დაყოფა

თემური და მისი ასისტენტი, ულუგბექი, ილუზიონისტურ ნომერს ამზადებენ *Uzbekistan Got Talent* შოუსთვის. ნომერი ეფუძნება თემურის მიერ შემდეგი **დაყოფის ამოცანის** გადაჭრას: დაიყოს მთელი დადებითი რიცხვების მოცემული ნაკრები K რაოდენობის ისეთ არაცარიელ ჯგუფებად, რომ თითოეულ ჯგუფში შემავალი რიცხვების ჯამი ერთნაირი იყოს. სხვა სიტყვებით რომ ვთქვათ, მოცემული ნაკრების თითოეული მთელი რიცხვი ამ K რაოდენობის ჯგუფებიდან ზუსტად ერთ მათგანს უნდა ეკუთვნოდეს და ყოველ ჯგუფში შემავალი რიცხვების ჯამი ტოლი უნდა იყოს. მაგალითად, თუ მოცემული ნაკრებია $[2, 1, 6, 4, 5]$ და $K = 3$, ვალიდური დაყოფა იქნება $[2, 4]$, $[1, 5]$ და $[6]$. ამ შემთხვევაში, თითოეულ ჯგუფში ელემენტების ჯამი ზუსტად 6-ის ტოლია.

ნომერი შემდეგნაირად სრულდება:

- ულუგბექი და თემური შედიან ცალ-ცალკე კაბინებში და ერთმანეთთან კომუნიკაცია არ შეუძლიათ.
- ჟიურის წევრი ულუგბექს აძლევს A მასივს, რომელიც შედგება N რაოდენობის მთელი დადებითი $A[0], A[1], \ldots, A[N-1]$ რიცხვისაგან, სადაც თითოეული რიცხვის მნიშვნელობა არის 1-დან და M-მდე შუალედში (საზღვრების ჩათვლით). ჟიურის წევრი ულუგბექს ასევე აძლევს K-ის მნიშვნელობასაც.
- ულუგბექი ირჩევს არაუმეტეს $K - 1$ რაოდენობის (არა აუცილებლად განსხვავებულ) მთელ რიცხვს, რათა **დაამატოს ისინი, როგორც ახალი ელემენტები** A მასივში. დამატებული თითოეული მთელი რიცხვი ასევე უნდა იყოს 1-დან და M-მდე შუალედიდან (საზღვრების ჩათვლით).
- ჟიური ულუგბექის მიერ არჩეულ მთელ რიცხვებს თავდაპირველ მასივში ამატებს, ამ გაფართოებულ მასივს **არაკლებადობით ალაგებს** და თემურს აძლევს K-ს მნიშვნელობასთან ერთად.
- შემდეგ თემურმა უნდა ამოხსნას დაყოფის ამოცანა ამ გაფართოებული, დალაგებული მასივისთვის.

თქვენი ამოცანაა, შეიმუშაოთ და იმპლემენტაცია გაუკეთოთ სტრატეგიას თემურისა და ულუგბექისათვის. მტკიცდება, რომ მოცემული შეზღუდვების პირობებში არსებობს სტრატეგია, რომელიც მათ საშუალებას აძლევს წარმატებით ამოხსნან დაყოფის ამოცანა მიუხედავად იმისა, თუ რა მასივს მიიღებენ ისინი ჟიურისაგან.

იმპლემენტაციის დეტალები

თქვენ იმპლემენტაცია უნდა გაუკეთოთ ორო პროცედურას.

პროცედურა, რომლის იმპლემენტაციაც **ულუგბექისათვის** უნდა მოახდინოთ არის:

```
std::vector<int> add_numbers(std::vector<int> A, int K, int M)
```

- A: N სიგრძის მქონე მასივი, რომელიც წარმოადგენს ულუგბექისთვის მიცემულ თავდაპირველ მასივს.
- K: ჯგუფების სამიზნე რაოდენობა. გაითვალისწინეთ, რომ ულუგბექს მასივში შეუძლია დაამატოს არაუმეტეს $K-1$ რაოდენობის მთელი რიცხვი.
- M: თავდაპირველ მასივში შემავალი თუ ახლად დამატებული მთელი რიცხვების მაქსიმალური შესაძლო მნიშვნელობა.
- ეს პროცედურა თითოეული ტესტისათვის ზუსტად ერთხელ გამოიძახება.

პროცედურამ უნდა დააბრუნოს იმ მთელი რიცხვების C მასივი, რომელთა დამატებაც ულუგბეგს სურს თავდაპირველ მასივში. ავღნიშნოთ S-ით C მასივის სიგრძე.

- S არ უნდა აღემატებოდეს $(K-1)$-ს.
- C მასივის თითოეული ელემენტი უნდა იყოს 1-სა და M-ს შორის (მათი ჩათვლით).

პროცედურა, რომლის იმპლემენტაციაც **თემურისათვის** უნდა მოახდინოთ არის:

```
std::vector<int> find_partition(std::vector<int> B, int K)
```

- B: $N+S$ სიგრძის მასივი, რომელიც შეიცავს როგორც თავდაპირველი A მასივის ელემენტებს, ასევე ულუგბეკის მიერ დამატებულ რიცხვებს. B მასივის ელემენტები არაკლებადობითაა დალაგებული.
- K: ჯგუფების სამიზნე რაოდენობა.
- ეს პროცედურა თითოეული ტესტისათვის ზუსტად ერთხელ გამოიძახება.

