D. გამოცნობანას თამაში

ამოცანის სახელი	გამოცნობანას თამაში
დროის ლიმიტი	4 წამი
მეხსიერების ლიმიტი	1 გიგაბაიტი

ლუნდის ძველ უბანში არის ქუჩა, რომელზეც ერთ მწკრივში განლაგებულია N ცალი სახლი, რომლებიც გადანომრილია 0-დან $(N-1)$-მდე. ამ სახლებიდან ერთ-ერთში ცხოვრობს ემა და მის მეგობრებს - ანას და ბერტილს სურთ გაარკვიონ კერძოდ რომელ სახლში იგი. იმის ნაცვლად, რომ უბრალოდ უთხრას მეგობრებს თავისი სახლის ნომერი, ემა გადაწყვეტს ითამაშოს მათთან. თამაშის დაწყებამდე ანამ და ბერტილმა იციან მხოლოდ სახლების რაოდენობა ქუჩაზე. ამ მომენტისათვის ანას და ბერტილს შეუძლიათ აირჩიონ მთელი დადებითი რიცხვი K და მოილაპარაკონ სტრატეგიის შესახებ. ამის შემდეგ ყოველგვარი კომუნიკაცია მათ შორის აკრძალულია.

თავად თამაში შედგება ორი ეტაპისაგან. პირველ ეტაპზე ემა ირჩევს სახლების თანმიმდევრობას იმგვარად, რომ მისი სახლი ინახულონ ყველაზე ბოლოს. შემდეგ მას მიჰყავს ანა ამ სახლებთან არჩეული თანმიმდევრობით ისე, რომ წინასწარ არ ატყობინებს მას ამ თანმიმდევრობის შესახებ. ყოველი სახლისათვის, რომელიც არ წარმოადგენს ემას სახლს, ანას ნება ეძლევა დააწეროს ცარცით რაიმე მთელი დადებითი რიცხვი 1-დან K-მდე შესასვლელ კარზე. უკანასკნელ სახლში, რომელსაც ისინი ესტუმრებიან და რომელიც ემას სახლს წარმოადგენს, თავად ემა დააწერს კარზე მთელ რიცხვს 1-დან K-მდე.

თამაშის მეორე ეტაპზე ბერტილი გაივლის ქუჩას 0 სახლიდან $N-1$ სახლამდე და და კითხულობს ყველა იმ რიცხვს, რომელიც ანამ და ემამ დააწერეს სახლებზე. ახლა მან უნდა გამოიცნოს რომელ სახლში ცხოვრობს ემა. საამისოდ მას აქვს მხოლოდ ორი ცდა და თუ მაქსიმუმ ორ ცდაში გამოიცნობს, ის და ანა გაიმარჯვებენ თამაშში. წინააღმდეგ შემთხვევაში გამარჯვებულად ემა ითვლება. შეგიძლიათ თუ არა, თქვენ შეიმუშავოთ სტრატეგია, რომლის შედეგადაც ანა და ბერტილი გარანტირებულად გაიმარჯვებენ? თქვენი სტრატეგია შეფასდება K-ს მნიშვნელობის მიხედვით (რაც უფრო ნაკლებია, მით უკეთესი).

იმპლემენტაცია

ეს არის ამოცანა მულტი-გაშვებით, ანუ თქვენი თქვენი პროგრამა შესრულდება რამდენჯერმე. პირველი გაშვებისას ის მოახდენს ანას სტრატეგიის რეალიზაციას, ხოლო მეორე გაშვებისას მოახდენს ბერტილის ქმედებათა რეალიზაციას.

შესატანი მონაცემების პირველი სტრიქონი შეიცავს ორ მთელ რიცხვს P და N, სადაც P ყოველთვის 1 ან 2-ია (პირველი ან მეორე ეტაპი), და N არის მასივის ზომა. **სანიმუშო შეტანის გარდა (არ გამოიყენება შეფასებისას), N ყოველთვის ტოლია $100\,000$-ის**.

