IP over Avian Carriers
Problem ID: avian

When you live out in the Finnish woods, far away from all the cities, Internet connectivity has a wildly varying quality. This has clear disadvantages when doing competitive programming, especially since most competitions take place online.

To improve reliability of your connection to the Internet, the Internet Engineering Task Force issued a standard on April 1, 1990 called *IP over Avian Carriers*. This standard describes how birds can be used to deliver Internet traffic. Birds do not suffer from e.g. cut-off cables or power outages, thus serving as an effective fallback to a standard Internet connection. Unfortunately, birds suffer from other problems. When sending a number of birds with data, they don't deliver data in order, and sometimes birds are lost on the way.

Your task is to implement two programs to add some redundancy to this bird delivery protocol – an encoder and a decoder. The encoder will be given a string X which consists of $C \cdot N$ bits (i.e. digits 0 or 1), with $N > 1$. It should produce a vector of K elements $S[0], S[1], \ldots, S[K-1]$ where each $S[i]$ is a string of N bits.

Afterwards, your decoder will be given a subset of C elements from this vector. It should then output the original bit string X.

Encoder

Your encoder should implement the function `vector<string> encode(int C, int K, int N, string X)`. It will be given the integers C, K and N from the task description, and the bit string X. X will have length $C \cdot N$, and only consists of characters `0` and `1`.

The encoder should output a vector of K strings. Each string should be of length N, and only contain characters `0` and `1`.

Decoder

The decoder should implement the function `string decode(int C, int K, int N, vector<string> Y, vector<int> I)`. It will be given the integers C, K and N from the task description, and a subset Y of the strings S output by `encode(C, K, N, X)`. Y and I will both have length C. I will contain the **zero-based** indices of the strings taken from the encoder, such that $Y[i]$ is the $I[i]$:th string (i.e. $Y[i] = S[I[i]]$).

The decoder should output the original string X of length exactly $C \cdot N$.

Implementations

You can implement your solution in C++, Java, Python or C#. The templates you should implement can be downloaded here. Note that you should only submit one of the files `avian.cpp`, `avian.java`, `avian.py` or `avian.cs`.

Notes: you should not use standard in or standard out (i.e. reading or writing to the terminal) in this task. Doing so will likely cause confusing errors.

Additionally, the program **will be restarted** before running the decoder. Thus, any global state you use will be erased.

Grading

Your solution will be graded on a set of subtasks. A subtask will consist of multiple test cases. To get the points for a subtask, your solution must pass all test cases that are part of the subtask.

Subtask	Score	Constraints
1	30	$C = 3, K = 4, 1 < N \leq 1000$
2	25	$C = 2, K = 4, 1 < N \leq 1000, N \equiv 0 \pmod 2$ i.e. N is even.
3	20	$C = 2, K = 4, 1 < N \leq 1000$
4	25	$C = 3, K = 5, 1 < N \leq 1000$

Input Format

The included sample judge reads input in the following format:

- line 1: C K N

- line 2: X
- line 3: $I[0]\ I[1]\ \ldots\ I[C-1]$

Output Format

The sample judge will output the return value of the `decode` function.