პროცედურამ უნდა დააბრუნოს P მასივი, რომელიც წარმოადგენს B მასივის K რაოდენობის არათანამკვეთ ჯგუფებად დაყოფას.

- P-ს სიგრძე უნდა იყოს $N+S$.
- თითოეული j-ისათვის, სადაც $0 \le j < N+S$, $P[j]$ აღნიშნავს ჯგუფს, რომელსაც $B[j]$ მიეკუთვნება.
- ჯგუფები გადანომრილი უნდა იყოს 0-დან $(K-1)$-მდე, რაც ნიშნავს, რომ თითოეული $0 \le j < N+S$-ისათვის უნდა სრულდებოდეს პირობა: $0 \le P[j] < K$.
- თითოეული i-ისათვის, რომელიც მოთავსებულია 0-სა და $(K-1)$-ს შორის (მათი ჩათვლით), უნდა არსებობდეს სულ მცირე ერთი j ელემენტი მაინც ($0 \le j < N+S$), რომლისთვისაც $P[j]=i$.
- K რაოდენობის ჯგუფებიდან თითოეულში შემავალი ელემენტების ჯამი ერთმანეთის ტოლი უნდა იყოს.

რეალური შეფასებისას პროგრამა, რომელიც ზემოთ აღნიშნულ პროცედურებს იძახებს, სრულდება ზუსტად ორჯერ.

- პროგრამის პირველი გაშვებისას:
  - `add_numbers` გამოიძახება ზუსტად ერთხელ.

    - დაბრუნებული მასივი მუშავდება შეფასების სისტემის მიერ B მასივის გამოსათვლელად.
- პროგრამის მეორე გაშვებისას:
    - `find_partition` გამოიძახება ზუსტად ერთხელ.

შეზღუდვები

- $3 \leq N \leq 100\,000$
- $2 \leq K \leq 100\,000$
- $K \leq N$
- $1 \leq M \leq 10^9$
- $1 \leq A[i] \leq M$ თითოეული i-სათვის, სადაც $0 \leq i < N$.

ქვეამოცანები

ქვეამოცანა	ქულები	დამატებითი შეზღუდვები
1	5	$N = 3$
2	4	$M = 1$
3	7	$M \leq 2$
4	10	$N \leq 10$
5	7	$K = 2$
6	12	$K \leq 3$
7	17	$K \leq 10$
8	20	$K \leq 100$
9	18	დამატებითი შეზღუდვების გარეშე.

მაგალითი

განვიხილოთ შემდეგი გამოძახება:

```
add_numbers([8, 2, 9, 6, 1, 5, 5], 3, 9)
```

ამ მაგალითში $A = [8, 2, 9, 6, 1, 5, 5]$. ჩვენ გვსურს რიცხვები დავყოთ $K = 3$ ჯგუფად. ულუგბექს შეუძლია დაამატოს რიცხვები 1-სა და $M = 9$-ს შორის.

პროცედურამ შეიძლება დააბრუნოს $[5, 4]$, რაც ნიშნავს, რომ ულუგბექმა გადაწყვიტა დაამატოს $S = 2$ ახალი მთელი რიცხვი: 5 და 4. ისინი ვალიდურია, რადგან ორივე ელემენტი მდებარეობს 1 -სა და $M = 9$-ს შორის.

შემდეგ გაფართოებული მასივი ლაგდება და გადაეცემა თემურს შემდეგი გამოძახებით:

```
find_partition([1, 2, 4, 5, 5, 5, 6, 8, 9], 3)
```

ამ მასივში არსებული მთელი რიცხვები შეგვიძლია დავყოთ სამ ჯგუფად: $[1, 5, 9]$, $[2, 5, 8]$ და $[4, 5, 6]$. შევნიშნოთ, რომ თითოეულ ამ ჯგუფში ელემენტთა ჯამი 15-ის ტოლია. ამ დაყოფის დასაბრუნებლად, პროცედურამ უნდა დააბრუნოს მასივი $[0, 1, 2, 0, 1, 2, 2, 1, 0]$.

სანიმუშო გრადერი

შეტანის ფორმატი:

```
N K M
A[0] A[1] ... A[N-1]
```

გამოტანის ფორმატი:

როცა `add_numbers`-ის გამოძახება დასრულდება, სანიმუშო გრადერი:

- გამოითვლის დალაგებულ B მასივს.
- ბეჭდავს C და B მასივებს შემდეგ ფორმატში, **რომელსაც მოჰყვება ცარიელი სტრიქონი**:

```
S
C[0] C[1] ... C[S-1]
B[0] B[1] ... B[N+S-1]
```

როცა `find_partition`-ის გამოძახება დასრულდება, სანიმუშო გრადერს გამოაქვს:

```
L
P[0] P[1] ... P[L-1]
```

აქ, L არის P მასივის სიგრძე, რომელიც `find_partition`-მა დააბრუნა.