Eiffel Assignment

Due Nov 14 by 11:59pm **Points** 40 **Submitting** a file upload **File Types** tgz
Available until Nov 14 at 11:59pm

This assignment was locked Nov 14 at 11:59pm.

Language

This assignment has to be completed using the Eiffel Programming Language. The documentation can be found at **https://www.eiffel.org/documentation** (https://www.eiffel.org/documentation) .

Problem 1: Inverse of a Matrix

Given a ‘n x n’ square matrix, write a program to find the inverse of the given matrix if it is invertible. Annotate the program with appropriate preconditions and postconditions.

Input:

- Input should be taken from STDIN.
- The first line contains a positive integer ‘n’ which is the order of the input square matrix.
- Next ‘n’ lines contain ‘n’ integers each which are the elements of the matrix.

Output:

- Print the output to the STDOUT.
- If the inverse of the matrix does not exist, then print “INVALID” (without quotes).
- Else print n lines where each line containing n elements which correspond to the inverse of the input matrix.

Examples:

Input:

```
2
1 0
0 1
```

Output:

```
1 0
```

```
0 1
```

Input:

```
3
0 1 0
0 5 0
2 1 3
```

Output:

```
INVALID
```

Input:

```
2
1 2
3 4
```

Output:

```
-2 1
1.5 -0.5
```

Problem 2: Stable Marriage Problem

There are given n men and n women. Each woman ranks all men in the order of her preference. Similarly, each man ranks all women in the order of his preference. Write a program to find a matching (i.e. a mapping between men and women) such that it is stable. A matching is said to be unstable if there are two people of opposite sex who would both rather have each other than their current partners i.e if there is a man A who prefers woman B over his current partner and B also prefers A over her current partner, then the marriage is unstable. Annotate the program with appropriate preconditions, postconditions and loop invariants.

For information about stable marriage problem, see this Wikipedia page: **https://en.wikipedia.org/wiki/Stable_marriage_problem (https://en.wikipedia.org/wiki/Stable_marriage_problem)**
You are allowed to refer to the algorithm mentioned in the above Wikipedia page. You are allowed to read this but start writing the program, only after you have understood the algorithm and close this page. Chances of two programs looking similar increases if you write program while keeping this page open.

Input:

- Input should be taken from STDIN
- The first line of the input contains a positive integer n (the number of marriages to be found).
- Next n lines contains are the women's preferences i.e. ith contains i and a permutation of {1, 2, …, n}

which denotes the preference of the ith woman i.e. if 2nd line contains “2 3 1 2”, then 2nd woman prefers 3rd man over 1st and 2nd and she prefers 1st over the 2nd. (Note that the first integer specifies that the next n integers are the preferences of the 2nd woman.)

- Similarly, the next n lines contain the men’s preferences in the same format.

Output:

- Print the output to the STDOUT.
- The output should contain n lines and each line contains an integer. If mth line contains integer w, then it means that the mth man should be married to wth woman.
- Note that there may be more than one sets of stable marriages possible, you have to output one of them.
- If the input is invalid then print “INVALID” (without quotes).

Examples:

Input:

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

Output:

```
3
2
1
4
```

Input:

```
7
1 3 4 2 1 6 7 5
2 6 4 2 3 5 1 7
3 6 3 5 7 2 4 1
4 1 6 3 2 4 7 5
5 1 6 5 3 4 7 2
6 1 7 3 4 5 6 2
7 5 6 2 4 3 7 1
1 4 5 3 7 2 6 1
```

```
2 5 6 4 7 3 2 1
3 1 6 5 4 3 7 2
4 3 5 6 7 2 4 1
5 1 7 6 4 3 5 2
6 6 3 7 5 2 4 1
7 1 7 4 2 6 5 3
```

Output:

```
4
5
1
3
7
6
2
```

General instructions:

1. Input/Output should be read from/written to STDIN/STDOUT only.
2. The output format must be strictly followed. No other output should be given, including trailing newlines. Evaluation will be done using automated scripts. Marks will be deducted for deviating from the format. In particular, do not print messages like “Enter the number” etc.
3. Use comments and meaningful variable names in the program.
4. Please ensure to annotate the program with as many non-trivial and trivial preconditions, postconditions, and invariants as possible.
5. If some precondition/postcondition/invariant fails due to an invalid input, the program should print “INVALID” to STDOUT (using exception handling capabilities of eiffel) before terminating. The Eiffel will exit the program with stack trace printed to STDERR.
6. Put the source files (.e files only, do not include any generated files) for the first program in ROLLNO_P1 and for the second program in ROLLNO_P2.
7. Keep the root class name as “APPLICATION” and the file containing the class “APPLICATION” should be named as application.e. You can have additional .e files in the same directory.
8. Your program should be able to compile using the following commands.

   ```
   cd ROLLNO_P1
   ec application.e
   ```

   Similarly for the 2nd problem.
9. Use the following command to compress and archive the files and generate a single .tgz file.

   ```
   tar -cvzf ROLLNO.tgz ROLLNO_P1 ROLLNO_P2
   ```

10. Only .tgz files will be accepted by Canvas.
11. If programs submitted by different students are found similar, all of these students will get an FR. “I don’t know how someone copied my program” is NOT a valid excuse. Please refer to the department’s plagiarism related policy at **https://cse.iith.ac.in/index-q=Academics-Plagiarism%20Policy.html** **(https://cse.iith.ac.in/index-q=Academics-Plagiarism%20Policy.html)**
12. Under no circumstances, the deadline (14Nov 11:59 pm) will be extended.
13. After the deadline, you will not be able to submit any file. So, please make sure you submit well in advance before the deadline.
14. For any clarifications, send a message through canvas to the instructor and all the TAs.