Ramen 解説

2014/03/20 情報オリンピック春合宿にて

MASAKI HARA

問題概要

問題概要

ラーメン屋がN軒ある

問題概要

ラーメン屋がN軒ある

「**こってり度**」が定まっている

問題概要

ラーメン屋がN軒ある

「**こってり度**」が定まっている　　こってり度は互いに異なる

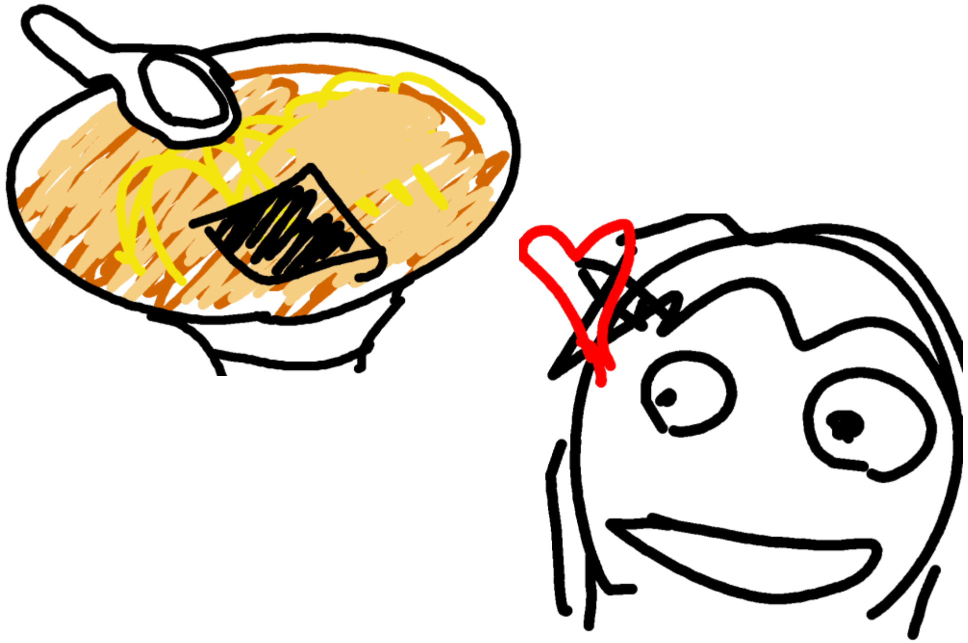
問題概要

JOI君はあっさりしたラーメンが好き

問題概要

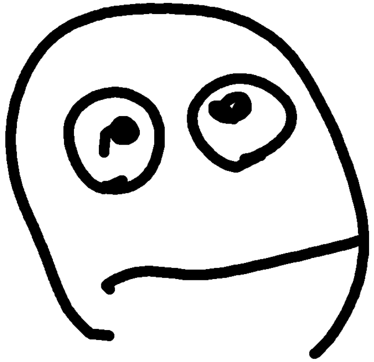
JOI君はあっさりしたラーメンが好き

IOIちゃんはこってりしたラーメンが好き



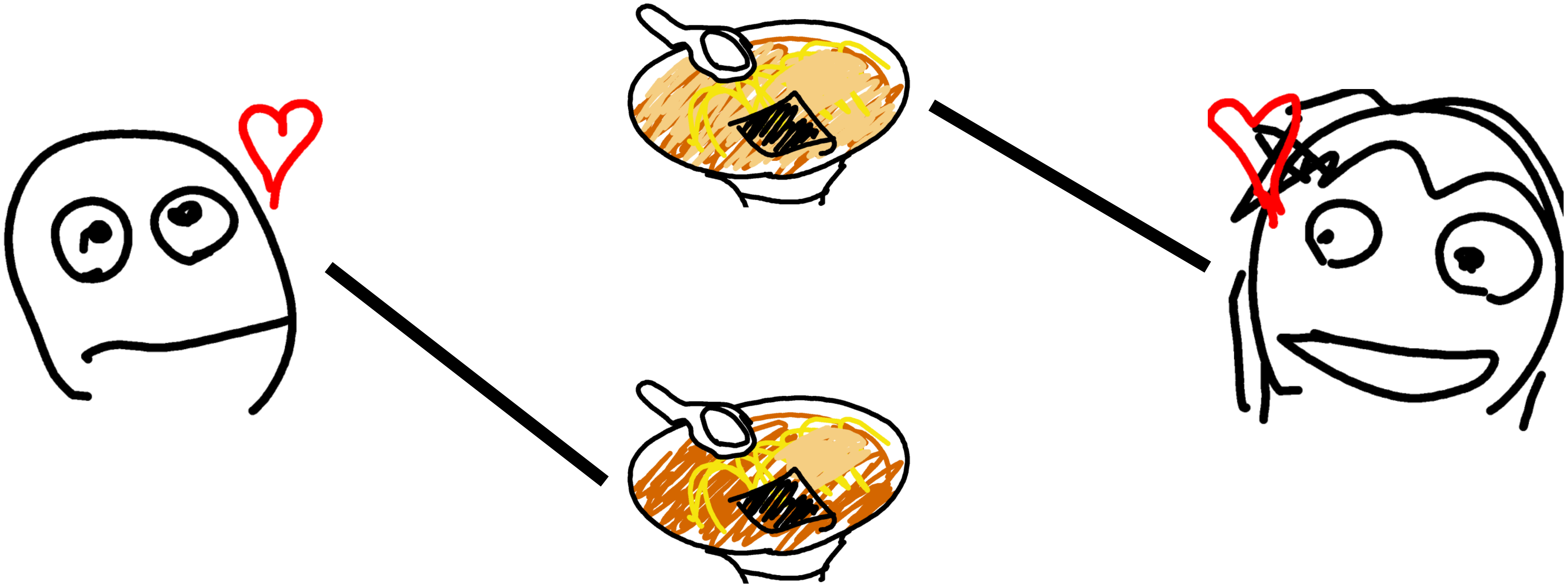
問題概要

1日一回、食べ比べを行う



問題概要

1日一回、食べ比べを行う　　　　二つのラーメン屋のこってり度の大小関係がわかる



問題概要

健康のため、食べ比べは**600日**より多くは行えない

問題概要

健康のため、食べ比べは**600日**より多くは行えない

問題概要

健康のため、食べ比べは**600日**より多くは行えない

一番あっさりしたラーメンと一番こってりしたラーメンを決めたい

小課題1 (20点)

