Mix

Alice is a member of the Cyberland Chemistry Foundation. Open Day is approaching. To reproduce the sensational show a decade ago, the foundation has to prepare a special kind of reagent.

Although it is not easy to prepare, fortunately, Bob finds X bottles of reagent left by previous members, with a sheet of paper aside:

> At least 1, and at most K bottles of them contain impure reagent.

However, the labels are all detached, so Alice can only use a special machine to detect the impure reagent.

The machine can equip M test tubes. For each of them,

- Alice can choose **several bottles (possibly none)**, take a drop of sample from each of them, and mix all samples in the test tube.
- After the machine finishes working, if **any sample** comes from impure reagent, the machine would report `impure`, otherwise report `pure`. If it contains no sample, the result would always be `pure`.

Since the machine is slow, Alice is allowed to run the machine **only once**. During the only run, the machine would test for all tubes, so **before running the machine, Alice should determine which bottles of reagent are sampled and tested for each test tube.**

Now, Alice is required to detect all impure bottles of reagent from the machine. Of course, Alice knows how to solve it, but she wants to challenge you, a new member of the foundation.

Implementation Details

You need to implement the following two functions:

```
std::vector<std::vector<bool>> propose(int X, int K, int M)
```

- In this function, you need to provide which bottles of reagent are sampled and tested for each test tube.
- X: the number of bottles.
- K: the maximum possible number of impure bottles of reagent.
- M: the number of test tubes the machine can equip.
- Bottles and test tubes are labeled from zero.

- You need to return an $M \times X$ boolean matrix A, where $A_{i,j} = 1 (0 \leq i \leq M-1, 0 \leq j \leq X-1)$ represents that the sample from bottle j is mixed in the i-th test tube, $A_{i,j} = 0$ represents the opposite.
- For each test case, the grader would only run this function once.

```
std::vector<int> detect(int X, int K, int M, const std::vector<std::vector<bool>> &A, const std::vector<bool> &R)
```

- In this function, given the result, you are required to report all impure bottles.
- X: the number of bottles.
- K: the maximum possible number of impure bottles.
- M: the number of test tubes the machine can equip.
- A: the return matrix of `propose`.
- R: an array with length M, where $R_i (0 \leq i \leq M-1)$ is 1 if the result for the i-th tube is `impure`, and $R_i = 0$ represents the opposite.
- You need to return an array B containing no more than K elements, describing all bottles of reagent with impurities. You need to ensure that $0 \leq B_i \leq X-1$, and values in array B are distinct.
- **This function may run multiple times in one test case.**

Evaluation

For each test case, we will run the code exactly once. The evaluation process is as follows:

- Call `propose(X, K, M)` once and get matrix A.
- Repeat the following procedure T times:
  - Generate, by `mt19937` random number generator, integer $C \in [1, K]$ and C distinct integers from $[0, X-1]$ to form array `P`. Bottles in array `P` contain impurities, and others do not. The random number generator does not depend on matrix A.
  - Generate result R based on matrix A and array `P`.
  - Call `detect(X, K, M, A, R)` and get the result array `B`.
  - Check whether numbers in `B` are distinct, and form the same set as numbers in array `P`.
    - If it is not in one time, the result would be `Wrong Answer`, otherwise the result would be `Accepted`.

We will give the value of parameter T in each subtask. T can be large, so you should pay attention to the running time of your `detect` function.

It is guaranteed that given all constraints in the statement, the grader would use no more than 0.5 seconds and 128MiB memory.

Example

Attention: The following example does not belong to any subtask.

Consider the following call:

```
propose(2, 1, 1);
```

It means, there are two bottles of reagent with at most one of them containing impurities, and the machine has one tube. A possible return matrix is `[[1,0]]`, meaning that only the sample from the first bottle is dropped into the tube.

After that, a possible call may be:

`detect(2, 1, 1, [[1,0]], [0])`。

It means that given the return matrix of `propose`, the machine examines the only tube and reports that it is pure. Since at least one of the two bottles is impure, the only correct answer is `[1]`。

The sample `mix.cpp` can pass cases where $N = 2, K = 1, M = 1$.

Constraints

- $1 \leq X \leq 10^5$
- $1 \leq K \leq 100$
- $1 \leq M \leq 500$
- $1 \leq T \leq 10^5$

Subtasks

Subtask ID	Score	$X =$	$K =$	$M =$	$T =$
1	10	100	100	100	100
2	15	10^4	1	500	100
3	15	10^5	1	20	10^5
4	10	10^3	2	250	10^3
5	10	10^3	2	60	10^5
6	15	10^3	2	48	10^5
7	25	2.8×10^4	4	170	10^3

Sample grader

The sample grader has the same evaluation logic as mentioned above. The sample grader reads the input in the following format:

- Line 1: 5 integers $X, K, M, T, seed$, where $seed \in [0, 2^{32} - 1]$ is the seed for random number generator. When $seed$ is the same, the generated array `P` for each `detect` call is fixed.

If there are invalid cases during evaluation, the grader would print one of the following errors and exit:

- `Invalid Proposal`: The return value of the `propose` call is not an $M \times X$ matrix.
- `Detection C Repeating Numbers`: The C-th call of `detect` returns an array with repeating numbers.
- `Detection C Out of Bound`: The C-th call of `detect` returns an array that contains numbers not belonging to $[0, X - 1]$.
- `Failed Detection C`: The C-th call of `detect` returns an array that does not consistent with the answer. It would print your return array and the correct answer afterwards.

If there are no invalid cases during evaluation, the grader would print one line `Accepted.` and exit.