AtCoder Beginner Contest 008

解説

AtCoder株式会社

競技プログラミングを始める前に

AtCoder

・競技プログラミングの基本的なこと(標準入出力など)は過去の ABC の解説にも描かれてあります。

・特に ABC004 のものが詳しく書かれてあります。

・過去問を学ぶことは有益なことですので、積極的に活用しましょう。

AtCoder

A問題

アルバム

A問題　問題概要

AtCoder

- 2 つの整数 S,T が与えられます。
- S 以上 T 以下の整数の個数を計算してください。

- 1 ≦ S ≦ T ≦ 1,000

A問題　入出力

AtCoder

- 標準入力から 2 つの整数を読み込んで、答えを計算して、1 つの整数を出力してください。
- 出力の末尾には改行を入れてください（改行コードの種類に注意すること）。
- 標準入出力の使い方等に関しては、過去の ABC 解説や練習ページに詳しい説明があります。
- 練習ページ: http://practice.contest.atcoder.jp/

A問題　計算

AtCoder

- S 以上 T 以下の整数は、S に 1 つずつ数を足していって、T に等しくなるまでに足した回数に 1 を加えた値 (S 自身) が答えとなります。
- 1 つずつ数え上げなくても、T−S+1 という式を計算することでも求めることができます。

A問題　計算の補足

AtCoder

- 前者のアルゴリズムは O(T−S)、後者のアルゴリズムは O(1) ということができます。
- このような計算量の考え方は、アルゴリズムの設計において役に立つことがあるので、余裕があれば是非とも習得してください。
- 今回の問題ではどちらの方針でも解くことができます。

B問題

投票

B問題　問題概要

AtCoder

- 整数 N と、N 個の名前が与えられます。
- 最も多く重複して現れる名前を出力してください。
- 条件を満たす名前が複数ある場合は、そのうちどれでも構いません。

- 1 ≦ N ≦ 50
- 1 ≦ (それぞれの名前の長さ) ≦ 50

B問題　ステップ

AtCoder

- この問題には 2 つの重要な点があります。

1. ループ処理について

2. 文字列の処理について

B問題　ステップ 1

AtCoder

1. ループ処理について

- この問題では入力によって読み込む回数が変わるので、ループ関数などを用いて読み込みを行うことになります。
- 後述する文字列についても、ループ関数が密接に関わってきます。

※言語によっては実装の方針などが変わる場合があります。その場合はその言語に対応したプログラムを用いてください。

B問題　ステップ 2

AtCoder

2. 文字列の処理について

- 文字列の処理は、言語によって様々な仕様があります。
- 予め文字列の問題を幾つか解いておいて、文字列をどのように扱えばいいのかを学んでおきましょう。
- ライブラリ関数（C だと strcmp とか）を使用すると実装が用意になることがあります。
- 原理を知っておくのが望ましいです。

B問題　解法

AtCoder

- 各文字列ごとに、その文字列と完全に一致する他の文字列の個数を計算し、配列等に格納しておきます。
- 最大の要素が格納された場所を調べ、その場所に対応する文字列を出力します。
- 文字の長さの最大値を S とおいたとき、O(SN^2) で計算することができます。

B問題　補足

AtCoder

- 言語によっては、文字列を比較したつもりでも、実際は文字列のポインタ同士の比較になってしまう場合などがあります。注意しましょう。
- 別解として、文字列同士の比較の代わりにハッシュ関数同士の比較を用いた解法もあります。ただしこの場合はハッシュの衝突に注意する必要があります。

AtCoder

C問題

コイン

C問題　問題概要

AtCoder

- N 枚のコインを無作為に一列に並べます。
- 左端から順に、そのコインよりも右側にある、そのコインに書かれた数の倍数が書かれたコインをすべてひっくり返す動作をします。
- 最終的に表を向いている(偶数回反転した)コインの枚数の期待値を計算してください。

- $1 \leqq N \leqq 100$

C問題　浮動小数点について

AtCoder

- この問題では浮動小数点が出てきます。
- 文字列同様、浮動小数点の扱いにも慣れておきましょう。
- 整数型と浮動小数点型間の移行などの際には注意してください。