მომდევნო შესატანი მონაცემები დამოკიდებულია ეტაპზე:

ეტაპი 1

თქვენმა პროგრამამ გამოტანა უნდა დაიწყოს K-ს მნიშვნელობის ($1 \leq K \leq 1\,000\,000$) გამოტანით ერთადერთ სტრიქონში. შემდეგ, $(N-1)$-ჯერ, უნდა წაიკითხოთ სტრიქონი, რომელიც შეიცავს i ($0 \leq i < N$) ინდექსს, და გამოიტანოთ სტრიქონი რიცხვით A_i ($1 \leq A_i \leq K$) - ესაა რიცხვი, რომელსაც ანა დააწერს i ინდექსის მქონე სახლს. ყოველი i ინდექსი, გარდა ემას სახლის საიდუმლო ინდექსისა გამოჩნდება ზუსტად ერთხელ გრადერის მიერ დადგენილი თანმიმდევრობით.

ეტაპი 2

თქვენმა პროგრამამ უნდა წაიკითხოს სტრიქონი N მთელი რიცხვით, $A_0, A_1, ..., A_{N-1}$ და შემდეგ უნდა გამოიტანოს სტრიქონი ორი მთელი რიცხვით s_1 და s_2 ($0 \leq s_i < N$), გამოცნობილი ინდექსები. s_1 და s_2 შეიძლება ტოლი იყოს.

იმპლემენტაციის დეტალები

გაითვალისწინეთ, რომ პროგრამის მე-2 ეტაპზე გაშვებისას, პროგრამა გადაიტვირთება. ეს ნიშნავს, რომ თქვენ ვერ შეინახავთ ინფორმაციას ზოგიერთ ცვლადში გაშვებებს შორის.

ყოველი დაბეჭდილი სტრიქონის შემდეგ გაასუფთავეთ სტანდარტული გამოტანა, წინააღმდეგ შემთხვევაში შეიძლება ჩაითვალოს, რომ თქვენმა პროგრამამ გადააჭარბა დროის ლიმიტს.პითონში `print()` ავტომატურად ასუფთავებს. C++-ში `cout << endl;` ასუფთავებს და ბეჭდვა გადააქვს ახალ სტრიქონზე printf-ის შემთხვევაში გამოიყენეთ `fflush(stdout)`.

გრადერი ამ ამოცანაში შეიძლება იყოს ადაპტიური, ანუ მან შიძლება შეიცვალოს ქცევა თქვენი პროგრამის გამოტანის მიხედვით, რათა თავიდან იქნას აცილებული ევრისტიკული ამოხსნები გრადერმა შეუძლია აამოქმედოს პირველი ეტაპი, ნახოს თქვენი გამოტანილი მონაცემები და ხელახლა გაუშვას პირველი ეტაპი იმ ინფორმაციის გათვალისწინებით, რომელიც მან პირველი გაშვებისას მიიღო.

თქვენი პროგრამა *უნდა იყოს* დეტერმინირებული, ანუ უნდა მოიქცეს ერთნაირად, თუკი მას ორჯერ გავუშვებთ ერთსა და იმავე შემავალ მონაცემებზე. თუ თქვენ იყენებთ შემთხვევითი რიცხვების ფუნქციას თქვენს პროგრამაში, საწყისად გამოიყენეთ ფიქსირებული შემთხვევითი რიცხვი. ეს შეიძლება განახორციელოთ `srand` (C++) ან `random.seed` (Python) ფუნქციებისთვის მკაცრად განსაზღვრული მუდმივის გადაცემით (კერძოდ, თქვენ არ შეგიძლიათ გამოიყენოთ `srand`

`(time(NULL))` C++-ში). თუ გრადერი აღმოაჩენს, რომ თქვენი პროგრამა არაა დეტერმინირებული, იგი მიიღებს ვერდიქტს Wrong Answer.

თუ ცალკეულ შესრულებათა *ჯამი* (3-მდე) აჭარბებს დროის ლიმიტს, მაშინ თქვენი პროგრამა მიიღებს ვერდიქტს Time Limit Exceeded.

