Problem 1

Think about a Classmate device designed for your University, which can answer anything about your classmates quickly & efficiently. For example, Classmate starts its conversation by saying "Hello, What's your name?" and a student responds to it by saying "Jon Snow". Jon Snow (alias Checker) is a student of Class A, and he asks about Samwell Tarly (alias Mate) when prompted for "How can I help you, Jon Snow". Classmate should identify Jon Snow belongs to Class A and search Samwell Tarly with in Class A and respond if found by saying "Samwell Tarly is from Tennessee (home town) and he likes Basketball (likes)". If it cannot find Samwell Tarly, it should prompt "Samwell Tarly not found in Class A. Please ask about students in your class."

Each student has the following information stored at the beginning of the program.

First Name, Last Name, Likes, Home Town, Class

For example,

First Name	Last Name	Likes	Home Town	Class
Jon	Snow	Football	New York	A
Samwell	Tarly	Basketball	Tennessee	A
Arya	Stark	Hunting	New York	B
Daenerys	Targaryen	Flying	Dragon Stone	A
..	..	..	..	..

System: Hello, What's your name?
User: Jon Snow
System: Whose information would you like to know, Jon Snow?
User: Samwell Tarly
System: Samwell Tarly is from Tennessee and he likes Basketball.
User: Arya Stark
System: Arya Stark not found in Class A. Please ask about students in your class.
User: Daenerys Targaryen
…
…
…
User: Exit
System: Bye.

Tasks:

- Design an efficient way to store the information of all the students in the University.
- Matching Strategy to quickly & efficiently identify student (checker)
- Searching strategy to identify Mate with in the Class.

Note: Assume the strength of University is greater than 10,000 to test the efficiency of your algorithm.

Please comment your code and explain your methodology clearly

Problem 2

Given an **N** by **N** matrix, and a string (s1) (length may be less than NXN), fill the matrix with s1 row by row going left to right. Fill the matrix completely by repeating the string. Now, given another string (s2) from the user and display all the occurrences of s2 in the matrix by traversing only:

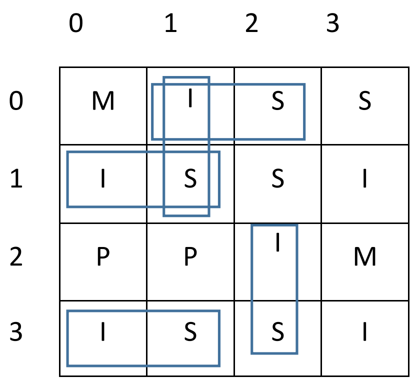
1. Left to right.

2. Top to bottom.

If there are no such occurrences, output "**No matches**".

Here is an example, **N = 4, s1 = "MISSISSIPPI" and s2 ="IS"**

Matrix looks like this



	0	1	2	3
0	M	I	S	S
1	I	S	S	I
2	P	P	I	M
3	I	S	S	I

Output:

(0, 1) to (0, 2) -> left to right

(0, 1) to (1, 1) -> top to bottom

(1, 0) to (1, 1) -> left to right

(2, 2) to (3, 2) -> top to bottom

(3, 0) to (3, 1) -> left to right

Please comment your code and explain your methodology clearly