C問題　解法その 1

AtCoder

- N！通りのすべての組み合わせを作り、実際に実験してみようと考えます。
- N ≦ 8 であれば、計算量が O(N!) でも大丈夫なことが多いです（今回も、この方針で 99 点を得ることができます）。
- 残り 1 点を得るには N ≦ 100 の制約に対処する必要があります。

C問題　解法その 1

AtCoder

- もしも、N！通りのすべての組み合わせを N=100 において計算しようとなると、それにはとてつもない時間がかかることになります(2 sec の時間制限は大幅に超えてしまう)。
- より高速に動作するアルゴリズムを設計する必要があります。

C問題　数学の時間

AtCoder

- 期待値の性質を用いることで高速に解くためのアプローチが得られます。
- 求める期待値は、すべての組み合わせのうちそれぞれにおいて表を向いているコインの枚数の総合計を N! で割ったものとして計算できます。
- コインの枚数の総合計について考えます。

C問題　数学の時間

AtCoder

- コインの枚数の総合計は、組み合わせごとに区切って数えても、各コインが全組み合わせのうち何通りの組み合わせで表を向くのかという数え方で数えても一致します。
- よって、それぞれのコインについて、そのコインが、何通りの並べ方で表を向くのかという方針で計算することにします。

C問題　数学の時間(補足)

AtCoder

- 補足として、全体を足しあわせてから N! で割った結果と、それぞれの要素を N! で割ったものの合計は等しくなります。
- それぞれのコインごとに、表を向いている組み合わせ数を N! で割った値は、そのコインが表を無いた状態で終了する確率に等しくなります。
- 今回の場合、この確率の足し合わせが期待値になります。

C問題　解法その 2

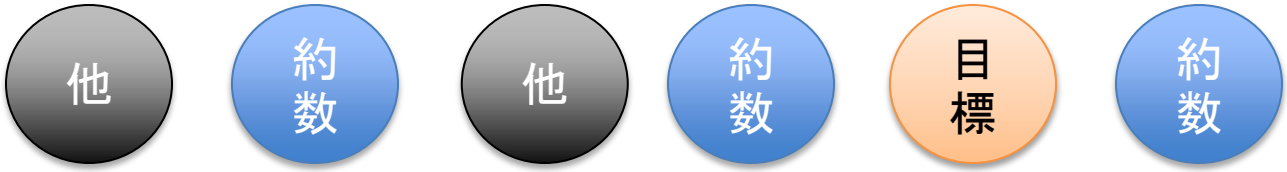
AtCoder

- あるコインについて、このコインに書かれた数を C としたとき、どのような条件を満たせばこのコインが最終的に表を向いているかを考えます。
- このコインよりも左側にあるコインに書かれた数のうち、C の約数となっているものの枚数が奇数枚なら裏、偶数枚なら表となります。

C問題　解法その 2

AtCoder

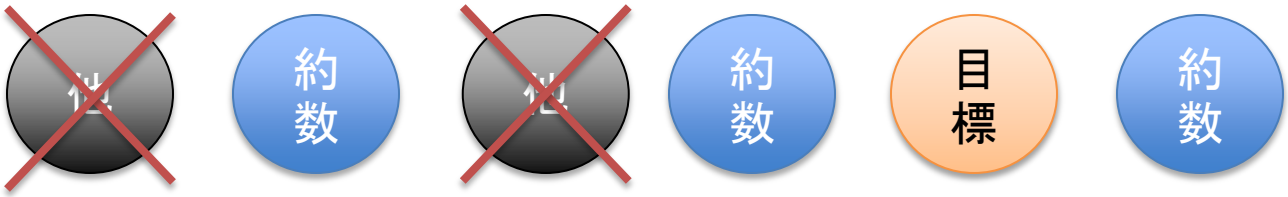
- 全コインのうち、目標のコインと、それ以外の中で C の約数となっているコイン(S 枚あるとする)の並びについて考えます。



C問題　解法その 2

AtCoder

- 全コインのうち、目標のコインと、それ以外の中で C の約数となっているコイン(S 枚あるとする)の並びについて考えます。
- C の約数でないコインの並びは関係なく、純粋に目標のコインより左に何個 C の約数があるかを考えます。



