# Akademia Górniczo-Hutnicza
# im. Stanisława Staszica w Krakowie
# Ogólnopolska Olimpiada
# "O Diamentowy Indeks AGH" 2021/22

## Informatyka – Etap III

Maksymalna ocena za każde zadanie to 20 punktów.

# Zadanie 1. Dice

Mamy do dyspozycji $N$ kostek. Każda z kostek ma od 1 do 9 ścian. Kostka o $m$ ścianach posiada dokładnie jedną ścianę z $k$ oczkami dla każdej liczby całkowitej $1 \leq k \leq m$. Przy rzucie dowolną kostką, wypadnięcie każdej liczby oczek spośród znajdujących się na niej jest jednakowo prawdopodobne.

Wykonano $T$ rzutów wszystkimi kostkami równocześnie. Dla każdego z rzutów zapisano liczby oczek, jakie wypadły na każdej z kostek (w dowolnej kolejności). Wyniki $i$-tego rzutu, gdzie $0 \leq i \leq T-1$, zawarte są w stringu `t[i]`. Długość tego stringu wynosi $N$ (jeden znak dla każdej kostki). Przykładowo, jeżeli `t[0] =` "423" to oznacza, że mamy 3 kostki i w rzucie numer 0 zaobserwowano na jednej z nich 4 oczka, na drugiej 2 oczka a na trzeciej 3.

**Uwaga:** Kolejność kostek może być różna w kolejnych rzutach.

Napisz program, który na podstawie wyżej opisanej $T$-elementowej tablicy stringów, wyznaczy możliwie najmniejszą sumaryczną liczbę ścian na wszystkich kostkach.

## Wejście

W pierwszym wierszu standardowego wejścia znajduje się jedna liczba całkowita $1 \leq T \leq 50$ – liczba rzutów kostkami. Każdy z kolejnych $T$ wierszy zawiera string złożony wyłacznie z cyfr '1'–'9' opisujący wyniki kolejnego rzutu. Długość wszystkich stringów jest taka sama i równa liczbie kostek $1 \leq N \leq 50$.

## Wyjście

W pierwszym i jedynym wierszu program powinien wypisać liczbę całkowitą – najmniejszą sumaryczną liczbę ścian na wszystkich $N$ kostkach.

## Przykład

1. Dla danych wejściowych:

   ```
   4
   137
   364
   115
   724
   ```

   poprawną odpowiedzią jest:

   ```
   14
   ```

   Mamy 3 kostki. W pierwszym rzucie wypadło odpowiednio 1, 3 i 7 oczek, w drugim 3, 6 i 4 oczka, w trzecim 1, 1 i 5 a w czwartym 7, 2 i 4. W tym przypadku możliwe było użycie kostek o 3, 4 i 7 ścianach, dając sumę 14 ścian. Mniejsza liczba ścian nie jest możliwa.

2. Dla danych wejściowych:

   ```
   4
   1112
   1111
   1211
   1111
   ```

   poprawną odpowiedzią jest:

   ```
   5
   ```

   W tym przypadku można było użyć trzech kostek z jedną ścianą i jednej kostki o dwóch ścianach.

# Zadanie 2. Encryption

Dany jest binarny string $S$ (string źródłowy, składający się wyłącznie ze znaków '0' i '1') oraz binarny string $K$ (klucz). Kodowanie stringu $S$ polega na wykonaniu operacji XOR ($\wedge$) na kolejnych bitach stringów $S$ i $K$. W wyniku otrzymujemy string zakodowany $D$. Wszystkie trzy stringi mają tę samą długość $L$. Formalnie $D_i = S_i \wedge K_i$, $i = 0, \ldots, L-1$.

Znamy $N$ stringów źródłowych. $M$ z nich zostało zakodowanych nieznanym (ale tym samym) kluczem. Napisz program, który wyznacza liczbę różnych kluczy, które mogły zostać użyte do zakodowania stringów. Formalnie, klucz $K$ mógł zostać użyty, jeżeli dla każdego z zakodowanych stringów, $D$, istnieje string źródłowy, $S$, taki, że $D$ został uzyskany przez zastosowanie klucza $K$ do stringu $S$.

## Wejście

W pierwszym wierszu standardowego wejścia znajdują się dwie liczby całkowite $1 \leq N \leq 50$ i $1 \leq M \leq N$. W kolejnych dwóch liniach znajduje się odpowienio $N$ i $M$ stringów binarnych, wszystkie o jednakowej długości $1 \leq L \leq 50$. Są one odpowiednio stringami źródłowymi i zakodowanymi.

## Wyjście

W pierwszym i jedynym wierszu program powinien wypisać jedną liczbę całkowitą – liczbę różnych kluczy, jakie mogły zostać użyte do zakodowania stringów źródłowych. Można założyć, że zawsze istnieje przynajmniej jeden taki klucz.

## Przykłady

1. Dla danych wejściowych:

   ```
   2 2
   110 001
   101 010
   ```

   poprawną odpowiedzią jest:

   ```
   2
   ```

   W tym przypadku możliwymi kluczami są: “011“ i “100“.

2. Dla danych wejściowych:

   ```
   2 1
   01 10
   00
   ```

   poprawną odpowiedzią jest:

   ```
   2
   ```

# Zadanie 3. The last one wins

$N$ kolegów postanowiło zagrać w grę. Każdy z nich otrzymał kolejny numer od 1 do $N$ i ustawili się zgodnie z otrzymaną numeracją w kółko (numery rosną zgodnie z kierunkiem wskazówek zegara).

