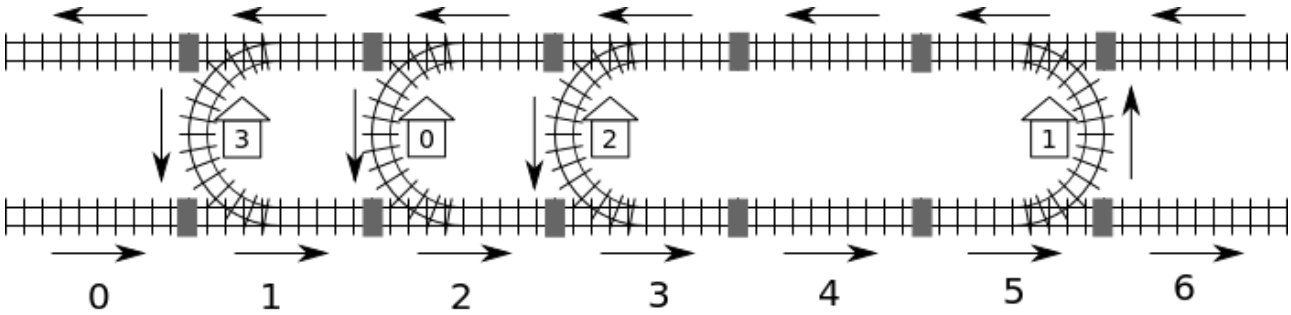
რკინიგზა

ტაივანში არის დიდი სარკინიგზო ხაზი, რომელიც აერთებს კუნძულის დასავლეთ და აღმოსავლეთ სანაპიროებს. რკინიგზის ხაზი შეიცავს m რაოდენობის ბლოკს. ბლოკები თანმიმდევრულადაა გადანომრილი 0-დან $(m-1)$-მდე. გადანომვრა იწყება აღმოსავლეთ ბოლოდან. ყოველ ბლოკს ჩრდილოეთიდან გააჩნია დასავლეთისკენ მიმართული ცალმხრივი გზა, ხოლო სამხრეთიდან გააჩნია აღმოსავლეთისკენ მიმართული ცალმხრივი გზა.

სულ არსებობს სამი ტიპის ბლოკი. *C* ტიპის ბლოკს გააჩნია სადგური, რომელშიც შესაძლებელია შეხვიდე ჩრდილოეთის ხაზიდან და გადახვიდე სამხრეთის ხაზზე. *D* ტიპის ბლოკს გააჩნია სადგური, რომელშიც შესაძლებელია შეხვიდე სამხრეთის ხაზიდან და გადახვიდე ჩრდილოეთის ხაზზე. *ცარიელი* ტიპის ბლოკი სადგურს არ შეიცავს. მაგალითად, ქვემოთ მოყვანილ ნახაზზე 0 ნომრის მქონე ბლოკი ცარიელი ტიპისაა, 1 ნომრის მქონე ბლოკი C ტიპისაა, ხოლო 5 ნომრის მქონე ბლოკი D ტიპისაა. ბლოკები ერთმანეთთან შეერთებულია ჰორიზონტალურად. მეზობელი ბლოკები ერთმანეთთან დაკავშირებულია *კონექტორებით*, რომლებიც ნახაზზე ნაჩვენებია მუქი მართკუთხედებით.



სარკინიგზო სისტემაში n ცალი სადგური გადანომრილია 0–დან $(n-1)$–მდე. იგულისხმება, რომ *ჩვენ შეგვიძლია მოვხვდეთ ნებისმიერი სადგურიდან ნებისმიერ სადგურში* მხოლოდ სარკინიგზო ხაზების საშუალებით. მაგალითად, სადგურ 0–დან სადგურ 2–მდე შეიძლება ასე მივიდეთ: დავიწყოთ 2 ნომრის მქონე ბლოკიდან, გავიაროთ 3 და 4 ბლოკები სამხრეთის ხაზზე, შემდეგ პირველი სადგურის საშუალებით გადავიდეთ ჩრდილოეთის ხაზზე და ბლოკი 4–ის გავლის შემდეგ მივაღწევთ მე-2 სადგურამდე ბლოკი 3–დან.

რადგან ორ სადგურს შორის არსებობს ბევრი შესაძლო მარშრუტი, მანძილი ერთი სადგურიდან მეორემდე განვსაზღვროთ, როგორც კონექტორების *მინიმალური* რაოდენობა, რომელზეც მარშრუტი გადის. მაგალითად, უმოკლესი მარშრუტი 0 სადგურიდან მე-2 სადგურამდე გადის ბლოკებზე 2-3-4-5-4-3 და გაივლის 5 კონექტორს, მაშასადამე მანძილი არის 5.

