Trains

Fie o retea de transport feroviar ce acopera N orase. Exista exact $N-1$ rute directe intre anumite perechi de orase astfel incat rutele sa formeze structura unui arbore avand orasul 1 drept radacina. Pentru fiecare oras i cu exceptia orasului 1 se cunoaste F_i, parintele imediat al nodului i.

In fiecare secunda, incepand cu secunda 0, un tren pleaca din orasul 1 pe cel mai scurt drum catre alt oras. Toate trenurile au aceeasi viteza si toate rutele directe sunt construite in asa fel incat dureaza exact 1 secunda ca trenurile sa se deplaseze intre doua orase ce impartasesc o ruta directa.

Pentru fiecare oras se cunoaste cel mai tarziu timp T_i cand primul tren trebuie sa ajunga in oras, conform dorintelor clientilor.

Trebuie sa ajuti dispecerul de trenuri si sa determine daca exista o modalitate de a directiona fiecare tren astfel incat cel putin unul dintre ele ajunge sau trece prin orasul i inainte de sau la timpul T_i (destinatiile finale ale trenurilor sunt irelevante).

Trebuie sa implementezi o functie numita solution care primeste urmatoarele argumente:

`int N` : numarul de orase;

`int *F` : un vector indexat de la 0 cu $N+1$ elemente ($F[0]$ si $F[1]$ sunt nedefinite) reprezentand harta;

`int *T` : un vector indexat de la 0 cu $N+1$ elemente ($T[0]$ este nedefinite) reprezentand dorintele clientilor.

si returneaza `int` : `1` daca toti clientii pot fi multumiti sau `0` in caz contrar.

Standard input

Codul sablon citeste datele de intrare in format binary. Nu ar trebui sa (poti) edita partea aceasta a codului.

Standard output

Codul sablon apeleaza functia ta de mai multe ori in acelasi test, de fiecare data posibil cu alta retea de transport feroviar. De fiecare data, functia va trebui sa returneze raspunsul correct.

Testing

Aceasta este o problema ineractiva. Nu trebuie sa cititi sau afisati date.

Codul care va este pus la dispozitie citeste din fisierul de intrare in urmatorul format:=

- Prima linie contine un numar `C` , de cate ori este apelata functia `solution` .
- `C` blocuri de date similare, descriind parametri functiei. Fiecare are formatul:

```
N
F[2] F[3] F[4] ... F[N]
T[1] T[2] T[3] ... T[N]

```

- Observati ca `F[0]` , `F[1]` si `T[0]` nu apar in fisier deoarece sunt nedefiniti si nu ar trebui folositi pentru rezolvarea problemei.

Constraints and notes

- Functia va fi apelata de maxim 50 de ori.
- $1 \leq N \leq 1.5 \cdot 10^5$
- $1 \leq S \leq 5 \cdot 10^6$, $S =$ suma tuturor N -urilor dintr-un test
- $0 \leq T_i \leq 10^9$ pentru fiecare $1 \leq i \leq N$

Subtasks

Testele sunt punctate ***individual***.

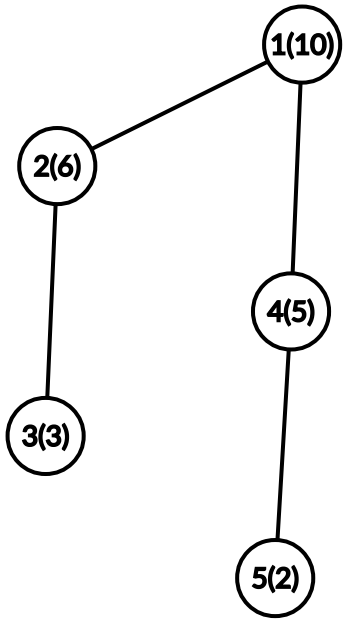
Subtask	Procent din teste	Restrictii aditionale
1	10%	$N \leq 50$
2	10%	$50 < N \leq 10^3$
3	20%	$10^3 < N \leq 10^4$
4	30%	$10^4 < N \leq 5 \cdot 10^4$
5	30%	-

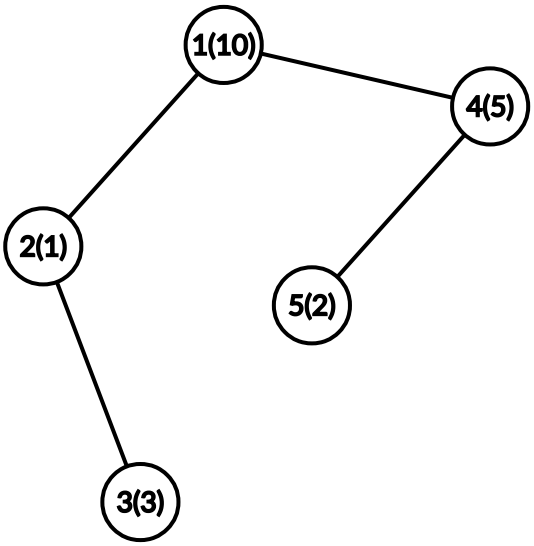
Examples

Parametri	Return

`N = 5`
`F = - - 1 2 1 4`
`T = - 10 6 3 5 2`
	`1`

`N = 5`
`F = - - 1 2 1 4`
`T = - 10 1 3 5 2`
	`0`


1(10)
2(6)
4(5)
3(3)
5(2)


1(10)
4(5)
2(1)
5(2)
3(3)