Enigma Numbers (AiSD 2020 P2)

Enigma

English

Your task is to implement a variant of an Enigma machine which encrypts messages in an alphabet consisting of n letters denoted by numbers $\{1,2,...,n\}$. Symbol 0 ends the message.

Rotors $W_x=\{w_{x,\ 0},\ w_{x,\ 1},...,w_{x,\ n}\}$and reflectors $R_y=\{r_{y,\ 0},\ r_{y,\ 1},...,r_{y,\ n}\}$ are given as permutations of numbers $\{1,2,...,n\}$ and simulate mechanical parts that can be used by an operator. The internal "wiring" of these parts does not change.

Typing a letter into the input encrypts it by passing it through the rotors into reflector and back through the rotors in the inverse sequence, as seen here. A stationary sequence of rotors and a reflector represent a simple substitution cipher, it is their movement which makes the decoding chalenging.

Each rotor can move counterclockwise by at most one step during each encoding. The first rotor moves before every encoding; therefore if the initial position of this single rotor is denoted by the last letter of the alphabet (i.e. n, see example I) the first letter is encoded according to substitutions as stated in the rotor's definition. Further rotors move according to rules outlined in scetion "Turnover notch positions" of this link and this section. Additional information about double stepping can be found here.

In case there are more than 3 rotors only the first 3 rotate.

Input can be thought as divided into two parts: definitions of pieces of the machine and p instructions how to encode a given message using some of those parts. Before encoding a message the machine is assembled from k out of m rotors and a single reflector (out of l) and the rotors are set to some initial positions. Refer to the input section for details.

Input:

Defintions of parts of the machine:

- n- number of letters (numbers) in the alphabet
- m - number of rotors, followed by their definitions $W_0,...,W_{m-1}$

    - definition of rotor W_i - permutation of the alphabet
    - number of letters which cause turnover, followed by these letters as in "Position of turnover notches" table
- *l* - number of reflectors
- definitions of reflectors $R_0,...,R_{l-1}$ - permutation of the alphabet

A set of *p* tasks to perform:

each task consists of 2 parts: machine setup and a message to encrypt.

- *p* number of tasks
- *k* - number of rotors in the machine
- *k* pairs $K_0,...,K_{k-1}$ - where $K_i=(j, t)$, where *j* is an index of a copy of rotor W_j and *t* is its initial position
    - rotors are given in order from fastest to slowest
    - initial position of a rotor - it is set as if it was rotated (no actual rotations occur) prior to encoding, e.g. 1 - rotor is set as stated in permutation, 2 - it is shifted by one letter, etc.
- *r* index of a reflector R_r
- a sequence of numbers separated by whitespaces to encrypt, ending with the 0 character

Output:

Encrypted sequences of numbers (without the 0 character).

Exmple of input:

```
4
4
1 2 4 3
1 2
3 2 1 4
0
4 3 1 2
0
3 2 1 4
0
3
2 1 4 3
4 3 2 1
```

```
1 2 4 3

6
1 0 4 2
1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 0
1 0 4 0
1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 0
1 0 4 1
1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 0
1 2 4 0
1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 0
1 3 4 0
1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 0
2 0 4 0 1 0
1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 0
```

Output:

```
1 1 1 3 2 4 2 2 4 3 4 1 3 2 3 4
2 3 2 3 1 4 1 4 4 1 4 1 3 2 3 2
3 4 3 4 4 3 4 3 1 2 1 2 2 1 2 1
2 3 2 3 1 4 1 4 4 1 4 1 3 2 3 2
4 4 4 4 3 3 3 3 2 2 2 2 1 1 1 1
2 2 4 2 3 4 1 4 4 4 2 4 1 2 3 2 2 2 4 2 3 4 1 4 4 4 2 4 1 2 3 2
```