Assignment 2 - Pathfinder

In this assignment, a file named “output.txt” will be produced by reading the information in the “input.txt” file located in the same directory as the application. The application will not receive any information from the console interface. After it is opened, it should read the “input.txt” file in the folder where it is located and write the result to “output.txt”. Grading is done according to the presence of “output.txt”. Homework submission will be made as a single ".cpp" extension file. The file name must be your student number. (Sample file to send: 1306000001.cpp)

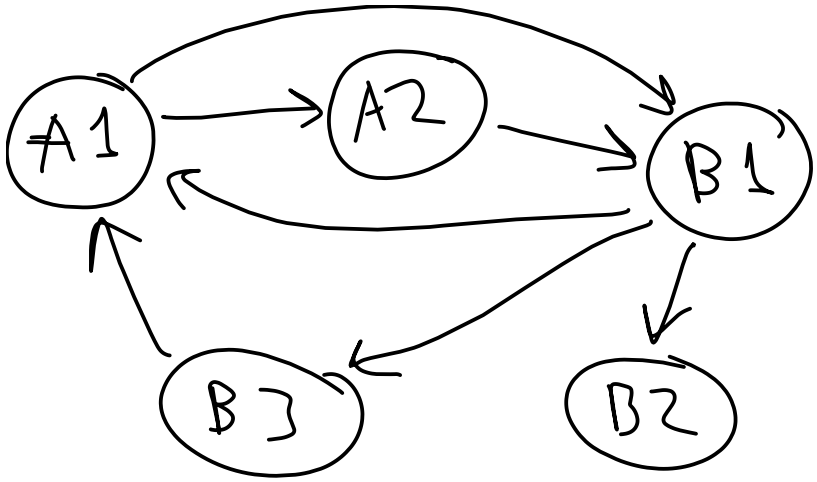
The assignment file should be compiled with GCC or Microsoft C ++ Compiler. At the top of the homework file**, your name, student number and development environment** information should be included as a description. In addition, additional explanations about the homework should be given here. Other explanations in the code are not taken into account in grading.

Ex:

```
//////////////////////////////////////////
// Özgür Can TURNA
// 1306000001
// Date: 2021.05.05
// IDE: Visual Studio 2019
//////////////////////////////////////////
// Annotations will be made here.
//////////////////////////////////////////
```

Problem:

A graph definition is given in the "input.txt" file. In this graph definition, nodes are expressed as a combination of uppercase letters and numbers. (Ex: A1, A2, B1, B2, B3) The letter indicates the type of the node and the number indicates the amount of this type. Paths (transitions) in graph are written as pairs of nodes. (Ex: A1-> A2, A1-> B1, A2-> B1, B1-> B2, B1-> B3, B1-> A1, B3-> A1) These data define the graph. The visual version of the graph will be as follows.



Next lines in the file will contain Word for the paths traversed on this graph.

Ex: AAB, ABB, ABBAAB

Determining whether each desired path definition is a traceable path on the graph, and write [YES], [NO] next to the path in the output.txt file. There may be cyclical paths in the file, that is, a transition from A1 to B1 and from B1 to A1. In this case, your algorithm should be able to scan the path without going into an endless loop. The path definition can be searched by starting from any node.

Below are sample input.txt and output.txt file contents.

Example 1:

input.txt

```
A1, A2, B1, B2, B3
Links:
A1-> A2
A1-> B1
A2-> B1
B1-> B2
B1-> B3
B1-> A1
B3-> A1
Paths:
AAB
ABB
ABBAAB
AAAB
```

output.txt

```
AAB [YES]
ABB [YES]
ABBAAB [YES]
AAAB [NO]
```

Example 2:

input.txt

```
A1, B1, C1, D1, D2, E1
Links:
A1-> C1
C1-> B1
D1-> C1
D1-> D2
D1-> E1
D2-> E1
E1-> C1
Paths:
CBA
CB
ACB
DEC
CDE
DDEC
```

output.txt

```
CBA [NO]
CB [YES]
ACB [YES]
DEC [YES]
CDE [NO]
DDEC [YES]
```