Więzienie

⏳ 2 sek. 💾 1024 MB

Alicja i Bajtazar zostali niesłusznie skazani do odsiadki w więzieniu o podwyższonym rygorze. Teraz muszą zaplanować ucieczkę. Aby to zrobić, muszą być w stanie komunikować się tak efektywnie, jak tylko to możliwe (w szczególności, Alicja musi codziennie wysyłać informacje do Bajtazara). Niestety, nie mogą się spotykać. Mogą jedynie wymieniać informacje napisane na chusteczkach. Każdego dnia Alicja chce wysłać nową porcję informacji do Bajtazara – liczbę od 0 do $N-1$. Podczas każdego obiadu Alicja zabiera trzy chusteczki i zapisuje liczbę od 0 do $M-1$ na każdej z nich (liczby mogą się powtarzać) i zostawia chusteczkę na swoim krześle. Następnie ich wróg, Chociemir, niszczy jedną z chusteczek i miesza pozostałe dwie. Na koniec, Bajtazar odnajduje pozostałe dwie chusteczki i odczytuje liczby zapisane na nich. Musi dokładnie odszyfrować oryginalną liczbę, którą Alicja chciała mu przekazać. Ponieważ na chusteczkach jest ograniczone miejsce, liczba M jest stała. Jednak Alicja i Bajtazar starają się zmaksymalizować ilość informacji, które mogą przekazać, zatem mogą wybrać N tak duże, jak chcą. Pomóż Alicji i Bajtazarowi, zaimplementuj strategię dla każdego z nich, która próbuje zmaksymalizować wartość N.

Szczegóły implementacji

Ponieważ jest to zadanie komunikacyjne, Twój program zostanie uruchomiony dwukrotnie (raz dla Alicji, raz dla Bajtazara) i nie może współdzielić żadnych danych i komunikować się w żaden inny sposób, niż podany. Musisz zaimplementować trzy funkcje:

```
int setup(int M);
```

Funkcja zostanie wywołana raz na początku wywołania Twojego programu przez Alicję i raz na początku wywołania Twojego programu przez Bajtazara. Jako argument dostaje M i musi zwrócić wybraną wartość N. Oba wywołania `setup` musza zwrócić to samo N.

```
std::vector<int> encode(int A);
```

Funkcja powinna implementować strategię Alicji. Zostanie wywołana z argumentem do zakodowania A ($0 \leq A < N$) i musi zwrócić trzy liczby W_1, W_2, W_3 ($0 \leq W_i < M$) kodujące A. Ta funkcja zostanie wywołana T razy – jeden raz na każdy dzień (wartości A mogą się powtarzać).

```
int decode(int X, int Y);
```

Funkcja powinna implementować strategię Bajrazara. Zostanie wywołana z argumentami będącymi dwoma z trzech liczb zwróconych przez `encode` w pewnej kolejności. Musi zwrócić tą samą wartość A, z którą została wywołana funkcja `encode`. Ta funkcja również

zostanie wywołana T razy – odpowiadając T wywołaniom funkcji `encode`. Wywołania będą w tej samej kolejności. Wszystkie wywołania `encode` będą przed wszystkimi wywołaniami `decode`.

Ograniczenia

- $M \leq 4300$
- $T = 5000$

Ocenianie

Dla każdego podzadania, ułamek S punktów które otrzymasz, zależy od najmniejszego N zwróconego przez `setup` ze wszystkich testów tego podzadania. Twój wynik jest również zależny od N^*, które jest docelowym N, które musisz uzyskać, żeby otrzymać maksymalną liczbę punktów za podzadanie:

- Jeśli twoje rozwiązanie będzie niepoprawne na dowolnym teście, wtedy $S = 0$.
- Jeśli $N \geq N^*$, wtedy $S = 1.0$.
- Jeśli $N < N^*$, wtedy $S = \max\left(0.35 \max\left(\frac{\log(N) - 0.985\log(M)}{\log(N^*) - 0.985\log(M)}, 0.0\right)^{0.3} + 0.65\left(\frac{N}{N^*}\right)^{2.4}, 0.01\right)$.

Podzadania

Podzadanie	Punkty	M	N^*
1	10	700	82017
2	10	1100	202217
3	10	1500	375751
4	10	1900	602617
5	10	2300	882817
6	10	2700	1216351
7	10	3100	1603217
8	10	3500	2043417
9	10	3900	2536951
10	10	4300	3083817

Przykład

Rozważmy następujący przykład z $T = 5$. Użyte zostało przykładowe kodowanie, gdzie Alicja wysyła trzy równe liczby żeby zakodować 0, lub trzy różne liczby żeby zakodować 1.

Zwróć uwagę, że Bajtazar może odkodować oryginalną liczbę, z dowolnych dwóch liczb wśród każdych trzech liczb wysłanych przez Alicję.

Uruchomienie	Wywołana funkcja	Zwrócona wartość
Alicja	`setup(10)`	`2`
Bajtazar	`setup(10)`	`2`
Alicja	`encode(0)`	`{5, 5, 5}`
Alicja	`encode(1)`	`{8, 3, 7}`
Alicja	`encode(1)`	`{0, 3, 1}`
Alicja	`encode(0)`	`{7, 7, 7}`
Alicja	`encode(1)`	`{6, 2, 0}`
Bajtazar	`decode(5, 5)`	`0`
Bajtazar	`decode(8, 7)`	`1`
Bajtazar	`decode(3, 0)`	`1`
Bajtazar	`decode(7, 7)`	`0`
Bajtazar	`decode(2, 0)`	`1`

Przykładowa biblioteczka

W przykładowej biblioteczce, wszystkie wywołania `encode` i `decode` będą w tym samym uruchomieniu twojego programu. Dodatkowo, `setup` będzie wywołane tylko raz (a nie dwa razy, raz na uruchomienie, tak jak w systemie sprawdzającym).

Wejście to jedna liczba całkowita – M. Następnie zostanie wypisane N, które twój `setup` zwrócił. Następnie zostaną wywołane funkcje `encode` i `decode` w tej kolejności T razy z losowo wygenerowanymi liczbami od 0 do $N-1$ i losowymi wyborami, które dwie liczby z trzech zwróconych przez `encode` zostaną przekazane do `decode` (i w jakiej kolejności). Jeśli twoje rozwiązanie nie zadziała zostanie wypisany komunikat błędu.