$N \leq 30$

小課題1 (20点)

$N \leq 30$

ラーメン屋の組み合わせは$\frac{N \times (N-1)}{2} \leq 435$

小課題1 (20点)

$N \leq 30$

ラーメン屋の組み合わせは$\frac{N \times (N-1)}{2} \leq 435$

すべての組み合わせについてCompareを呼び出しても間に合う

小課題2 (累計50点)

$N \leq 300$

小課題2 (累計50点)

$N \leq 300$

こってり度最大を300手で決定できればよい

小課題2 (累計50点)

$N \leq 300$

こってり度最大を300手で決定できればよい (最小も同様にできるので)

小課題2 (累計50点)

$N \leq 300$

こってり度最大を300手で決定できればよい (最小も同様にできるので)

普通の最大値決定アルゴリズムでOK

小課題2 (累計50点)

普通の最大値決定アルゴリズム

小課題2 (累計50点)

普通の最大値決定アルゴリズム

小課題2 (累計50点)

普通の最大値決定アルゴリズム

暫定的な最大を決める

小課題2 (累計50点)

普通の最大値決定アルゴリズム

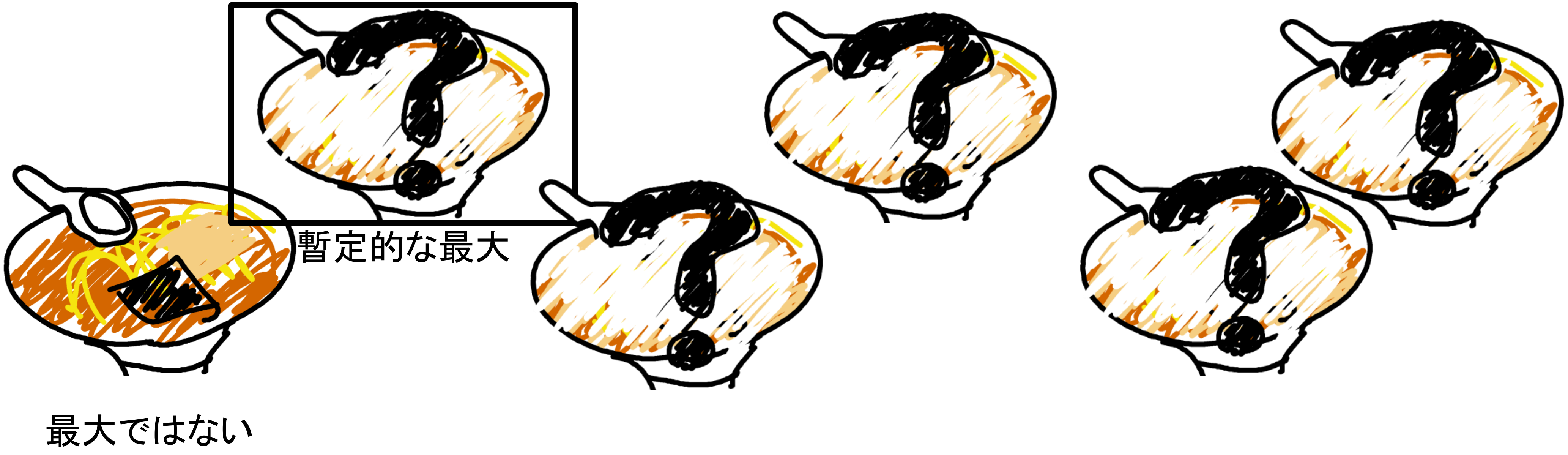
比較する

暫定的な最大

小課題2 (累計50点)

普通の最大値決定アルゴリズム

比較する→低いほうはリタイヤ



小課題2 (累計50点)

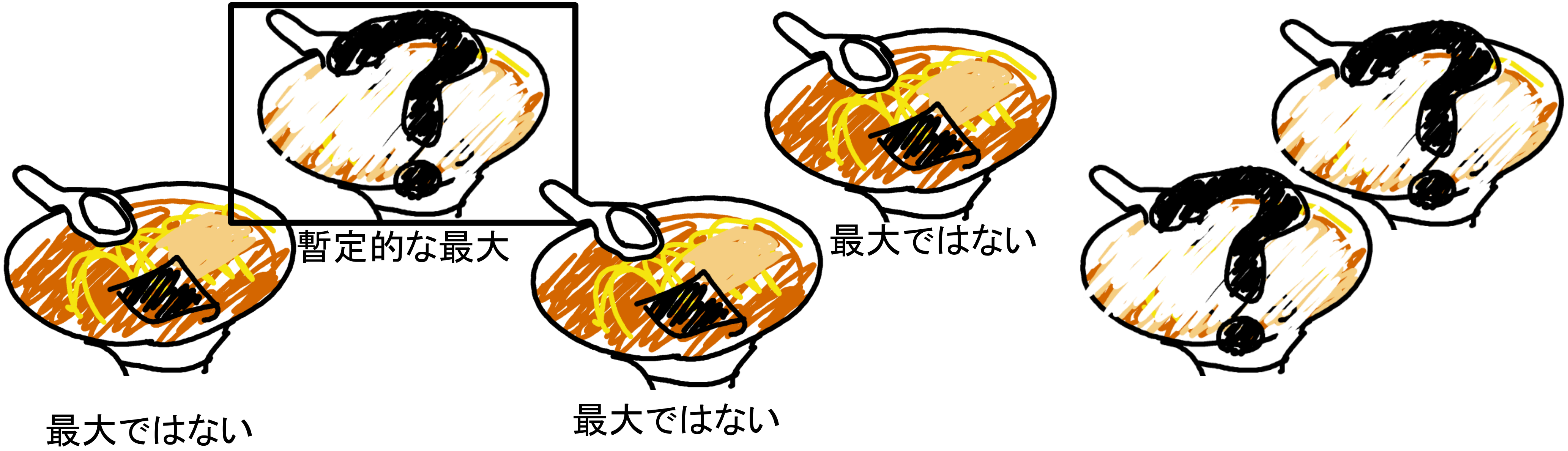
普通の最大値決定アルゴリズム

比較する→低いほうはリタイヤ

小課題2 (累計50点)

