Hackeando o sistema

Descrição

A história é sobre um hacker, uerlei, que está tentando invadir um sistema de um computador de outra pessoa, uiliã, e para isso precisa saber em que tipo de estrutura é baseado o armazenamento e execução de processos desse sistema. Com as habilidades de hacker dele, já conseguiu acesso a essa estrutura mas somente pode fazer dois tipos de operação sobre ela:

1. *PUSH X*: que coloca o número de um processo X na estrutura
2. *POP: que retorna o número de processo que é retirado da estrutura*

Uerlei executou uma sequencia de operacoes sobre a estrutura mas não conseguiu descobrir que tipo de estrutura de dados o sistema é baseado: pilha, fila, nenhuma ou indeterminado. Então ele pediu sua ajuda.

Uerlei enviou por email para você a sequência de comandos que ele executou sobre o sistema junto com os retornos das operações do tipo *POP* e pediu que você determinasse a estrutura. Porém, nem tudo são flores. Além dessa dificuldade toda, no momento que uerlei enviou a sequência para você através da internet o sistema de uiliã identificou o vazamento de dados sensíveis e, na tentativa de dificultar o vazamento, embaralhou a ordem dos pacotes na mensagem.

Mais detalhadamente: o protocolo de internet havia quebrado a sequência em vários pacotes menores que contém partes da sequência original. Cada pacote é composto de 4 coisas: *COD_ESQ, CMD, ARG e COD_DIR.*

CMD é um comando da sequência, ou seja, é a palavra *POP* ou *PUSH. ARG* é o número do processo inserido na estrutura se *CMD* for *PUSH*, ou é o número do processo que foi retornado da estrutura caso o comando tenha sido *POP*. *COD_ESQ* e *COD_DIR* são códigos gerados pelo próprio protocolo e servem para determinar o pacote que fica à frente e o que fica atrás desse pacote. Se um pacote A, por exemplo, tiver seu *COD_DIR* igual ao *COD_ESQ* de um pacote B então o pacote A vem imediatamente antes do pacote B na sequência de envio. Não existem dois pacotes diferentes com mesmo *COD_ESQ* e não existem dois pacotes diferentes com mesmo *COS_DIR*.

Para ilustrar considere um pacote no seguinte formato:

[*COD_ESQ, CMD, ARG, COD_DIR*]

Agora considere os seguintes pacotes embaralhados:

C : [28, POP, 20, 17]
A : [6, PUSH, 20, 28]
B : [17, POP, 10, 56]
D : [8, PUSH, 10, 6]

Colocando os pacotes em ordem (observe como eles ‘se unem pelas pontas’):

[8, PUSH, 10, 6] => [6, PUSH, 20, 28] => [28, POP, 20, 17] => [17, POP, 10, 56]

Ou seja, a ordem é:

D => A => C => B

Logo a sequência original de comandos executados por uerlei foi:

PUSH 10
PUSH 20
POP 20
POP 10

Ou seja, uerlei inseriu o processo 10, depois o processo 20, e quando quando executou uma retirada o sistema retornou o 20 e quando fez uma segunda retira o sistema retornou 10. Com isso é visível que o sistema funciona como uma estrutura de dados do tipo **pilha.**

Sua tarefa é, dado a sequência embaralhada de pacotes recebida, determinar qual é o tipo de estrutura de dados que é similar ao funcionamento do sistema: pilha, fila, nenhuma dessas ou ambas.

Especificações de entrada

A primeira linha da entrada contém um inteiro N sendo a quantidade de pacotes recebidos. Cada uma das N linhas seguintes possui quatro coisas: COD_ESQ, CMD, ARG, COD_DIR representando os componentes do pacote de acordo como explicado na descrição.

Especificações de saída

A saída consiste de uma única linha contendo uma das quatro mensagens: “pilha” (caso o sistema de uiliã só possa ser uma pilha), “Fila” (Caso só possa ser uma fila), “nenhuma” (Caso não possa ser nem fila e nem pilha) ou “ambas” (caso possa ser tanto uma fila quanto uma pilha).

Exemplos

Exemplo de entrada	Exemplo de saída
4 28 POP 20 17 6 PUSH 20 28 17 POP 10 56 8 PUSH 10 6	pilha

Exemplo de entrada	Exemplo de saída
2 11 PUSH 5 22 22 POP 5 33	ambas

Exemplo de entrada	Exemplo de saída
10 70 PUSH 34 80 20 POP 5 30 10 PUSH 5 20 60 POP 13 70 90 POP 55 100 80 POP 8 90 40 PUSH 8 50 30 PUSH 13 40 100 POP 34 110 50 PUSH 55 60	fila