სარკინიგზო სისტემას კომპიუტერი მართავს. სამწუხაროდ, დენის გამორთვის შემდეგ მან აღარ იცის სადაა სადგურები და რომელი ტიპის ბლოკებში იმყოფებიან ისინი. ერთადერთი ფაქტი, რომელიც კომპიუტერმა იცის, არის ის,

რომ ბლოკის ნომერი, რომელშიც მოთავსებულია სადგური 0, ყოველთვის C ტიპისაა. საბედნიეროდ, კომპიუტერს შეუძლია დასვას შეკითხვა მანძილის შესახებ ნებისმიერ ორ სადგურს შორის. მაგალითად, 'როგორია მანძილი 0 სადგურიდან მე-2 სადგურამდე?' და პასუხად მიიღებს 5-ს.

ამოცანა

თქვენ უნდა შექმნათ ფუნქცია `findLocation`, რომელმაც თითოეული სადგურისათვის უნდა განსაზღვროს ბლოკის ნომერი და ბლოკის ტიპი.

- `findLocation(n, first, location, stype)`
  - `n`: სადგურების რაოდენობა.
  - `first`: ბლოკის ნომერი 0 ნომრის მქონე სადგურისათვის.
  - `location`: n ზომის მქონე მასივი; i-ური სადგურისათვის მისი შესაბამისი ბლოკის ნომერი უნდა მოათავსოთ `location[i]`-ში.
  - `stype`: n ზომის მქონე მასივი; i-ური სადგურისათვის მისი შესაბამისი ტიპის ნომერი უნდა მოათავსოთ `stype[i]`-ში: 1 აღნიშნავს C ტიპს და 2 აღნიშნავს D ტიპს.

თქვენ შეგიძლიათ გამოიძახოთ ფუნქცია `getDistance`, რომელიც დაგეხმარებათ დაადგინოთ სადგურთა მდებარეობა და ტიპები.

- `getDistance(i, j)` აბრუნებს მანძილს `i`-ური სადგურიდან `j`-ურ სადგურამდე. `getDistance(i, i)` აბრუნებს 0-ს. `getDistance(i, j)` აბრუნებს -1, თუ `i` ან `j` არ არის დიაპაზონში $0 \leq i, j \leq n - 1$.

ქვეამოცანები

ყველა ქვეამოცანაში ბლოკების რაოდენობა m არ აღემატება 1,000,000-ს. ზოგიერთ ქვეამოცანაში შეზღუდულია `getDistance`-ს გამოძახება. შეზღუდვა მითითებულია ქვეამოცანაში. თქვენი პროგრამა მიიღებს 'wrong answer'-ს, თუკი გადააჭარბებს ამ მოთხოვნას.

ქვეამოც.	ქულა	n	getDistance გამოძახება	შენიშვნა
1	8	$1 \leq n \leq 100$	ულიმიტო	0 ნომრის მქონე სადგურის გარდა, ყველა სადგური D ტიპის ბლოკშია.
2	22	$1 \leq n \leq 100$	ულიმიტო	ყველა სადგური 0 ნომრის მქონე სადგურის მარჯვნივ არის D ტიპის და ყველა სადგური 0 ნომრის მქონე სადგურის მარცხნივ არის C ტიპის.
3	26	$1 \leq n \leq 5,000$	$n(n-1)/2$	შეზღუდვის გარეშე
4	44	$1 \leq n \leq 5,000$	$3(n-1)$	შეზღუდვის გარეშე

იმპლემენტაციის დეტალები

თქვენ უნდა წარმოადგინოთ ერთი ფაილი `rail.c`, `rail.cpp` ან `rail.pas`. ეს ფაილი გამოიძახებს `findLocation`-ს, ისე როგორც ეს ზემოთაა აღწერილი. გარდა ამისა, თქვენ უნდა გამოიყენოთ `rail.h` ფაილი (C/C++)-ში იმპლემენტაციისათვის.

C/C++ პროგრამა

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

Pascal პროგრამა

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

`getDistance`-ს გამოყენების შაბლონი შემდეგია.

C/C++ პროგრამა

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

Pascal პროგრამა

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

სანიმუშო გრადერი

სანიმუშო გრადერი კითხულობს მონაცემებს შემდეგი ფორმატით:

- სტრიქონი 1: ქვეამოცანის ნომერი
- სტრიქონი 2: `n`
- სტრიქონი $3 + i$, ($0 \leq i \leq n - 1$): `stype[i]` (C ტიპისათვის არის 1 და D ტიპისათვის არის 2), `location[i]`.

სანიმუშო გრადერი დაბეჭდავს `Correct`-ს, თუ თქვენი პროგრამის მიერ გამოთვლილი `location[0]` ... `location[n-1]` და `stype[0]` ... `stype[n-1]` შეესაბამება `findLocation`-ის შემავალ მონაცემებს, ან `Incorrect`-ს, თუკი არ შეესაბამება.