普通の最大値決定アルゴリズム

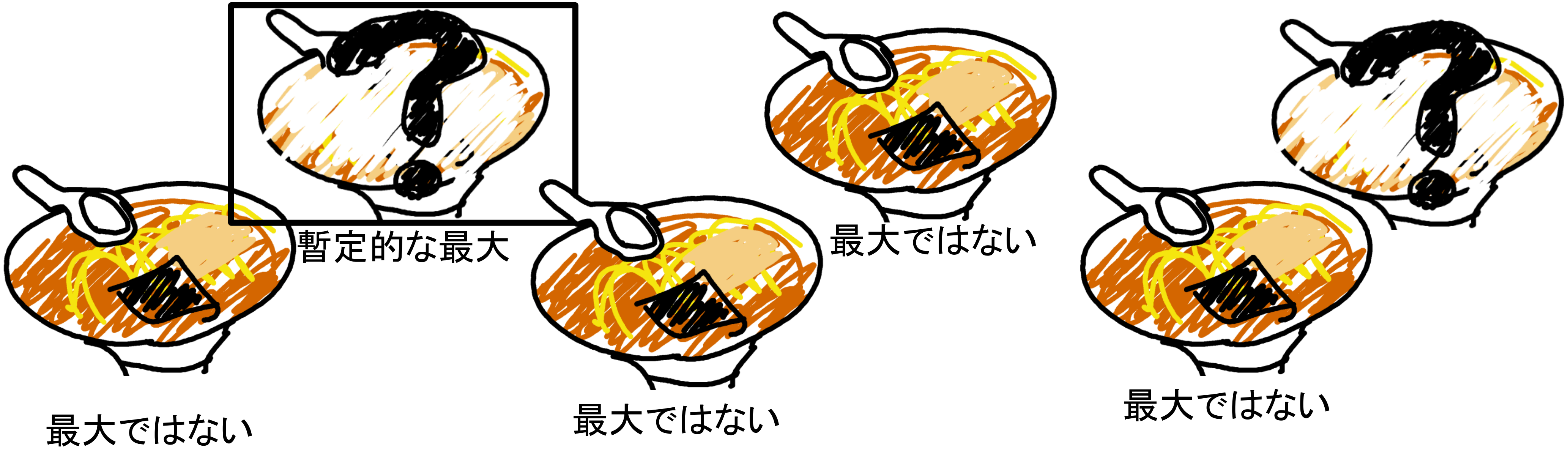
比較する→低いほうはリタイヤ



小課題2 (累計50点)

普通の最大値決定アルゴリズム

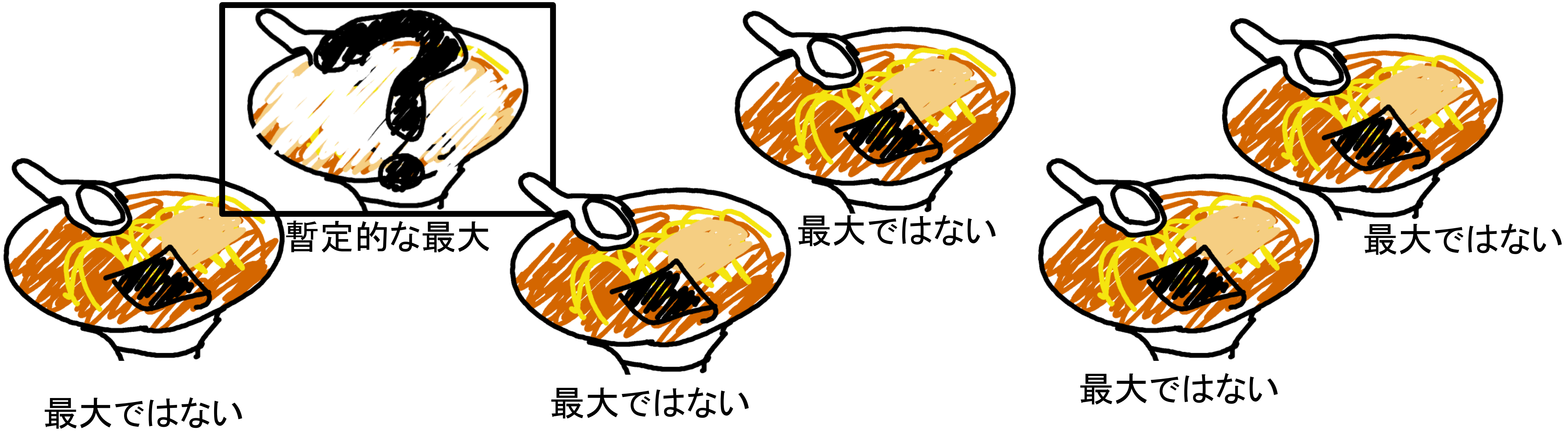
比較する→低いほうはリタイヤ



小課題2 (累計50点)

