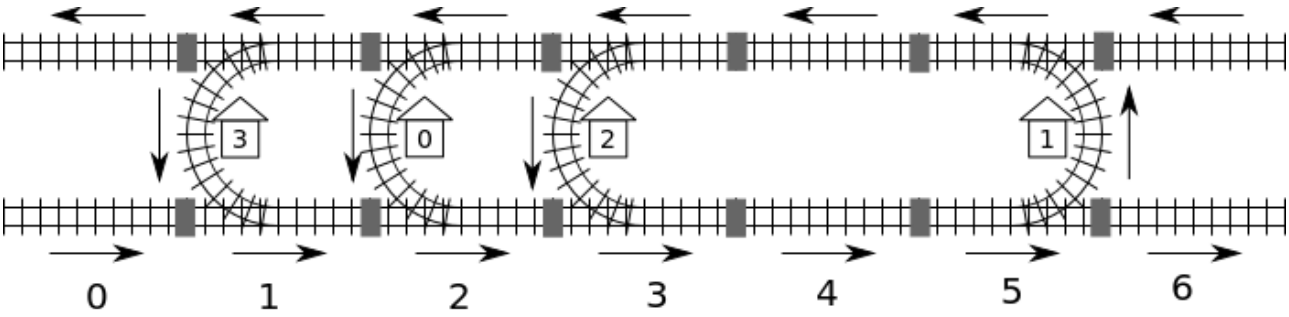
Geležinkelis

Taivano salos vakarinę ir rytinę pakrantes jungia geležinkelio linija. Ji suskirstyta į m blokų, sunumeruotų nuo 0 iki $m-1$ iš vakarų į rytus. Kiekviename bloke įrengti dveji vienakrypčiai bėgiai: vakarų krypties bėgiai šiaurėje (jais galima važiuoti į vakarus) ir rytų krypties (galima važiuoti į rytus) — pietuose. Kai kuriuose blokuose tarp šiaurinių ir pietinių bėgių yra įrengtos traukinių stotys.

Blokai būna trijų rūšių: `C` rūšies, `D` rūšies ir *tušti*. `C` rūšies blokuose yra stotys, į kurias įvažiuojama iš šiaurinių bėgių, o iš jų išvažiuojama į pietinius. `D` rūšies blokuose yra stotys, į kurias įvažiuojama iš pietinių bėgių, o iš jų išvažiuojama į šiaurinius. Tuščiuose blokuose stočių nėra.

Žemiau esančiame pavyzdyje 0-asis, 4-asis ir 6-asis blokai yra tušti, 1-asis, 2-asis ir 3-iasis blokas yra `C` rūšies, o 5-asis —— `D` rūšies. Gretimuose blokuose esantys bėgiai jungiasi *jungtimis*, kurias pavyzdyje atitinka pilki stačiakampiai.



Iš viso yra n stočių, sunumeruotų nuo 0 iki $n-1$. Galite tarti, kad *iš bet kurios stoties galima pasiekti bet kurią kitą stotį* keliaujant geležinkelio linija. Nagrinėtame pavyzdyje iš 0-osios stoties galima pasiekti 2-ąją stotį pradedant 2-ajame bloke, iš jos pietiniais bėgiais keliaujant per 3-ąjį ir 4-ąjį, 5-ajame bloke apsisukant prie 1-osios stoties, keliaujant per 4-ąjį bloką šiauriniais bėgiais ir galiausiai pasiekiant 2-ąją stotį 3-iajame bloke.

Gali būti taip, kad vieną stotį pasiekti iš kitos galima keliais maršrutais. Trumpiausias maršrutas yra tas, kuriuo keliaujant yra pravažiuojama mažiausia jungčių ir jo ilgis vadinamas *atstumu*. Nagrinėtame pavyzdyje atstumas nuo 0-osios stoties iki 2-osios yra 5, nes trumpiausias maršrutas, einantis per blokus 2–3–4–5–4–3, eina per 5 jungtis.