C問題　解法その 2

AtCoder

- 目標のコインについて、そのコインが左から何番目にあったとしても、C の約数内では S! 通りの並べ方があります。
- つまり、左から 1 番目、2 番目、… (S+1) 番目のいずれも当確率で出てくることになります。
- より左側に偶数枚ということは、左から奇数枚目にある確率が、目標のコインが表を向いている確率に等しくなります。

C問題　解法その 2

AtCoder

- この確率は、S が奇数なら 1/2 , S が偶数なら (S+2)/(2S+2) の確率になります。
- それぞれのコインごとに、上記の確率を求め、足し合わせることで答えが得られます。
- 計算量は O(N^2) となり、高速に答えを計算することが出来ます。

AtCoder

D問題

金塊ゲーム

D問題　問題概要

AtCoder

- N 個の装置と W×H 個の金塊についての情報が与えられます。
- 最適に装置を動かして、できるだけ多くの金塊を取ってください。

- 1 ≦ N ≦ 30
- 1 ≦ R ≦ 10^9
- 1 ≦ C ≦ 10^9

D問題　解法その 1

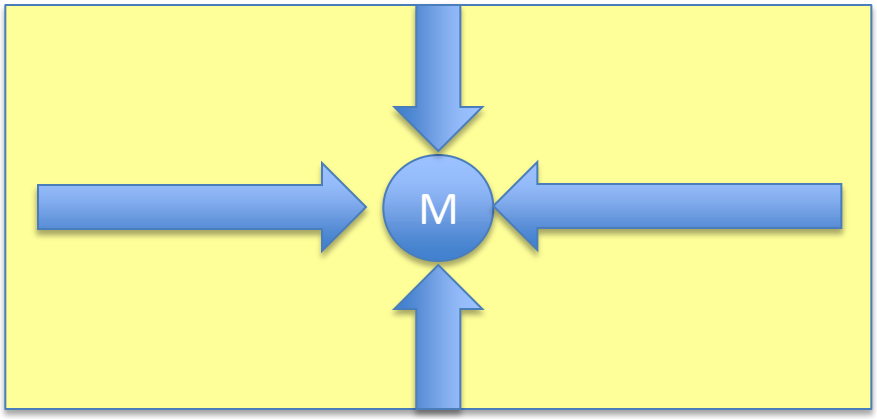
AtCoder

- C 問題同様、N！通りのすべての組み合わせをシミュレーションする解法があります。
- 計算量は O(R×C×N!) で、この方針で 80 点を得ることができます。
- さらに高い点数を得るためにはこの問題についての考察が必要になります。

