Submission deadline (on or before):

- 31st October, 2021, 10:00 PM

Policies for Submission and Evaluation:

- You must submit your assignment in the Eduserver course page, on or before the submission deadline.
- Your programs should be written in C language and should be compatible with the `gcc` compiler in Linux.
- During the evaluation, failure to execute programs without compilation errors may lead to zero marks for the evaluation.
- Detection of ANY malpractice related to the course can lead to awarding an F grade in the course.

Naming Conventions for Submission

- Submit the source file as a single C (.c) file (no other file types will be accepted on Eduserver). The file must be named as

  ```
  DCS_<ROLLNO>_<FIRST-NAME>_E.c
  ```

  (For example: *DCS_BxxyyyyCS_LAXMAN_E.c*). If you do not conform to the above naming conventions, your submission might not be recognized by our automated tools, and hence will lead to a score of 0 marks for the submission. So, make sure that you follow the naming conventions.

Standard of Conduct

- If we find two similar code submissions, both cases will be given zero marks, irrespective of any claim as to who wrote the original code or who submitted copied code. Please note that working with code available on the web is also considered plagiarism. Serious cases will be penalized up to F grade in the course irrespective of the marks scored in other quizzes/examinations, in accordance with the department's integrity policy (please refer to the course plan for the link).

1. Write a program to check whether a given undirected connected graph G, with no self-loops or multiple edges, has an Eulerian circuit or not. The program must also print the edges in the order that they appear in the Eulerian circuit, as per the algorithm given later in this document.

Input Format

The first line of the input contains an integer $n \in [1, 10^2]$, the number of vertices in G. The vertices are implicitly labeled from 0 to $n-1$.

The second line of the input contains an integer $m \in [0, 10^4]$, the number of edges in G.

The next m lines of the input each contain two *distinct* space-separated integers, say $u, v \in [0, n-1]$, that indicates the presence of the undirected edge (u, v) in G.

Output Format

If the graph is not Eulerian, the only output should be the string "Not Eulerian".

Otherwise, the output should contain m lines, corresponding to each of the m edges of G, listed in the order in which the edges are visited by the Eulerian circuit, as per the following algorithm.

Algorithm Outline

1. Determine if G is Eulerian and if it is not, output "Not Eulerian" and exit.
2. If G contains only one vertex, then the solution is straight forward. ▷ *Base condition*
3. Arbitrarily pick a vertex, say x, from $V(G)$. Let $(x, y_1), (x, y_2), \ldots, (x, y_k)$, where $k = \deg(x)$ is even, be the set of edges adjacent to x, listed in some arbitrary order.
4. Replace the edges (x, y_1) and (x, y_2) with the single edge (y_1, y_2). Similarly, replace (x, y_3) and (x, y_4) with the single edge (y_3, y_4) and so on up to the replacement of (x, y_{k-1}) and (x, y_k) with (y_{k-1}, y_k).
5. Finally, once all the replacements in the previous step are made, the vertex x will be isolated in the graph. Remove the vertex x. Now, you have a graph with one vertex less. Let us call the new graph G'.
6. *Recursively* construct an Eulerian circuit in the graph G'.
7. In the Eulerian circuit of G constructed recursively in Step 3, replace back the edge (y_1, y_2) with the pair of edges (y_1, x) and (x, y_2); (y_3, y_4) with the pair of edges $(y_3, x), (x, y_4)$; and so on upto the replacement of (y_{k-1}, y_k) with the pair (x, y_{k-1}) and (x, y_k). This gives an Eulerian circuit of G.
8. Finally, once the Eulerian circuit of the input graph is computed, print the edges in the circuit as specified in the Output Format section above.

Notes:

1. Only the broad steps of the algorithm are given above; the specifics of how to implement each step is left to the student's discretion.
2. The recursive steps of the algorithm would need to work with graphs that have multiple edges between the same pair of vertices and may also have self loops.

Sample Input and Output

Input

```
6
11
0 4
5 0
1 3
1 4
2 1
1 5
3 2
5 2
2 4
3 4
3 5
```

Possible Output

```
3 4
4 2
2 3
3 5
5 0
0 4
4 1
1 5
5 2
2 1
1 3
```

Assignment Evaluation Policy

1. Full marks (20) will be given if the program is not plagarized and passess all the test cases of the evaluation team.
2. Partial credits (15-19) will be given if the program is not plagiarized and works satisfactorily in most test cases.
3. If the program is not plagiarized and is a genuine attempt at the problem though not working satisfactorily, up to 7 marks will be given.
4. If the plagiarism check fails, then zero marks will be given.