ldots, L, R, R, \ldots, R$. Compute all left-greedy boundaries $F_0, \ldots, F_K$ with capacity V, and all right-greedy boundaries $R_0, \ldots, R_K$ with capacity $V-1$. Let t be the first index with $F_t \ge R_t$. As proved above, necessarily $F_t = R_t$. Therefore Ulug'bek directly uses the first t left-greedy blocks and the remaining $K-t$ right-greedy blocks, and adds their positive complements to V.

All boundaries can be computed by two linear scans, so after finding V, Ulug'bek takes $O(N+K)$ additional time.

Temur still uses the simple $O(NK)$ reconstruction from the previous subtasks. Since $K \leq 100$, this is fast enough.

Full Solution (100 points)

The construction from Subtask 8 already works for arbitrary K. The only remaining problem is that Temur's $O(NK)$ reconstruction is too slow.

Temur needs only two operations on the unused elements:

- find and remove the largest unused value;
- find and remove a value equal to a given missing amount.

There are multiple data structures can maintain the procedure, and even `std::set` is enough for an $O(L \log L)$ solution.

The overall complexity is $O(N \log(NM) + N \log N)$, with $O(N + K)$ memory.