D問題　考察

AtCoder

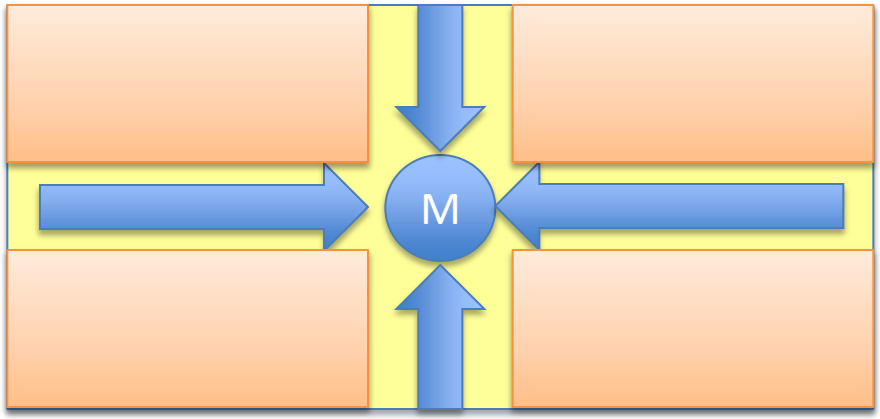
- ある装置を稼働させた後の状態を考える。



D問題　考察

AtCoder

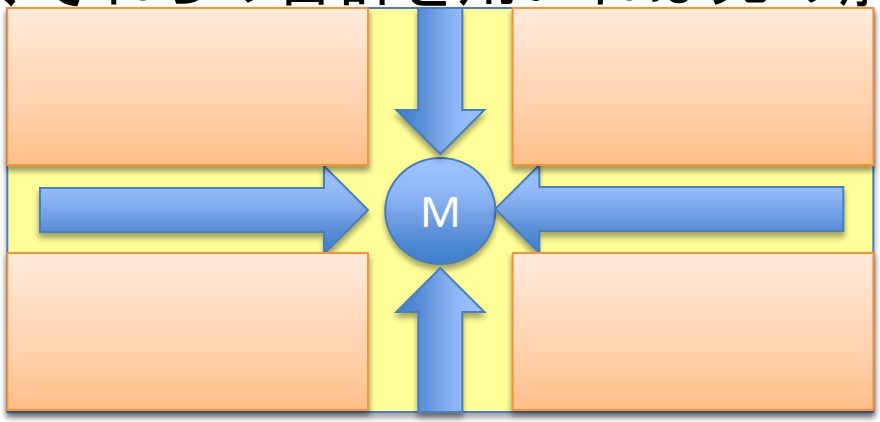
- ある装置を稼働させた後の状態を考える。
- 下図のように、1 つの長方形領域が最大 4 つの長方形領域に分かれる。



D問題　考察

AtCoder

- ある装置を稼働させた後の状態を考える。
- 下図のように、1 つの長方形領域が最大 4 つの長方形領域に分かれる。
- これらの領域同士は互いに干渉しないので、独立に最適解を求めて、それらの合計を用いれば元の解が求まる。



D問題　考察

AtCoder

- このように、ある大きさにおける最適解を求める際に、より小さい部分の最適解を求めてその値を用いて元の最適解を求める手法一般を動的計画法と言います。
- この問題では動的計画法を用いて効率的に解くことができます。

D問題　解法その 2

AtCoder

- dp[i][j][k][l]=(左下を(i,j), 右上を(k,l) とした長方形領域が残った際の最適解)
- とおくことにより、O(R^2×C^2) の空間計算量、各 dp で O(N) 通りの選択と適用により計算できるので、全体で O(N×R^2×C^2) となります。(99点)

D問題　補足

AtCoder

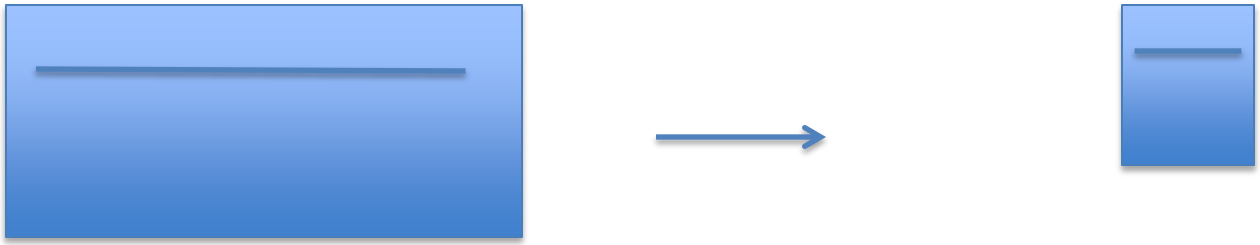
- 動的計画法を実行する際には、今回の場合、長方形の面積が小さい順から行います。
- また、今回の場合、参照される長方形のうち最初のもの以外のすべてはある装置がつくる 2 辺を用いるので、実際は状態数は少ないです。

D問題　解法その 3

AtCoder

- Ｒ と Ｃ が大きい場合は、これまでの方法では状態が多すぎて間に合いません。
- しかしながら、直線が連続する区間とかは圧縮して計算できそうです。

コンパクトにまとめる



D問題　補足

AtCoder

- 今回の問題では、

  →動的計画法: 座標圧縮で節約

  →メモ化探索: 余計な状態を生成しない

という方針で対策することができます。

- この方針であれば 100 点を得ることが出来ます。
- メモリ制限には注意してください！