普通の最大値決定アルゴリズム

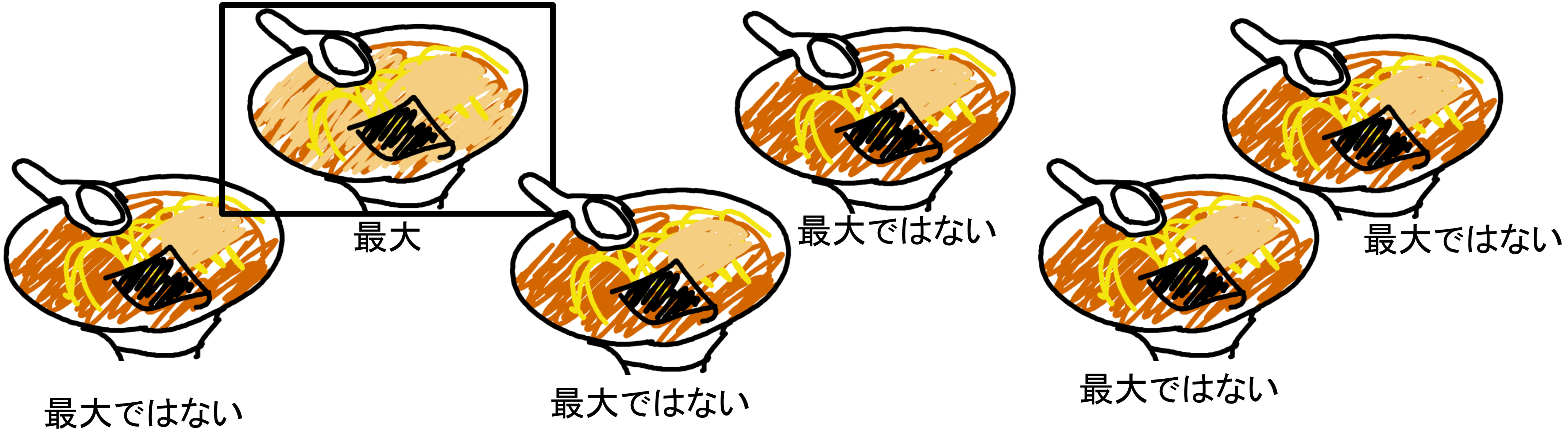
比較する→低いほうはリタイヤ



小課題2 (累計50点)

普通の最大値決定アルゴリズム

比較する→低いほうはリタイヤ



小課題2 (累計50点)

最大値を決めるのに$N-1$回の比較でできる

小課題2 (累計50点)

最大値を決めるのに$N-1$回の比較でできる

最大値と最小値を決めるのに$2(N-1) \leq 598$回でできる

小課題3 (累計100点)

$N \leq 400$

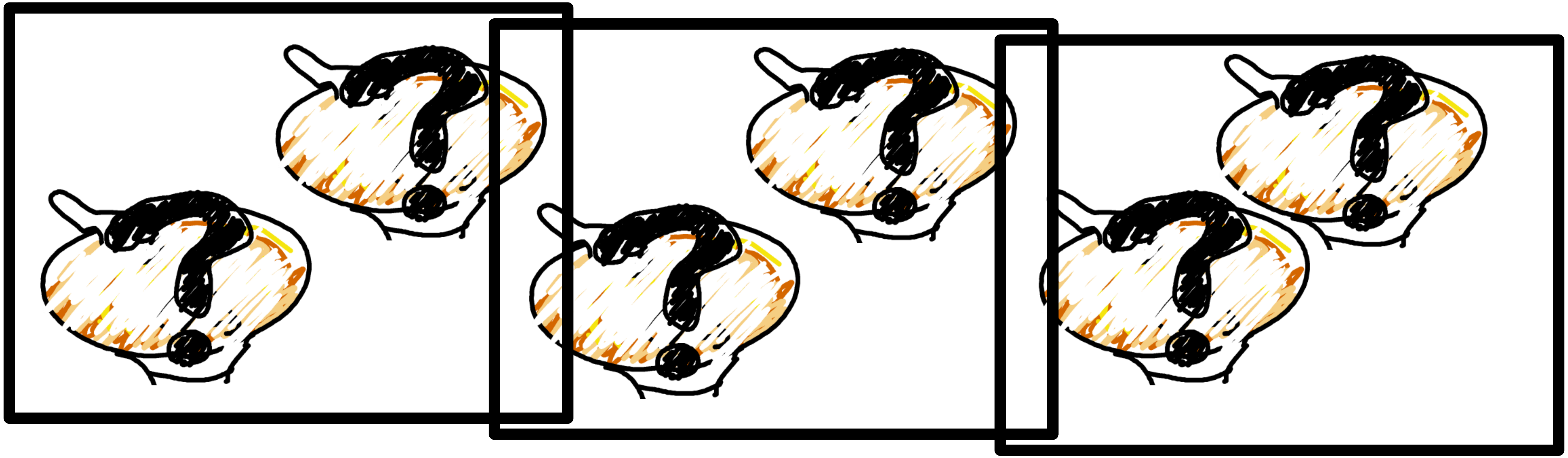
小課題3 (累計100点)

$N \leq 400$

小課題3 (累計100点)

