ABC 109 解説

writer: drafear

2018 年 9 月 8 日

A: ABC333

$A \times B$ が奇数なら $C = 1$ とすることで $A \times B \times C$ が奇数になります。一方、$A \times B$ が偶数なら、偶数にどんな自然数をかけても偶数のままです。従って、$A \times B$ が偶数かどうかで場合分けするとスマートです。C++ による実装例を以下に挙げます。

```
#include <iostream>

using namespace std;

int main() {
  int A, B; cin >> A >> B;
  if (A * B % 2 == 1) {
    cout << "Yes" << endl;
  }
  else {
    cout << "No" << endl;
  }
}
```

しかし、この事実に気付かなくとも全ての $C = 1, 2, 3$ に対して $A \times B \times C$ の偶奇を調べる方法でも正答できます。その場合の C++ による実装例は次のようになります。

```
#include <iostream>

using namespace std;

int main() {
  int A, B; cin >> A >> B;
  if (A * B * 1 % 2 == 1 || A * B * 2 % 2 == 1 || A * B * 3 % 2 == 1) {
    cout << "Yes" << endl;
  }
```

```
  else {
    cout << "No" << endl;
  }
}
```

B: Shiritori

問題文の条件を言い換えると、次のようになります。

1. $1 \le i \le n-1$ を満たす各 i について W_i の最後の文字と W_{i+1} の最初の文字が等しい
2. $1 \le i < j \le n$ を満たす各 i, j について $W_i \ne W_j$

これを C++ で実装した例を以下に示します。

```cpp
#include <iostream>
#include <vector>

using namespace std;

bool solve(const vector<string>& W) {
  const int n = W.size();
  // condition 1
  for (int i = 0; i < n-1; ++i) {
    if (W[i].back() != W[i+1].front()) {
      return false;
    }
  }
  // condition 2
  for (int i = 0; i < n; ++i) {
    for (int j = i+1; j < n; ++j) {
      if (W[i] == W[j]) {
        return false;
      }
    }
  }
  return true;
}
vector<string> input() {
  int n; cin >> n;
  vector<string> W(n);
  for (int i = 0; i < n; ++i) {
    cin >> W[i];
  }
  return W;
}
```

```
int main() {
  const vector<string> W = input();
  if (solve(W)) {
    cout << "Yes" << endl;
  }
  else {
    cout << "No" << endl;
  }
}
```

余談: std::set を用いてより高速に判定することもできますが、今回の場合はこれで十分です。

C: Skip

D が全ての i について $|X - x_i|$ の約数であるとき、全ての都市を訪れることができます。逆に、D が $|X - x_i|$ の約数でないような i が存在するとき、操作によって現在居る座標を D で割った余りが変化しないことから、$|X - x_i|$ を D で割った余りが $|X - x_i|$ または $D - |X - x_i|$ となり、0 になることがないため、座標 x_i を訪れることができません。

したがって、$|X - x_1|, |X - x_2|, ..., |X - x_N|$ 全ての約数であって最大の正整数 D を求めれば良いことになります。このような D は最大公約数と定義され、ユークリッドの互除法により高速に計算できることが知られています。ユークリッドの互除法とは、

$$\mathsf{gcd}(a,b) = \begin{cases} \mathsf{gcd}(b, a \ \mathsf{mod}\ b) & \mathsf{if}\ b > 0 \\ a & \mathsf{if}\ b = 0 \end{cases}$$

が成り立ち、これに従って計算するアルゴリズムのことです。ここで、$\mathsf{gcd}(a,b)$ は a と b の最大公約数表し、$a \ \mathsf{mod}\ b$ は a を b で割った余りを表します。ユークリッドの互除法は $a \leq b$ のとき 1 回の反復で $a \ \mathsf{mod}\ b < \frac{a}{2}$ となることから $O(log \max(a,b))$ で求まります。ちなみに、a, b がフィボナッチ数のときに遅くなりますが log で求まることには変わりありません。$\mathsf{gcd}(a,b,c) = \mathsf{gcd}(\mathsf{gcd}(a,b),c)$ なので、答えである $\mathsf{gcd}(|X - x_1|, |X - x_2|, ..., |X - x_N|)$ を $O(N log \max\{|X - x_1|, |X - x_2|, ..., |X - x_N|\})$ で求められます。

D: Make Them Even

次の手順で、$\sum a_{ij}$ が偶数なら全て偶数にでき、奇数なら 1 マスを除いて全て偶数にできます。

まず、一筆書きの経路を作ります。例えば、

$$(1,1),(1,2),...,(1,W),(2,W),(2,W-1),...,(2,1),(3,1),(3,2),...$$

です。この経路が通る i 番目のマスを (Y_i, X_i) とします。次に、$i = 1, 2, ..., H \times W - 1$ の順に $a_{Y_i X_i}$ が奇数ならマス (Y_i, X_i) からマス (Y_{i+1}, X_{i+1}) に箱を 1 個移動させる操作をします。この操作により、$a_{Y_i X_i}$ に -1、$a_{Y_{i+1} X_{i+1}}$ に 1 が加算され、$a_{Y_i X_i}$ が偶数になります。一連の操作は valid であり、マス $(Y_{H \times W}, X_{H \times W})$ を除いた全てのマスを偶数にできます。時間計算量は $O(HW)$ になります。