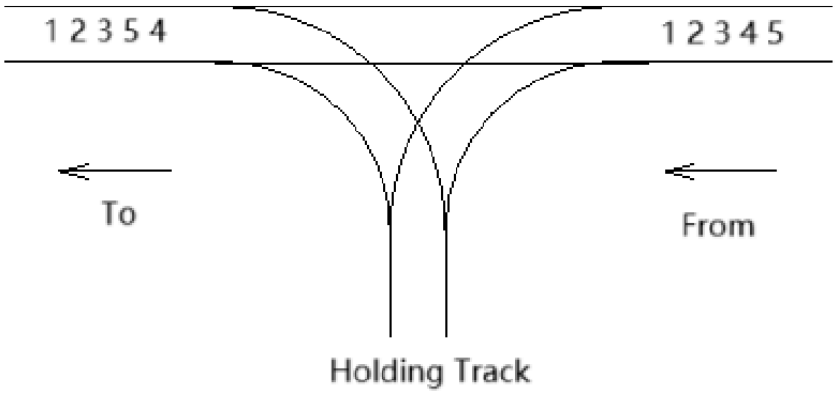
Rails Station Problem

Problem Background

In a small mountain railway station, train cars arrive in order (1, 2, 3...) and need to be sorted and every step should be recorded before departure.



e.g.

1 2 3 4 5 0	Train 1 passing through Train 2 passing through Train 3 passing through Train 4 passing through Train 5 passing through Success
1 2 3 5 4 0	Train 1 passing through Train 2 passing through Train 3 passing through Push train 4 to holding track Current holding track: 4 Train 5 passing through Moving train 4 from holding track Success

In the 1st example, trains are required to sort into order 1 2 3 4 5, and trains arrive in order 1 2 3 4 5. Therefore, every train is directly passed through to meet this requirement.

In the 2nd example, trains are required to sort into order 1 2 3 5 4, and trains arrive in order 1 2 3 4 5. Therefore

1. Train 1, 2 and 3 are directly passed through
2. Train 4 should move in to the holding track (4 in)
3. Train 5 is directly passed through
4. Train 4 move out from the holding track (4 out)

What is Stack Sorting?

1. Create a holding stack
2. While input stack is NOT empty:
    a. Pop an element from input stack, called **temp**
    b. While holding stack is NOT empty and top of holding stack is greater than temp
    c. Pop from holding stack and push it to the input stack
    d. Push temp in holding stack
3. The sorted numbers are in holding stack

Station Layout

- Main Track:
    - Trains initially arrive here in order
- Holding Track:
    - Works like a stack (Last-In-First-Out)
    - Used to temporarily store and rearrange trains

Problem Objective

Help the train dispatcher solve these challenges:

- Rearrange incoming trains to match a desired order
- Use only the holding track for train movement
- Determine if perfect sorting is possible
- You are required to implement using yacc (bison) and lex (flex)

Input

- A sequence of train numbers to be sorted in desired order
- Input ends with 0
- Total number of the train will not more than 20

Output

1. Step-by-Step Sorting Process
    - Show each train movement
    - Display holding track contents when trains are added
2. Final Result
    - Show **Success** if sorting is completed
    - Detailed error message if sorting is impossible

Sorting Rules

1. Trains can move through the main track in order (1, 2, 3...)
2. Trains can be temporarily stored in the holding track
3. Only one train can move at a time
4. Holding track follows stack principles

Tokens Definition

- NUM ::= [1-9][0-9]*
- END ::= 0

Grammar Specification

- startsymbol ::= numbers END
- numbers ::= numbers number | number
- number::= NUM

Step Message

- Moving train direct from main track

  Train <i> passing through

- Moving train from holding track

  Moving train <i> from holding track

- Moving train to the holding track

  Push train <i> to holding track

  Current holding track: <trains in the holding track>

Error Message

- Cannot move trains to match desired order
  - No more train is in the holding track

  Error: Impossible to rearrange

There is no any train in the holding track

- The first train in the holding track is not needed train

  Error: Impossible to rearrange

  The first train in the holding track is train <i> instead of train <j>

- Trains remain in holding track after sorting attempt

  Error: There is still existing trains in the holding track

Spec of the Stack for reference

```
%{
#include <stdio.h>
#define STACK_SIZE 20

int yylex();
void yyerror(const char* message);

struct stack {
   int data[STACK_SIZE];
   int top;
};
typedef struct stack stack_t;
stack_t stack;
int train;

int isEmpty(); // to check if the stack is empty
void push(int i);
int top();
int pop();
void dump(); // to dump (or print) all the data in stack
%}
```

Sample Inputs and Outputs

Input	Output
1 2 3 4 5 0	Train 1 passing through Train 2 passing through Train 3 passing through Train 4 passing through Train 5 passing through Success
5 4 1 2 3 0	Push train 1 to holding track Current holding track: 1 Push train 2 to holding track Current holding track: 1 2 Push train 3 to holding track Current holding track: 1 2 3 Push train 4 to holding track Current holding track: 1 2 3 4 Train 5 passing through Moving train 4 from holding track Error: Impossible to rearrange The first train in the holding track is train 3 instead of train 1
1 2 3 5 0	Train 1 passing through Train 2 passing through Train 3 passing through Push train 4 to holding track Current holding track: 4 Train 5 passing through Error: There is still existing trains in the holding track
1 2 3 1 0	Train 1 passing through Train 2 passing through Train 3 passing through Error: Impossible to rearrange There is no any train in the holding track

Note: If `fprintf()` does not output normally, you can try using `printf()` instead.