$N \leq 400$

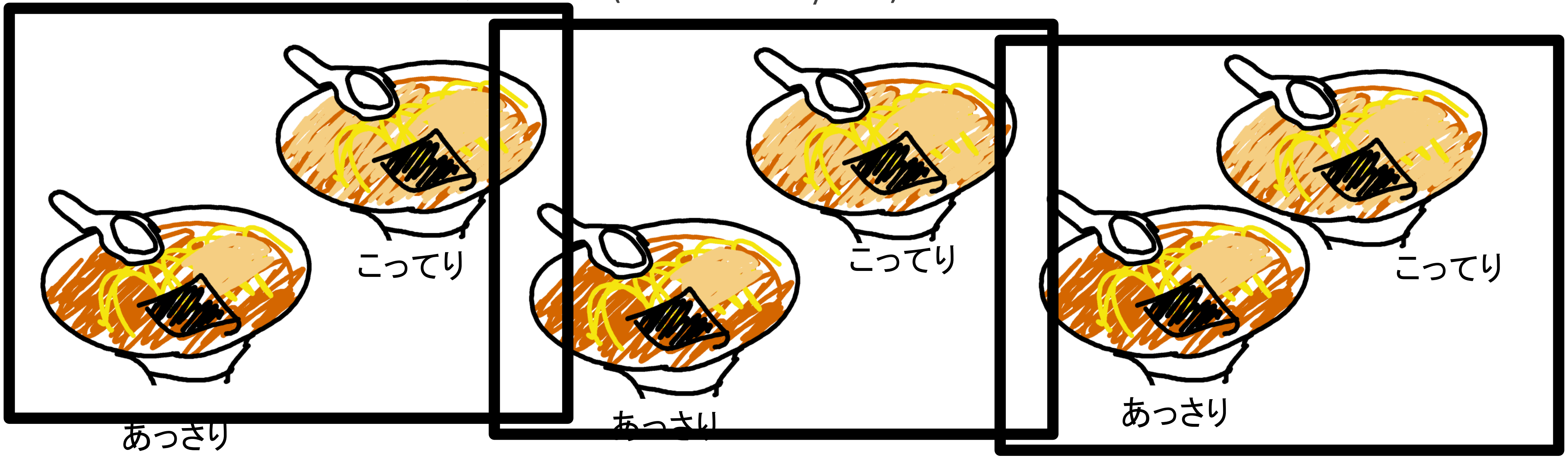
2つずつに分ける



小課題3 (累計100点)

$N \leq 400$

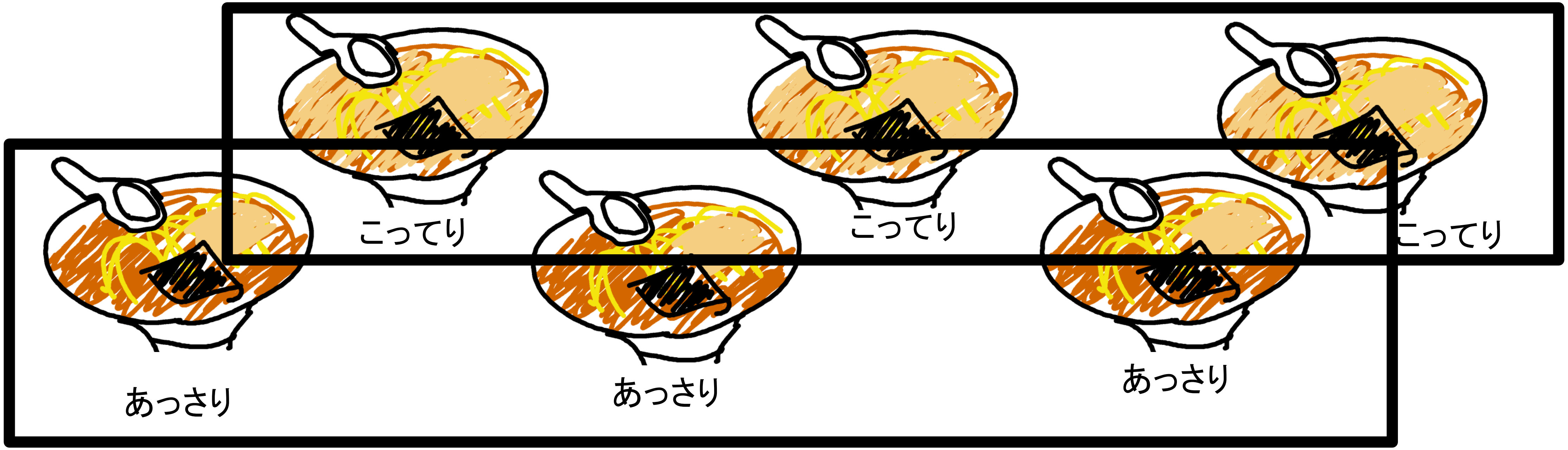
2つずつに分ける→こってり度を比較 (ここまでで$N/2$回)



小課題3 (累計100点)

$N \leq 400$

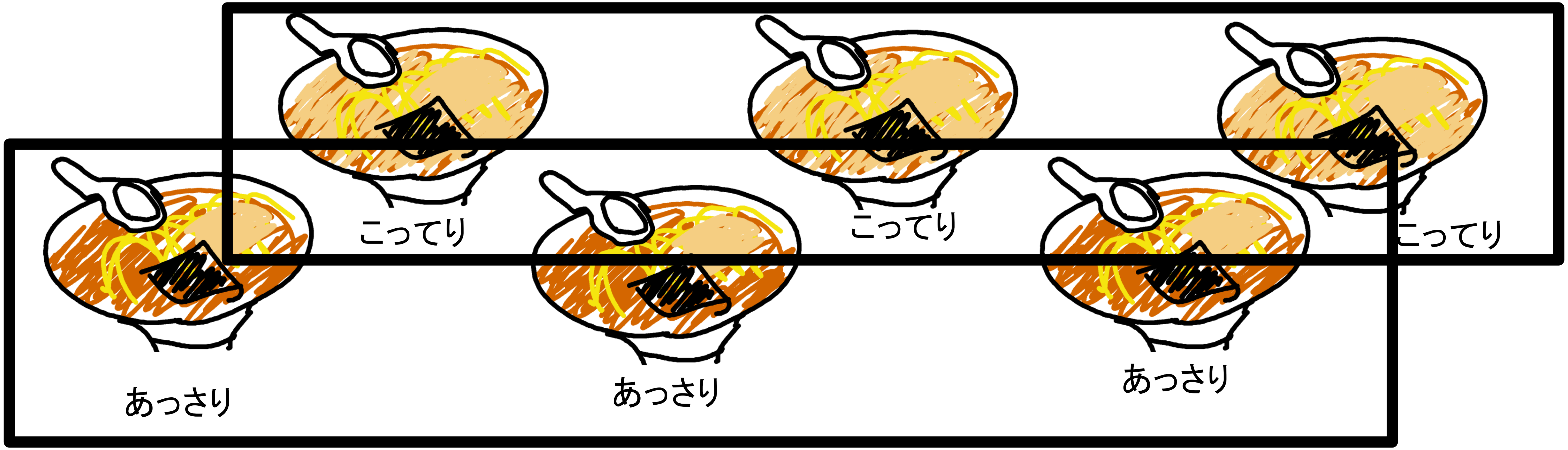
こってりグループとあっさりグループに分ける



小課題3 (累計100点)