Gra składa się z etapów. Etapy ponumerowane są poczynając od 1. W każdym etapie odliczanie zaczyna jeden z uczestników (jak opisano poniżej), który mówi "jeden". Kolejna osoba w kółku (w kierunku zgodnym z ruchem wskazówek zegara), mówi "dwa", i tak dalej, aż wypowiedziana liczba osiągnie limit wartości dla tego etapu. Limitem wartości dla etapu $T$ jest $T^3$, czyli limit wynosi 1 dla pierwszego etapu, 8 dla drugiego, 27 dla trzeciego, itd.

Na końcu każdego etapu osoba, która powiedziała liczbę równą wartości limitu dla tego etapu jest eliminowana i odchodzi z kółka. Pierwszy etap zaczyna osoba z numerem 1 (i ona mówi "jeden"). W kolejnych etapach odliczanie zaczyna kolejna osoba za tą, która właśnie została wyeliminowana.

Grę wygrywa ostatnia osoba, która pozostanie w kółku po wyeliminowaniu wszystkich pozostałych.

Napisz program, który na podstawie liczby uczestników gry, $N$, obliczy numer zwycięzcy.

## Wejście

W pierwszym wierszu standardowego wejścia znajduje się jedna liczba całkowita $1 \leq N \leq 5000$ – liczba uczestników gry.

## Wyjście

W pierwszym i jedynym wierszu program powinien wypisać jedną liczbę całkowitą – numer uczestnika, który zwyciężył (został w grze jako ostatni).

## Przykłady

1. Dla danych wejściowych:

   ```
   3
   ```

   poprawną odpowiedzią jest:

   ```
   2
   ```

2. Dla danych wejściowych:

   ```
   6
   ```

   poprawną odpowiedzią jest:

   ```
   6
   ```

# Zadanie 4. Valid strings

Rozważamy stringi o długości od 1 do $L$ złożone wyłącznie z małych liter alfabetu łacińskiego. Będziemy je nazywać **poprawnymi** stringami. Stringi są porównywane zgodnie z porządkiem leksykograficznym (czyli np. “car” $<$ “cat” i “cat” $<$ “cats”).

Dana jest liczba całkowita $L$ (maksymalna długość poprawnego stringu) i dwa stringi $A$ i $B$ takie, że $A < B$. Napisz program, który oblicza i drukuje liczbę poprawnych stringów $C$ spełniających warunek $A < C < B$.

## Wejście

W pierwszym wierszu standardowego wejścia znajduje się jedna liczba całkowita $1 \leq L \leq 10$ – maksymalna długość poprawnego stringu. W kolejnej linii znajdują się dwa poprawne stringi, $A$ i $B$ takie, że $A < B$.

## Wyjście

W pierwszym i jedynym wierszu program powinien wypisać jedną liczbę całkowitą – liczbę poprawnych stringów $C$ spełniających warunek $A < C < B$.

## Przykłady

1. Dla danych wejściowych:

   ```
   2
   ay c
   ```

   poprawną odpowiedzią jest:

   ```
   28
   ```

   W tym przypadku poprawnymi stringami są: az, b, ba, bb, ..., bx, by, bz.

2. Dla danych wejściowych:

   ```
   3
   car cas
   ```

   poprawną odpowiedzią jest:

   ```
   0
   ```

# Zadanie 5. Variables: Declaration hiding

Dany jest string zawierający symboliczną strukturę deklaracji zmiennych wewnątrz programu. Bloki programu zawarte są w nawiasach klamrowych (jak w języku C). W naszym programie istnieje tylko jeden blok zewnętrzny. Nazwy zmiennych są pojedynczymi literami alfabetu łacińskiego. Nazwa zmiennej w stringu oznacza jej deklarację wewnątrz określonego bloku. Na przykład string "{{ji}i}" oznacza, że w bloku zewnętrznym została zadeklarowana zmienna $i$, a w bloku wewnętrznym zmienne $i$ oraz $j$.

Napisz program, który znajduje i zwraca listę nazw zmiennych przesłaniających się (czyli takich, które zostały zadeklarowane w danym bloku i którymś z bloków w nim zawartych). Lista powinna być posortowana alfabetycznie i nie może zawierać powtórzeń.

Można założyć, że nawiasy są prawidłowo sparowane, jest dokładnie jeden blok zewnętrzny, oraz że zmienna nie jest nigdy deklarowana więcej niż raz w tym samym bloku (czyli string wejściowy "{a{b}a}" **nie** jest prawidłowy).

## Wejście

W pierwszym wierszu standardowego wejścia znajduje się string złożony wyłacznie z małych liter alfabetu łacińskiego i nawiasów klamrowych. String zawiera między 2 a 50 znaków. Pierwszym znakiem jest nawias otwierający '{' a ostatnim odpowiadający mu nawias zamykający '}'.

## Wyjście

W pierwszym i jedynym wierszu program powinien wypisać string złożony z posortowanych unikalnych nazw zmiennych przesłoniętych (bez spacji).

## Przykład

1. Dla danych wejściowych:

   ```
   {{ji}i}
   ```

   poprawną odpowiedzią jest:

   ```
   i
   ```

   Zmienna $i$ jest zadeklarowana w bloku zewnętrznym i wewnętrznym.

2. Dla danych wejściowych:

   ```
   {d{{e}{fd}}{e}{df{{a}}}a}
   ```

   poprawną odpowiedzią jest:

   ```
   ad
   ```