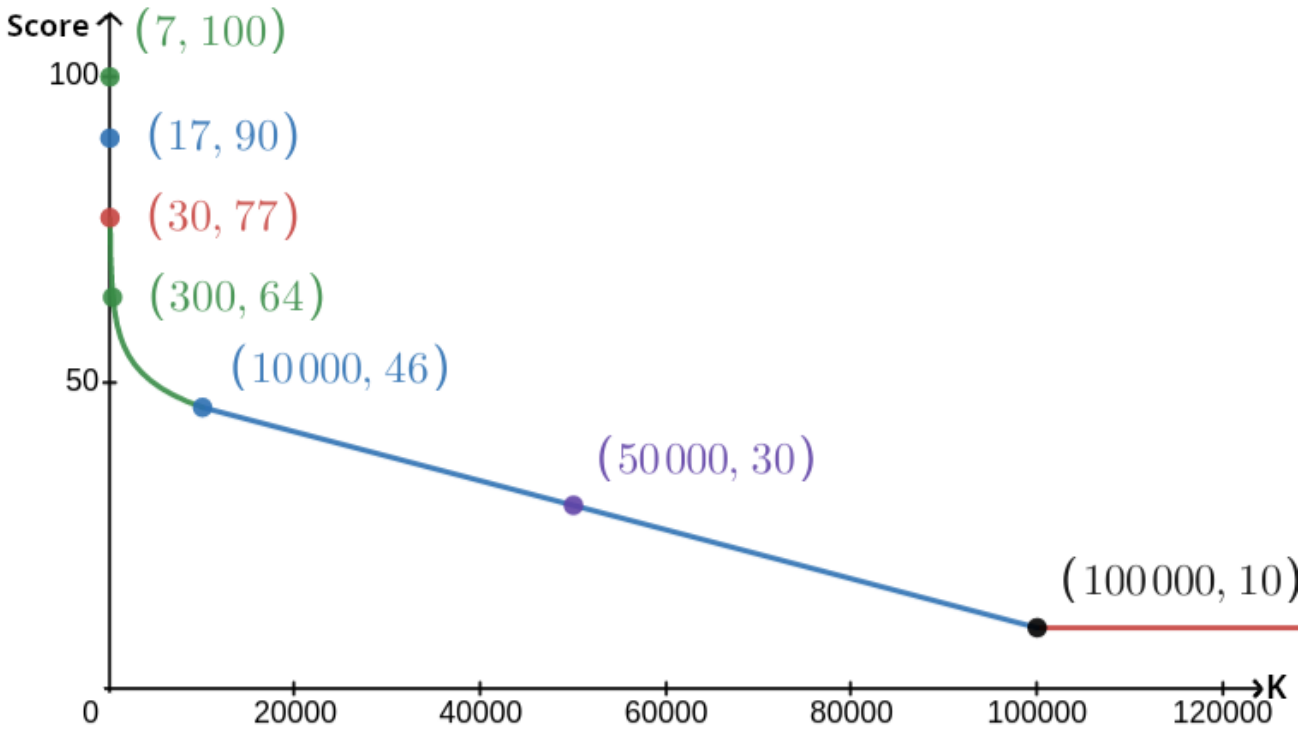
შეფასება

თქვენი პროგრამა გაიტესტება რამედენიმე ტესტური შემთხვევისთვის. თუ თქვენი ამოხსნა ვერ გაატარებს *რომელიმე* ტესტურ შემთხვევასIf your solution fails on *any* of these test cases (მაგალითად, არასწორი პასუხის wrong answers (Wrong Answer), დაქრაშვის (Run-Time Error), დროის ლიმიტის გადაჭარბების (Time Limit Exceeded), და ა.შ.), ამოხსნა მიიღებს 0 ქულას და შესაბამის ვერდიქტს.

თუ თქვენი პროგრამა წარმატებით იპოვის ემას სახლის ინდექსს *ყველა* ტესტურ შემთხვევაში, თქვენ მიიღებთ ვერდიქტს Accepted, და ქულას ქვემოთ აღწერილი პრინციპით. განვსაზღვროთ K-ს მნიშვნელობა, როგორც მაქსიმალური ყველა სატესტო შემთხვევაში, მაშინ K-სთან დამოკიდებულებაში:

	ქულა
$K_{max} > 99\,998$	10 ქულა
$10\,000 < K_{max} \leq 99\,998$	$10 + \lfloor 40(1 - K_{max}/10^5) \rfloor$ ქულა
$30 < K_{max} \leq 10\,000$	$46 + \lfloor 31(4 - \log_{10}(K_{max}))/(4 - \log_{10}(30)) \rfloor$ ქულა
$7 < K_{max} \leq 30$	$107 - K_{max}$ ქულა
$K_{max} \leq 7$	100 ქულა

შეფასების ფუნქცია ნაჩვენებია ქვემოთ მოცემულ ნახაზზე.