$N \leq 400$

こってりグループとあっさりグループに分ける



小課題3 (累計100点)

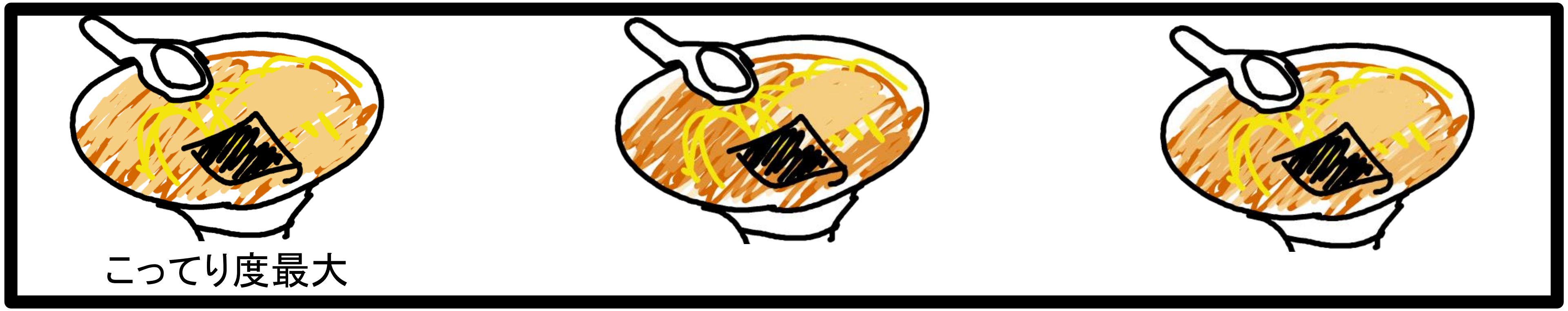
$N \leq 400$

こってりグループ→こってり度最大を見つける

小課題3 (累計100点)

$N \leq 400$

こってりグループ→こってり度最大を見つける (ここで$N/2$回)



小課題3 (累計100点)

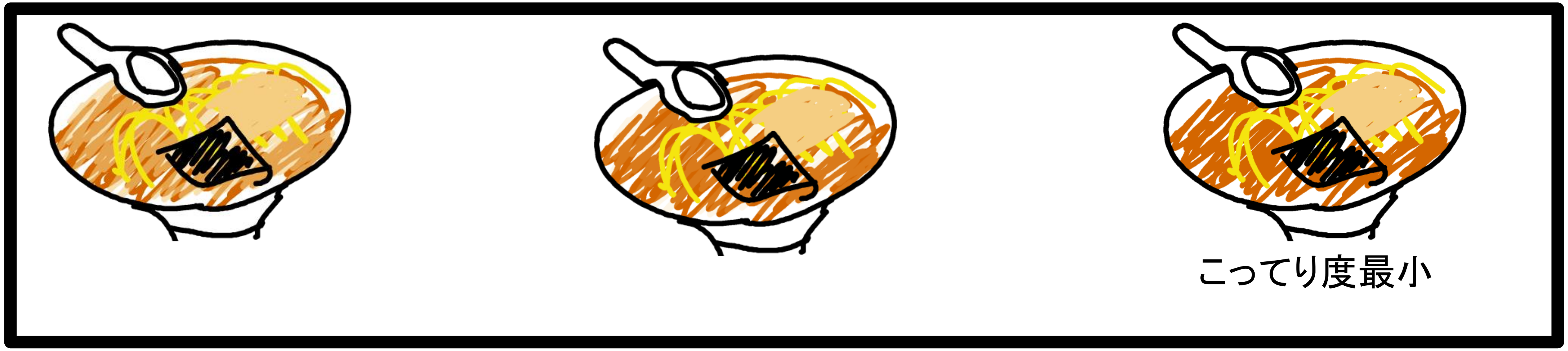
$N \leq 400$

あっさりグループ→

小課題3 (累計100点)

$N \leq 400$

あっさりグループ→こってり度最小を見つける(ここで$N/2$回)



小課題3 (累計100点)

$N \leq 400$

小課題3 (累計100点)

$N \leq 400$

最初の分類に $\left\lfloor \frac{N}{2} \right\rfloor$ 回

小課題3 (累計100点)

$N \leq 400$

最初の分類に $\left\lfloor \frac{N}{2} \right\rfloor$ 回

こってり度最大を決めるのに $\left\lceil \frac{N}{2} \right\rceil - 1$ 回

小課題3 (累計100点)

$N \leq 400$

最初の分類に $\left\lfloor \frac{N}{2} \right\rfloor$ 回

こってり度最大を決めるのに $\left\lceil \frac{N}{2} \right\rceil - 1$ 回、こってり度最小を決めるのに $\left\lceil \frac{N}{2} \right\rceil - 1$ 回

小課題3 (累計100点)

$N \leq 400$

最初の分類に $\left\lfloor \frac{N}{2} \right\rfloor$ 回

こってり度最大を決めるのに $\left\lceil \frac{N}{2} \right\rceil - 1$ 回、こってり度最小を決めるのに $\left\lceil \frac{N}{2} \right\rceil - 1$ 回

