Податоци за вработените во компанијата IOSolutions се чуваат во две еднострано поврзани листи. Во првата листа се чуваат податоците за вработените кои се дел од тимот Dev,

а додека пак во втората се чуваат податоците за вработените кои се дел од тимот Qa.

И кај двете листи, во секој од јазлите се чуваат број за идентификација (id) и возраста на вработениот за кој се однесува тој јазел.

Со зголемување на обемот на работа, потребно е да се зголеми бројот на членови во Dev тимот.

За таа цел, главниот менаџер решил да го отстрани најмладиот член од Qa тимот и да го додаде кон Dev тимот.

Тоа значи дека потребно е од листата каде се чуваат членовите на Qa тимот да се отстрани (избрише) најмладиот член (доколку има повеќе, последниот таков член),

и истиот да се додаде на средина на листата каде што се чуваат членовите на Dev тимот.

Влез:

Во првиот ред е даден бројот на вработени кои се дел од Dev тимот. Во вториот ред е даден бројот на вработени кои се дел од Qa тимот.

Во следните редови се дадени паровите податоци за секој вработен, одделени со празно место, во формат id возраст.

Излез:

Во првиот ред id на сите вработени кои се дел од Dev тимот. Во вториот ред id на сите вработени кои се дел од Qa тимот.

Забелешка:

Даден е целосниот код на структурата која што треба да се користи.

Потребно е да направат дополнувања во main функцијата и да се имплементирана void alterTeams(SLL devTeam, SLL qaTeam) функцијата.

Притоа, бришењето треба да биде имплементирано како бришење на цел јазел, а додавањето како додавање на цел јазел.

Промените (бришење/додавање елемент) не треба да се однесуваат на информациите во самите јазли туку во промени на врските помеѓу јазлите.

1.Тест пример

2

3

16100 20

17200 30

19898 20

14203 35

18100 19

16100 18100 17200

19898 14203

2. Тест пример

3

3

16100 20

17200 30

18112 45

14102 19

19203 35

18900 20

16100 17200 14102 18112

19203 18900

Corrected test cases (two errors removed):

Test case:

2

3

16100 30

17890 40

18677 25

12156 20

10900 30

Result:

16100 12156 17890

18677 10900

Test case:

4

3

19000 34

18788 45

17677 56

12390 23

13000 25

17800 24

14500 22

Result:

19000 18788 14500 17677 12390

13000 17800

Test case:

5

3

12300 34

13000 45

19002 56

18907 23

17900 45

15690 22

13989 24

12111 25

Result:

12300 13000 19002 15690 18907 17900

13989 12111

Test case:

3

3

12360 34

19000 45

18950 56

15678 32

13456 24

16788 29

Result:

12360 19000 13456 18950

15678 16788

import java.util.Scanner;

class SLLNode {

int id;

int age;

SLLNode succ;

public SLLNode(int id, int age, SLLNode succ) {

this.id = id;

this.age = age;

this.succ = succ;

}

@Override

public String toString() {

return String.valueOf(id);

}

}

class SLL {

SLLNode first;

public SLL() {

this.first = null;

}

@Override

public String toString() {

StringBuilder ret = new StringBuilder();

if (first != null) {

SLLNode tmp = first;

ret.append(tmp).append(" ");

while (tmp.succ != null) {

tmp = tmp.succ;

ret.append(tmp).append(" ");

}

} else

ret = new StringBuilder("NO ELEMENTS");

return ret.toString();

}

}

public class Company {

public static void alterTeams(SLL devTeam, SLL qaTeam) {

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int devNum = Integer.parseInt(scanner.nextLine());

int qaNum = Integer.parseInt(scanner.nextLine());

SLL devTeam = new SLL();

SLL qaTeam = new SLL();

for (int i = 0; i < devNum; i++) {

String line = scanner.nextLine();

String[] parts = line.split("\\s+");

//vashiot kod tuka

}

for (int i = 0; i < qaNum; i++) {

String line = scanner.nextLine();

String[] parts = line.split("\\s+");

//vashiot kod tuka

}

alterTeams(devTeam, qaTeam);

}

}