Geležinkelio liniją prižiūri kompiuterinė sistema. Kartą, trumpam nutrūkus elektros tiekimui, kompiuteris nebežino, kuriuose ir kokių rūšių blokuose yra stotys. Tačiau jis vis dar žino, kuriame bloke yra 0-oji stotis ir kad tas blokas yra `C` rūšies. Kompiuteris gali teirautis, koks yra atstumas nuo bet kurios stoties iki bet kurios kitos. Nagrinėtame pavyzdyje kompiuteriui pasiteiravus, koks yra atstumas nuo 0-osios stoties iki 2-osios, jam būtų pateikiamas atsakymas 5.

Užduotis

Parašykite funkciją `findLocation`, kuri nustatytų kiekvienos stoties bloko numerį ir rūšį:

- `findLocation(n, first, location, stype)`
  - `n`: stočių skaičius.
  - `first`: bloko, kuriame yra 0-oji stotis, numeris.
  - `location`: n ilgio masyvas, kuriame reikia pateikti stočių padėtis (bloko, kuriame yra i-oji stotis, numeris turi būti įrašytas į `location[i]`).
  - `stype`: n ilgio masyvas, kuriame reikia pateikti stočių rūšis (bloko, kuriame yra i-oji stotis, rūšis turi būti įrašyta į `stype[i]`: 1, jei rūšis yra `C`, 2 — jei `D`.)

Jūsų funkcija gali kviesti funkciją `getDistance`:

- `getDistance(i, j)` grąžina atstumą nuo i-osios stoties iki j-osios. `getDistance(i, i)` grąžina 0. `getDistance(i, j)` grąžins -1, jei i arba j netenkina sąlygos $0 \le i, j \le n-1$.

Dalinės užduotys

Visose dalinėse užduotyse blokų skaičius m neviršija $1\,000\,000$. Kai kuriose dalinėse užduotyse funkcijos `getDistance` iškvietimų skaičius yra ribojamas. Šis ribojimas priklauso nuo dalinės užduoties. Programai jį viršijus bus rodoma 'wrong answer'.

dalinė užduotis	taškai	n	getDistance ribojimas	papildoma informacija
1	8	$1 \le n \le 100$	neribojama	Visos stotys, išskyrus 0-ąją, yra `D` rūšies blokuose.
2	22	$1 \le n \le 100$	neribojama	Visos stotys, esančios į rytus nuo 0-osios, yra `D` rūšies blokuose; visos stotys į vakarus yra `C` rūšies blokuose.
3	26	$1 \le n \le 5\,000$	$n(n-1)/2$	Nėra kitų ribojimų.
4	44	$1 \le n \le 5\,000$	$3(n-1)$	Nėra kitų ribojimų.

Reikalavimai

Pateikite vieną failą, pavadintą `rail.c`, `rail.cpp` arba `rail.pas`. Jame turi būti procedūra `findLocation`. Jei naudojate C/C++, įtraukite antraštinį failą `rail.h`.

Programuojantiems C/C++

```
void findLocation(int n, int first, int location[], int stype[]);
```

Programuojantiems Paskaliu

```
procedure findLocation(n, first : longint; var location,
stype : array of longint);
```

Vertintojo funkcija `getDistance`:

C/C++ kalba

```
int getDistance(int i, int j);
```

Pascal kalba

```
function getDistance(i, j: longint): longint;
```

Pavyzdinis vertintojas

Pavyzdinis vertintojas skaito duomenis tokiu formatu:

- 1-oji eilutė: dalinės užduoties numeris.
- 2-oji eilutė: `n`.
- eilutės $3+i$, ($0 \le i \le n-1$): `stype[i]` (1, jei rūšis `C`, 2 – jei `D`), `location[i]`.

Pavyzdinis vertintojas pateiks `Correct`, jei jūsų programos apskaičiuoti `location[0]`, ..., `location[n-1]` ir `stype[0]`, ..., `stype[n-1]` sutaps su pateiktais duomenimis. Kitu atveju jis pateiks `Incorrect`.