教科書輪講 第1回

プログラミングコンテスト攻略のためのアルゴリズムとデータ構造

第2章

石田研

M1 賀来智博

kaku@cb.cs.titech.ac.jp

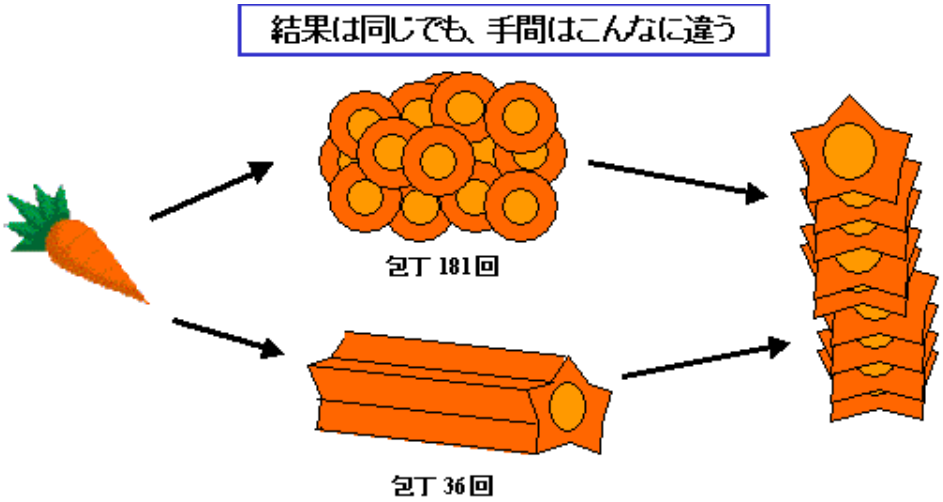
本日の内容

1. アルゴリズム
2. 計算量とは
3. 計算量の比較
4. 例題

アルゴリズム

コンピューターで計算を行う時の、計算方法のこと

同じ問題に対しての計算方法でも、やり方によって、
高速であったり、メモリ消費量が少なかったり
善し悪しがあるので、状況に応じて使い分けましょう



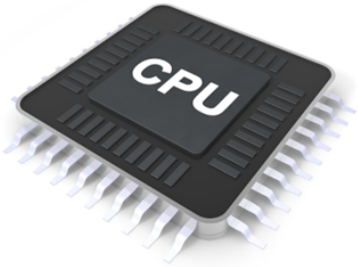
http://research.nii.ac.jp/~uno/algo_3.htm

計算量とは

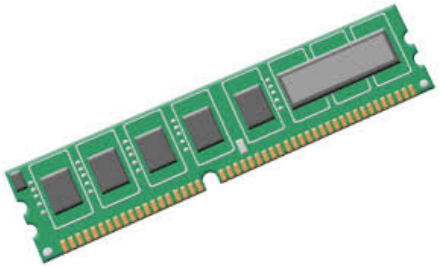
アルゴリズムには効率の良さがある

本教科書では
こっちを扱う

・時間計算量：プログラムの実行に必要な時間を評価する。



・領域計算量：プログラムの実行に必要な記憶領域を評価する。



計算量の表記

計算量はO表記法で表される
nを問題の入力サイズとしたとき
$O(g(n))$は計算量が$g(n)$に比例することを表す

与えられた問題には入力の上限があるので、
それを元にアルゴリズムの評価を行なう

計算量の求め方

1)プログラムのステップ数を数える

```
void calculate(int n) { //入力サイズをnとします
    int count = 0;      //1回実行

    for (int i = 0; i < n; i++) {
        count++;        //n回実行
    }

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            count++;    //n^2回実行
        }
    }

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            count++;    //n^2回実行
        }
    }
}
```

2)プログラム全体のステップ数を数える

$$1 + n + n^2 + n^2$$
$$= 1 + n + 2n^2$$

3)不要なものを除く

*a.*最大次数の項以外を除く

$$1 + n + 2n^2 \rightarrow 2n^2$$

*b.*係数を除く

$$2n^2 \rightarrow n^2$$

このコードの計算量は$O(n^2)$

計算量の比較

n	$logn$	$\sqrt{n}$	$nlogn$	n^2	2^n	$2!$
10,000	13	100	130,000	10^8	約$10^{3,000}$	約$10^{35,660}$
10,0000	19	316	1,600,000	10^{10}	約$10^{30,000}$	約$10^{456,574}$

SAFE ⬅ | ➡ OUT

例題

1. FX取引では、異なる国の通貨を変換することで
   為替差の利益を得ることが出来る。

   例えば、1ドル100円のときに1,000ドル買い、
   価格変動により1ドル108円になった時に売ると、
   (108円 ー 100円) × 1,000ドル = 8,000円の
   利益を得ること事ができる。

   ある通貨について、時刻tにおける
   価格$R_t(t = 0,1,2 \ldots, n-1)$が入力として与えられるので、
   価格の差$R_j - R_i$(ただし、$j > i$)の最大値を求めてください。

例題

入力：
最初の行に整数nが与えられる。
続くn行に整数$R_t(t = 0,1,2 \ldots, n-1)$が順番に与えられます。

出力：
最大値を1行に出力してください。

制約：

$$2 \leq n \leq 200{,}000$$
$$1 \leq R_t \leq 10^9$$

例題

入力例１：6 5 3 1 3 4 3

出力例１：3

入力例２：3 4 3 2

出力例２：-1

解説

そのアルゴリズム

a)最大利益の初期値やばくない？

b)計算量$O(n^2)$じゃない？

a)最大利益の初期値やばくない？
　最大利益が最悪の場合を考えた時、
　初期値は$1 \leq R_t \leq 10^9$より

$10^9 \times (-1)$　　または　　$R_1 - R_0$

が最適

解説

b)計算量$O(n^2)$じゃない？

```
for (int j = 0; j < n-1; j++){
    for (int i = 0; i < j - 1; i++){
        maxv = max(maxv, R[j] - R[i])
    }
}
```

ダメアルゴリズム

できるアルゴリズムは、
あるR_j以前の最小値を毎回探索しないで保持しておくの

```
minv = R[0]
for (int j = 0; j < n - 1; j++){
    maxv = max(maxv, R[j] - minv);
    minv = min(minv, R[j]);
}
```

このコードの計算量は$O(n)$

おまけ

ここに整数が書かれた複数のカードがあります
この中から3枚を選び、そこに書かれた整数の和が、
7で割り切れるかどうかという遊びをします

カードは必ず異なる3枚を選ぶ必要があり、
全てのカードには異なる数字が書いてあります

適当な複数のカードに対して、カードに書かれた3つの整数を足した和が7で割り切れるような組み合わせがいくつあるかを計算するプログラムを作成してください

おまけ

入力：
最初の行にカードの枚数として整数nが与えられる。
続くn行に整数$a_i (i = 0,1,2 \ldots, n-1)$が順番に与えられます。

出力：
組み合わせ数を1行に出力してください。

制約：

$$1 \leq n \leq 100{,}000$$
$$1 \leq a_i \leq 2^{32}$$

おまけ

入力例１：3 13 4 14

出力例１：1

入力例２：10 1 2 3 4 5 6 7 8 9 10

出力例２：17