

Tapşırıq 1. Ağır qəpiklər

Aran krallığında əsrlər boyu ərazisini işğallardan qoruyan möhtəşəm bir divar var idi. Ancaq düz bir il əvvəl bir quldur dəstəsi divarın çox mürəkkəb olduğuna qərar verdi və onu yandıraraq yerə yıxdılar. İndi şahzadə Mariya divarı orijinal gözəlliyinə qaytarmaq üçün bacarıqlı inşaatçılar işə götürməyi planlaşdırır. İnşaatçılara ödəniş etmək üçün o, kral xəzinəsinə getdi və hər birinin 0-dan $(N - 1)$ -ə unikal nömrəsi olan $N = 2^K$ sayda sikkə tapdı. Krallığın mühasibi ona bildirdi ki, iki növ sikkə var - yüngül və ağır, bütün ağır sikkələr eyni dərəcədə ağırdır, eləcə də bütün yüngül sikkələr. Həmçinin, mühasibin dediyinə görə, xəzinədə ən azı bir ağır və ən azı bir yüngül sikkə var.

Yeni divarı inşa edənlərlə razılaşdırılmış şərtlərə görə, şahzadə Mariya onlara yalnız ağır sikkələrlə ödəməlidir və işə başlamazdan əvvəl bir ağır sikkə avans borcu var. Problem ondadır ki, mühasib sikkələrin neçəsinin və hansıların ağır olduğunu bilmir. Xoşbəxtlikdən, onun eyni sayda sikkədən ibarət iki dəstin birləşmiş çəkisini müqayisə edə biləcək tərəziləri var. Mühasib indi kral xəzinəsindəki ağır sikkələrin sayının ödəniş üçün kifayət olacağına əmin olmalı və avansın ödənilə biləcəyi bir sikkə tapmalıdır. Ancaq bu, onun çox vaxtını aparacaq və şahzadə Mariya yeni divarın tikintisinə mümkün qədər tez başlanmasını istəyir, ona görə də kömək üçün sizə müraciət edir.

Tapşırıq

`count_heavy` funksiyasını icra edən `coins` proqramı yazın. Bu proqram münisflər heyətinin proqramı ilə birlikdə kompayl olunacaq və onunla əlaqə yaradaraq kral xəzinəsindəki ağır sikkələrin sayını, eyni zamanda belə sikkələrdən birinin nömrəsini tapmalıdır.

İmplementasiya detalları

Funksiyanız aşağıdakı formada olmalıdır:

`pair<int, int> count_heavy(int k)`

O, bir parametr – K ədədini (xatırladaq ki, sikkələrin sayı 2^K -dir) alacaq və iki tam ədəddən (ağır sikkələrin sayı və belə sikkələrdən birinin nömrəsi) ibarət cüt qaytarmalıdır.

Münisflər heyətinin proqramı ilə əlaqə yaratmaq üçün sizə aşağıdakı funksiya təqdim olunur:

`int weigh(const vector<int> &a, const vector<int> &b)`

Bu funksiya parametr olaraq çəkmək üçün sikkələrin nömrələrini ehtiva edən iki vektor götürür. Vektorlar bərabər sayda elementə malik olmalı və təkrarlanan ədədlərdən ibarət olmamalıdır. Funksiya aşağıdakı qiymətlərdən birini qaytaracaq:

- 0, iki sikkə dəstəsinin ümumi çəkisi bərabərdirsə;
- -1 , nömrələri a vektorunda olan sikkələrin ümumi çəkisi nömrələri b -də olan sikkələrin ümumi çəkisindən az olarsa;
- $+1$, nömrələri a vektorunda olan sikkələrin ümumi çəkisi nömrələri b -də olan sikkələrin ümumi çəkisindən çox olarsa.

Qeyd edək ki, `weigh` funksiyasının mürəkkəbliyi parametr kimi qəbul edilən vektorlarda olan elementlərin toplam sayı ilə mütənasibdir.

Siz `count_heavy` funksiyasını ehtiva edən `coins.cpp` faylını sistemə göndərməlisiniz. O, işiniz üçün lazım olan digər kod və funksiyaları ehtiva edə bilər, lakin `main` funksiyasını ehtiva etməməlidir. Həmçinin, standart

girişdən oxumamalı və ya standart çıxışa çap etməməlisiniz. Proqramınız həmçinin `coins.h` başlıq faylını daxil etməlidir:

```
#include "coins.h"
```

Məhdudiyyətlər

$$1 \leq K \leq 20$$

Qiymətləndirmə

Əgər həlliniz icra zamanı *weigh* funksiyasına 10 000-dən çox olmayan sayda çağırış etməklə testə düzgün cavab verirsə, həmin test üçün $coef = \min\left(1, \frac{11}{9} * \left(\frac{author+1}{yours+1}\right)\right)$ düsturuna əsasən əmsal hesablanacaq. Burada *author* nəzərdə tutulan müəllif həllərindən biri tərəfindən *weigh* funksiyasına edilən çağırışların **maksimum** sayı (aşağıdakı alt tapşırıqlara baxın), *yours* isə sizin həlliniz tərəfindən *weigh* funksiyasına edilən çağırışların sayıdır.

Əks halda həmin test üçün $coef = 0$ götürüləcək.

Hər bir alt tapşırıq müəyyən sayda qrupdan ibarətdir, hər qrupda tam olaraq üç test var. Qrupların hər biri **müstəqil olaraq** qiymətləndirilir və onun üçün alacağınız bal, onun üçün nəzərdə tutulan bal ilə bu qrupda olan testlər arasında *coef* əmsalının minimum qiymətinin hasilinə bərabərdir.

Alt tapşırıqlar

Alt tapşırıq	Bal	<i>author</i>	Digər məhdudiyyətlər
1	5	≤ 40	$K \leq 5$
2	10	≤ 250	Yalnız bir ağır və ya bir yüngül sikkə var.
3	20	≤ 1000	Ağır sikkələrin sayı K -dan çox deyil və ya yüngül sikkələrin sayı K -dan çox deyil.
4	25	≤ 100	$K \leq 10$
5	40	≤ 500	Əlavə məhdudiyyətlər yoxdur.

Komunikasiya nümunəsi

Yarışmaçının funksiyası	Münsiflər heyətinin proqramı
	<code>count_heavy(2)</code> çağırır
<code>weigh({0, 1}, {2, 3})</code> çağırır	-1 qaytarır
<code>weigh({1, 2}, {0, 3})</code> çağırır	+1 qaytarır
<code>weigh({1}, {2})</code> çağırır	0 qaytarır
<code>{3, 2}</code> qaytarır	

Yerli sınaq

Yerli sınaq üçün sizə `coins.h` və `Lgrader.cpp` faylları verilir. `coins.cpp` faylınızı və təqdim olunan iki faylı eyni qovluğa yerləşdirin. Sonra üç faylı birlikdə kompilyat edin. Beləliklə, funksiyanızın düzgünlüyünü yoxlamaq üçün bir proqram əldə edəcəksiniz. Proqram aşağıdakı verilənlər ardıcılığını tələb edəcək:

- birinci sətirdə: K dəyərini bildiren bir müsbət tam ədəd.
- ikinci sətirdə: Müvafiq sikkənin ağır və ya yüngül olmasından asılı olaraq 1 və ya 0 dəyərləri olan 2^K sayda ədəd.

Proqram *weigh* funksiyasının çağırışları haqqında məlumat, *count_heavy* funksiyanızın qaytardığı cavab və cavabın düzgün olub-olmadığını göstərən bir mesaj verəcəkdir.