სანიმუშო ტესტის შემთხვევა იგნორირებულია ქულების მისაღებად და თქვენმა ამოხსნამ არ უნდა იმუშაოს მასზე.

ტესტირების ინსტრუმენტი

თქვენი ამოხსნის ტესტირების გასაადვილებლად, ჩვენ გთავაზობთ მარტივ ხელსაწყოს, რომელიც შეგიძლიათ ჩამოტვირთოთ. იხილეთ „დანართები“ (“attachments”) Kattis-ის პრობლემის გვერდის ბოლოში. ინსტრუმენტი არჩევითია გამოსაყენებლად და თქვენ უფლება გაქვთ შეცვალოთ იგი. გაითვალისწინეთ, რომ ოფიციალური გრადერის პროგრამა Kattis-ზე განსხვავდება ტესტირების ხელსაწყოებისგან.

გამოყენების მაგალითი ($N = 4$, $s = 2$, სადაც s არის ბოლო სახლზე დაწერილი რიცხვი):

პითონის პროგრამისათვის, ნახეთ `solution.py`:

```
python3 testing_tool.py pypy3 solution.py <<<"4 2"
```

C++-ზე დაწერილი პროგრამისთვის პირველი კომპილაცია არის (მაგალითად, `g++ -g -O2 -std=gnu++17 -static solution.cpp -o solution.out`) და შემდეგ გაუშვით:

```
python3 testing_tool.py ./solution.out <<<"4 2"
```

ტესტირების ინსტრუმენტი ეწვევა სახლებს შემთხვევითი თანმიმდევრობით. სპეციფიური თანმიმდევრობის მისაღებად, შეცვალეთ ტესტირების ინსტრუმენტი, სადაც ნათქვამია “MODIFY HERE”.

ინტერაქციის მაგალითი

სანიმუშო ტესტის შემთხვევა იგნორირებულია ქულების მისაღებად და თქვენმა ამოხსნამ არ უნდა იმუშაოს მასზე.

დავუშვათ, გვაქვს $N = 4$ და რომ ემა ცხოვრობს 1 სახლში. დაე, A იყოს სახლებზე დაწერილი რიცხვების სია. თავდაპირველად, $A = [0, 0, 0, 0]$, სადაც 0 ნიშნავს, რომ შესაბამის სახლზე რიცხვი არ არის დაწერილი.

თქვენი კოდის პირველ გაშვებაში:

მოცემულია, რომ $N = 4$. თქვენი ამოხსნა პასუხობს $K = 3$.

მოთხოვნილია A_2. თქვენი ამოხსნა პასუხობს 3-ით. A ახლა არის $[0, 0, 3, 0]$.

მოთხოვნილია A_0. თქვენი ამოხსნა პასუხობს 1-ით. A ახლა არის $[1, 0, 3, 0]$.

მოთხოვნილია A_3. თქვენი ამოხსნა პასუხობს 2-ით. A ახლა არის $[1, 0, 3, 2]$.

დასასრულ გრადერი განსაზღვრავს $A_1 = 2$, მასივის საბოლოო სახეა A = $[1, 2, 3, 2]$. ეს ნიშნავს პირველი ეტაპის დასრულებას.

თქვენი კოდის მეორე გაშვებისას თქვენი ამოხსნას გადაეცემა სია `1 2 3 2`. ის პასუხობს `1 3`.

ვინაიდან ერთ-ერთი გამოცნობა არის (1) სახლის სწორი მაჩვენებელი, ანა და ბერტილი იგებენ თამაშს.

grader output	your output
1 4	
	3
2	
	3
0	
	1
3	
	2

grader output	your output
2 4	
1 2 3 2	
	1 3