合計 $\left\lceil \frac{3N}{2} \right\rceil - 2 \leq 598$ 回の比較でできる

コーナーケース

コーナーケース

小課題3を実装すると、はじめに Compare(0, 1) を呼び出すコードになることがある

コーナーケース

小課題3を実装すると、はじめに Compare(0, 1) を呼び出すコードになることがある

→ $N = 1$ に注意

下界の保証

比較回数は $\left\lceil \frac{3N}{2} \right\rceil - 2$ 回以上必要

下界の保証

比較回数は $\left\lceil\frac{3N}{2}\right\rceil - 2$ 回以上必要

「最大の可能性が残っているラーメン屋」の個数をKとおく

「最小の可能性が残っているラーメン屋」の個数をLとおく

下界の保証

比較回数は $\left\lceil\frac{3N}{2}\right\rceil - 2$ 回以上必要

「最大の可能性が残っているラーメン屋」の個数をKとおく

「最小の可能性が残っているラーメン屋」の個数をLとおく

「最大の可能性も最小の可能性も残っているラーメン屋」の個数をMとおく

下界の保証

比較回数は $\left\lceil \frac{3N}{2} \right\rceil - 2$ 回以上必要

「最大の可能性が残っているラーメン屋」の個数をKとおく

「最小の可能性が残っているラーメン屋」の個数をLとおく

「最大の可能性も最小の可能性も残っているラーメン屋」の個数をMとおく

$V = K + L - \frac{M}{2}$ とおく。

下界の保証

補題1: V の初期値は $\frac{3N}{2}$ である。

補題2: $N > 1$ のとき、最大と最小が決まった時点での V の値は 2 である。

補題2’: $N = 1$ のとき、最大と最小が決まった時点での V の値は $\frac{3}{2}$ である。

下界の保証

補題1: V の初期値は $\frac{3N}{2}$ である。

補題2: $N > 1$ のとき、最大と最小が決まった時点での V の値は 2 である。

補題2’: $N = 1$ のとき、最大と最小が決まった時点での V の値は $\frac{3}{2}$ である。

補題3: 初期状態から終了状態までの間に、V の値は $\frac{3\mathrm{N}}{2} - 2$ 以上減少する。

下界の保証

補題4: Compare(X, Y) をどのように呼び出しても、Vの減少が 1 以下であるような結果が返ってくる可能性がある。

下界の保証

補題4は場合分けで証明できる(長いので適当に)

下界の保証

X, Yの両方が、「最小かもしれないし、最大かもしれない」とき:

下界の保証

X, Yの両方が、「最小かもしれないし、最大かもしれない」とき:

XとYの大小関係が決まると、K と L は 1 ずつ減少し、M は 2 減少する。

したがって、V は 1 減少する。

下界の保証

Xは、「最小かもしれないし、最大かもしれない」が、Yは「最小**<u>ではない</u>**が、最大かもしれない」とき:

下界の保証

Xは、「最小かもしれないし、最大かもしれない」が、Yは「最小**<u>ではない</u>**が、最大かもしれない」とき:

X < Y となる可能性がある。このとき、K は 1 減少するが、L は減少しない。

M は 1 減少する。したがって、V は $\frac{1}{2}$ 減少する。

下界の保証

Xは、「最小かもしれないし、最大かもしれない」が、Yは「最小**<u>ではない</u>**が、最大かもしれない」とき:

X < Y となる可能性がある。このとき、K は 1 減少するが、L は減少しない。

M は 1 減少する。したがって、V は $\frac{1}{2}$ 減少する。

Xは、「最小かもしれないし、最大かもしれない」が、Yは「最小でも最大でもない」とき………も同様。

下界の保証

Xは、「最小かもしれないし、最大かもしれない」が、Yは「最小**<u>ではない</u>**が、最大かもしれない」とき:

X < Y となる可能性がある。このとき、K は 1 減少するが、L は減少しない。

M は 1 減少する。したがって、V は $\frac{1}{2}$ 減少する。

Xは、「最小かもしれないし、最大かもしれない」が、Yは「最小でも最大でもない」とき.........も同様。

XとY, 最大と最小を逆にしても同様。

下界の保証

XもYも「最大かもしれないが、最小ではない」とき:

下界の保証

XもYも｢最大かもしれないが、最小ではない｣とき:

XとYの大小関係が決まると、K は 1 減少する。

したがって、V は 1 減少する。

下界の保証

Xも Yも「最大かもしれないが、最小ではない」とき:

XとYの大小関係が決まると、K は 1 減少する。

したがって、V は 1 減少する。

最大と最小を逆にしても同様。

下界の保証

Xは「最小かもしれないが、最大ではない」のに対し、Yは「最大かもしれないが、最小ではない」とき:

下界の保証

Xは「最小かもしれないが、最大ではない」のに対し、Yは「最大かもしれないが、最小ではない」とき:

X < Y となる可能性がある。このとき、V は変化しない。

下界の保証

Xは「最小かもしれないが、最大ではない」のに対し、Yは「最大かもしれないが、最小ではない」とき:

X < Y となる可能性がある。このとき、V は変化しない。

最大と最小を逆にしても同様。

下界の保証

Xは「最小かもしれないが、最大ではない」のに対し、Yは「最大でも最小でもない」とき:

下界の保証

Xは「最小かもしれないが、最大ではない」のに対し、Yは「最大でも最小でもない」とき:

X < Y となる可能性がある。このとき、V は変化しない。

下界の保証

Xは「最小かもしれないが、最大ではない」のに対し、Yは「最大でも最小でもない」とき:

X < Y となる可能性がある。このとき、V は変化しない。

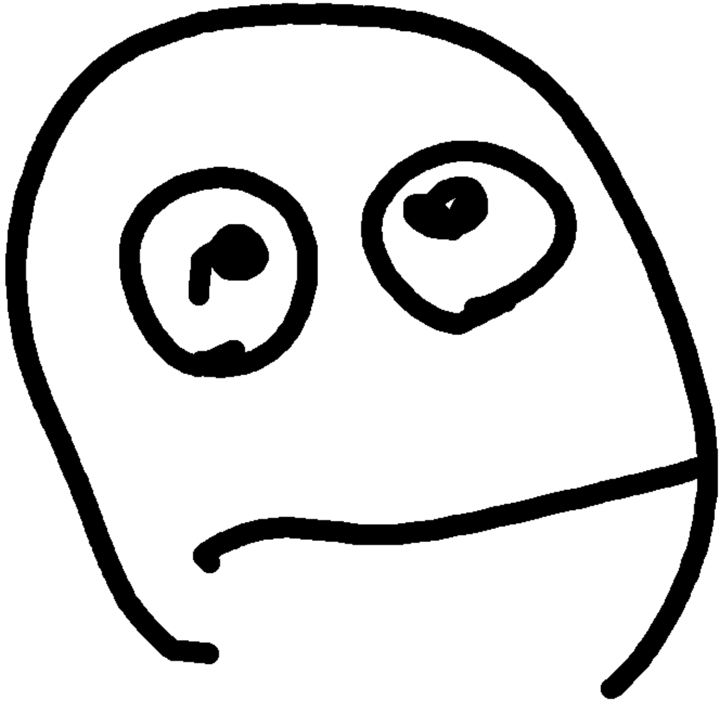
XとY, 最大と最小を逆にしても同様。

下界の保証

XもYも「最大でも最小でもない」とき:

下界の保証

XもYも「最大でも最小でもない」とき:



下界の保証

以上の考察より、V の減少量が 1 に満たないようなCompare呼び出しをたくさん行うプログラムはダメだということもわかる

別解

別解: 小課題1

別解: 小課題1

std::sortの比較関数としてCompareを用いる

別解: 小課題1

std::sortの比較関数としてCompareを用いる

```
bool cmp(int a, int b) {
  return a != b && Compare(a, b) < 0;
}
```

別解: 小課題1

std::sortの比較関数としてCompareを用いる

→std::sortの比較回数は $O(n\log n)$ なので普通はOK

別解: 小課題1

std::sortの比較関数としてCompareを用いる

→std::sortの比較回数は $O(n \log n)$ なので普通はOK

(オーダーの係数に関する保証はない / C++11以前は最悪計算量は保証されていない)

別解: 小課題2

別解: 小課題2

std::min_element / std::max_element の比較関数としてCompareを用いる

別解: 小課題2

std::min_element / std::max_element の比較関数としてCompareを用いる

→比較回数はそれぞれ $N-1$ であることが保証されているのでOK

別解: 小課題3

別解: 小課題3

std::minmax_element の比較関数としてCompareを用いる

別解: 小課題3

std::minmax_element の比較関数としてCompareを用いる

→比較回数は $\min\left(\left\lfloor\frac{3(N-1)}{2}\right\rfloor, 0\right)$ であることが保証されているのでOK

- これはさっきの式と同じ

別解: 小課題3

std::minmax_element の比較関数としてCompareを用いる

→比較回数は $\min\left(\left\lfloor\frac{3(N-1)}{2}\right\rfloor, 0\right)$ であることが保証されているのでOK

- これはさっきの式と同じ

C++11以降でしか使えない

別解: 小課題3

std::minmax_element の比較関数としてCompareを用いる

→比較回数は $\min\left(\left\lfloor\frac{3(N-1)}{2}\right\rfloor, 0\right)$ であることが保証されているのでOK

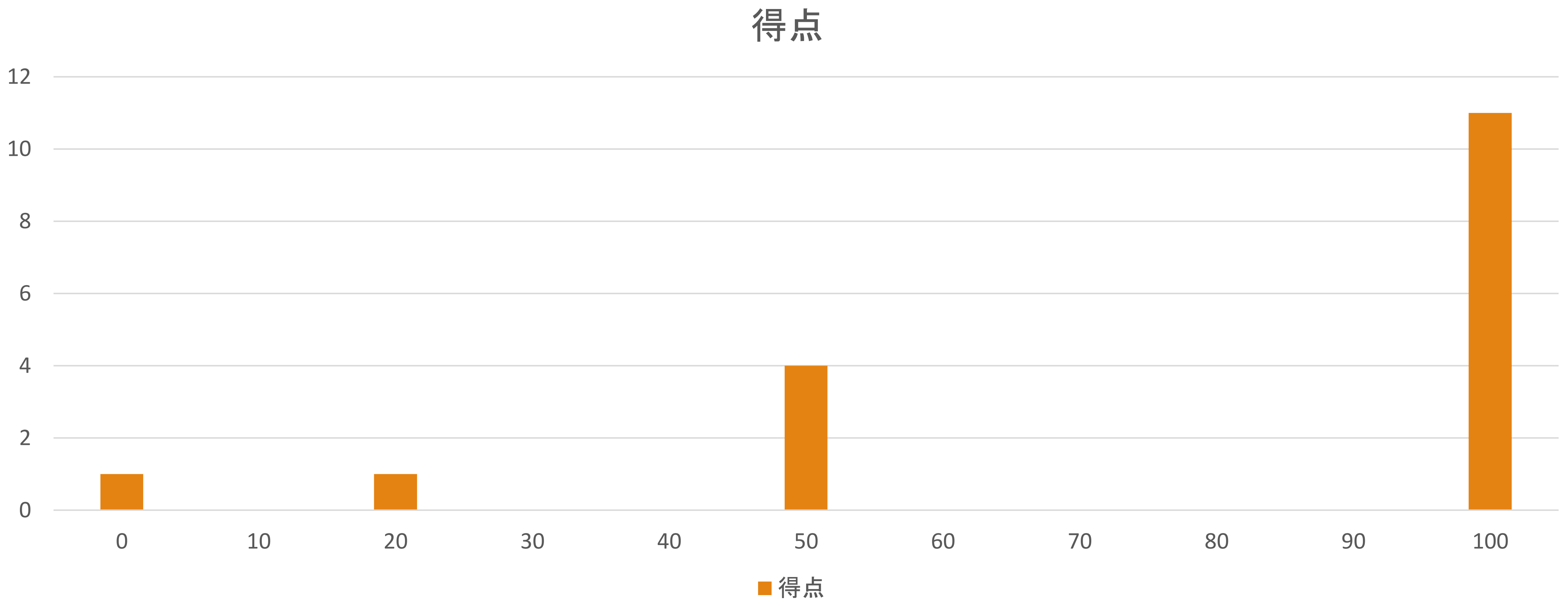
- これは $N \geq 1$ ではさっきの式と同じ

C++11以降でしか使えない

- 今回の合宿では使えなかったことになる

分布

分布



おわり

1. 問題概要
2. 小課題1 (20点)
3. 小課題2 (累計50点)
4. 小課題3 (累計100点)
5. コーナーケース
6. 下界の証明
7. 別解 (小課題1･小課題2･小課題3)
8. 分布