TAA - LEA 02

A. Bit Estranho, Desativar!

1 second, 256 megabytes

A cada segundo, milhares partículas chamadas de raios cósmicos, viajando perto da velocidade da luz, atingem cada metro quadrado da atmosfera da Terra, lançando uma cascata de partículas carregadas que caem na superfície do planeta. Quando essas partículas atingem os transistores de microchips, elas podem causar falhas e até mesmo provocar travamentos de computadores. Esse é um fenômeno raro, mas bits podem ser invertidos devido a uma ou várias dessas interferências.

Dessa vez, o problema é de excesso! Então, a empresa Inovação em Dispositivos Periféricos (IDP) pediu a sua ajuda para consertar possíveis excessos que possam ocorrer devido a essa interferência cósmica. Sua tarefa é a seguinte, após detectar que em um número um bit está estranho, você deve desativar esse bit, tranformando-o em 0, caso não o seja. Por exemplo, se na memória encontra-se o número 46, que está com problema no bit 3 (começando a contagem de zero e do bit menos significativo), a desativação desse bit fica o seguinte:

```
7 6 5 4 3 2 1 0    <- Posições
0 0 1 0 1 1 1 0    <- 46 em binário
_______-_____      <- Desativando-se o bit na posição 3
0 0 1 0 0 1 1 0    <- 38 em binário
```

Input

A primeira linha da entrada possui dois inteiros N ($0 < N \leq 100000$) e B ($0 \leq B \leq 31$), separados por espaço. As N linhas seguintes possuem N inteiros V ($0 \leq V \leq 10^9$) cada, que são os números cujo o bit na posição B deve ser desativado.

Output

A saída deve possuir N linhas, onde cada linha é o inteiro V_i com o respectivo bit desativado, conforme os exemplos.

input
1 3 46
output
38

input
3 2 7 48 18
output
3 48 18

B. Câmeras

1 second, 256 megabytes

No objetivo de modernizar as salas de aula do IDP, Lucas decidiu que seria a hora de implementar um sistema de frequências inteligente. Para isso, ele decidiu utilizar as câmeras de segurança para registrar a presença dos alunos.

Na hora de começar a implementar o sistema, ele percebeu que as câmeras não tinham cartões de memória para registrar tudo que ele precisava, e para resolver isso, ele decidiu que o melhor seria registrar apenas a presença dos alunos de maneira comprimida. O sistema de compressão de Lucas funciona da seguinte forma, nenhuma das turmas do IDP possui mais do que 64 alunos, então ele decidiu que ele poderia utilizar um inteiro para representar a presença dos alunos em uma aula, os quais ele fixou um índice para cada aluno.

Agora que o sistema está pronto, ele pediu a sua ajuda para validar o funcionamento do sistema. Assim, você deverá escrever um programa para, dada a chamada de uma turma feita pelo professor, responder qual deve ser a resposta comprimida gerada pelo sistema de Lucas.

Input

A entrada contém um único caso de teste. A primeira linha contém um inteiro N ($0 < N < 64$), que indica o número de alunos presentes na aula. A segunda linha contém N inteiros A_i ($0 \leq A_i < 64$), separados por um espaço, que indicam o índice do aluno que estava presente na aula.

Output

A saída deverá conter um único inteiro, que representa o valor esperado para o sistema de compressão de Lucas, conforme os exemplos.

input
7 2 1 7 4 3 8 10
output
1438

input
5 11 4 0 1 2
output
2071

input
3 8 0 2
output
261

C. XORING

1 second, 256 megabytes

Vamos definir uma nova operação sobre duas strings R e S chamada de XORING. Dadas duas strings R e S de mesmo tamanho N, a operação XORING é definida como a string T de tamanho N tal que dois caracteres R_i e S_i são sempre a mesma letra, podendo diferenciar-se apenas em caixa alta ou baixa. Considerando C e c como a mesma letra em caixa alta e baixa, temos que a operação XORING, representada por $\oplus$, é definida como:

$$C \oplus C = C$$

$$C \oplus c = c$$

$$c \oplus C = c$$

$$c \oplus c = C$$

Sua tarefa é, dadas duas strings R e S de tamanho N, calcular a string T resultante da operação XORING entre R e S.

Input

A primeira linha da entrada contém um inteiro N ($1 \leq N \leq 100$), que representa o número de casos de testes. Cada um dos próximos N linhas contém duas strings R e S de mesmo tamanho N ($1 \leq N = |R| = |S| \leq 100$), separadas por um espaço.

Output

A saída deverá conter N linhas, cada uma contendo cada uma uma string T de tamanho N, que representa a string resultante da operação XORING entre R e S.

input
3 idp IDP idP IDP idP idp

output
idp idP IDp

input
2 bAnAnA BaNaNa xor xor

output
banana XOR

D. Bitswap

1 second, 256 megabytes

Neste exercício, você deverá trocar os valores de dois bits de um número inteiro, como mostrado no exemplo abaixo.

```
7 6 5 4 3 2 1 0      <- posições dos bits

0 0 0 0 1 0 1 0      <- número inicial: 10 (decimal)
        |     |
        |     |      - exemplo trocando as posições 0 e 3
        └─────┘
0 0 0 0 0 0 1 1      <- número final: 3 (decimal)
```

Input

A entrada contém múltiplos casos de testes. A primeira linha de entrada contém um inteiro T ($1 \leq T \leq 10000$) que indica o número de casos de testes. A T linhas seguintes contém três inteiros N ($1 \leq N \leq 2^{64} - 1$), P e Q ($0 \leq P, Q \leq 63$), onde N é o número a ser modificado e P e Q são as posições dos bits que deverão ser trocados.

Output

A saída deve conter T linhas, cada uma com um inteiro representando o número modificado, conforme os exemplos.

input
1 10 0 3

output
3

input
2 38 4 1 1 0 32

output
52 4294967296