

データ構造と アルゴリズム

2019年4月－7月

教員名：松井くにお

研究室：67・106(やつかほ)内線：75-2206

E-mail: kmatsui@neptune.kanazawa-it.ac.jp

この授業について

■ 教室と時間

- 2EP2クラス: 水曜1限 @ 23.323
- 2EP3クラス: 水曜2限 @ 23.323

■ オフィスアワー

- 火曜5限、場所は 21.405室
- できるだけ事前にメールでアポをとって下さい。
- これ以外の時間帯: 必ずメールでアポをとって下さい。

■ 教科書

- アルゴリズムとデータ構造 第2版[森北出版]

学習計画

データ構造とアルゴリズム（松井クラス）講義日程と内容（予定）

2EP2、2EP3 @23. 323

第3版 5月31日

日付		曜日	講義回数	学習内容
4月	10日	(水)	第1回	授業のガイダンス, アルゴリズムの基礎, 時間計算量
	17日	(水)	第2回	基本データ構造 (配列とリスト, スタックとキュー)
	24日	(水)	第3回	アルゴリズムにおける基本概念 (木, 再帰)
5月	8日	(水)	第4回	データの探索
	15日	(水)	第5回	ソートアルゴリズム1 (選択ソート, 挿入ソート)
	22日	(水)	第6回	ソートアルゴリズム2 (クイックソート, マージソート)
	29日	(水)		休講
	31日	(金) 4限	第7回	レポート課題1-6の解説 2クラス合同小テスト (教室は23・221)
6月	5日	(水)	第8回	小テストの解答・解説, ソートアルゴリズムのまとめ
	12日	(水)	第9回	グラフアルゴリズム (グラフとそのデータ構造)
	19日	(水) 2EP2穴水	第10回	総合演習 (2EP2) / 重み付きグラフ, 最短経路探索 (2EP3)
	26日	(水) 2EP3穴水	第11回	重み付きグラフ, 最短経路探索 (2EP2) / 総合演習 (2EP3)
	28日	(金) 4限	第11回	重み付きグラフ, 最短経路探索 (2EP2) @23. 320 / 総合演習 (2EP3)
7月	3日	(水)	第12回	アルゴリズム設計手法, 総復習
	10日	(水)	第13回	達成度確認試験の過去問
	19日	(金) 4限	第14回	2クラス合同達成度確認試験 (教室は23・221) アルゴリズムの限界
	24日	(水)		休講
	31日	(水)	第15回	試験の解答, 総復習, 自己点検

前回のおさらい

■ クイックソート

- 分割: 基準値に対して大小で判断
- 最小単位になるまで再帰的に繰り返す

■ マージソート

- 分割統治法
- 分割、統治、組み合わせ
- 再帰的アルゴリズム

※マージ: 併合

今回の内容

■ 小テストの解答・解説

- 別紙

■ 第1週レポート課題

- スライド解答の間違い(スライドが間違いで採点はOK)

■ ソートアルゴリズムのまとめ

- 選択ソート
- 挿入ソート
- ヒープソート
- クイックソート
- マージソート

3. 計算時間が以下の各式であったとき, その時間計算量をオーダー記法で示せ. なお, n は問題サイズであり, 自然数とする.

(1) $4n^2 + 1024n + 65536$

$O(n^2)$

(2) $n + \log_2 n$

$O(n)$

(3) $n + e^n$ (e は自然対数の底で, 2.718...)

$O(e^n)$

(4) $1 + 2 + \dots + n$

数列の和であるため多項式に書き換えが可能

$$\frac{1}{2} \times n(n+1) = \frac{1}{2}n^2 + \frac{1}{2}n$$

$O(n^2)$

選択ソート

■ アルゴリズム

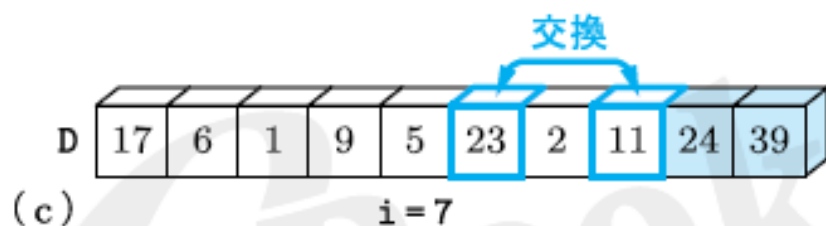
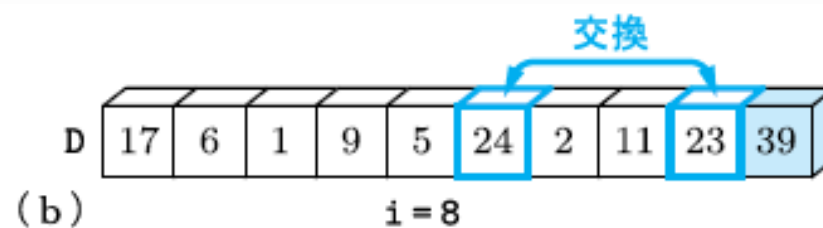
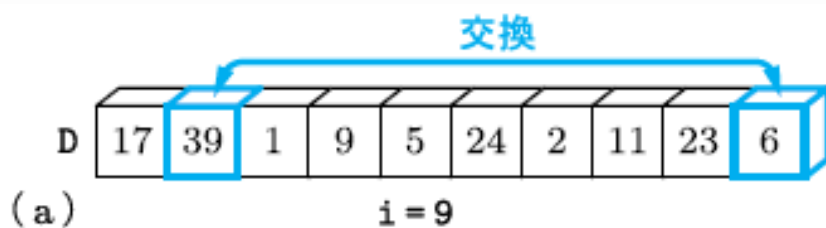
- 前提: n 個のデータが入力されている
- ① 入力データから最大値を見つける
- ② 見つけた最大値のデータを対象から外す
- ③ ①②の操作を $n-1$ 回繰り返す

■ 時間計算量

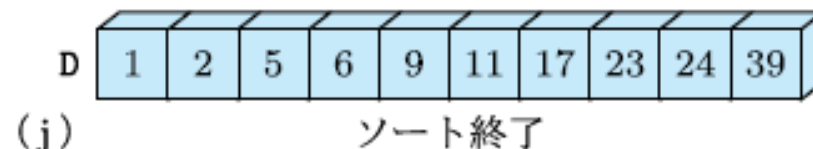
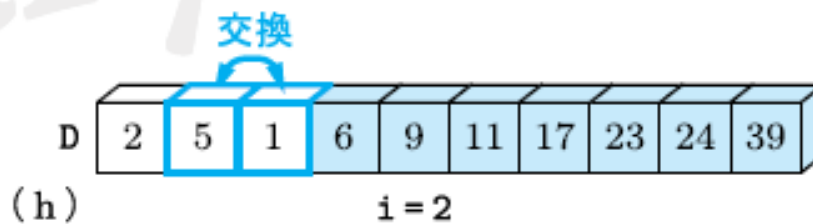
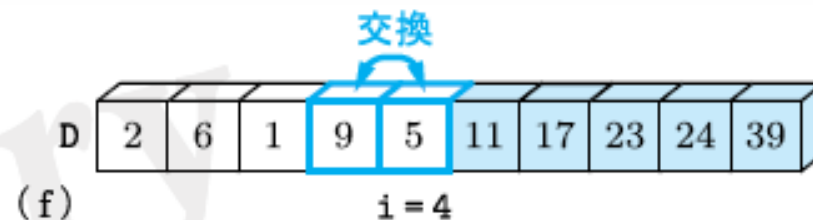
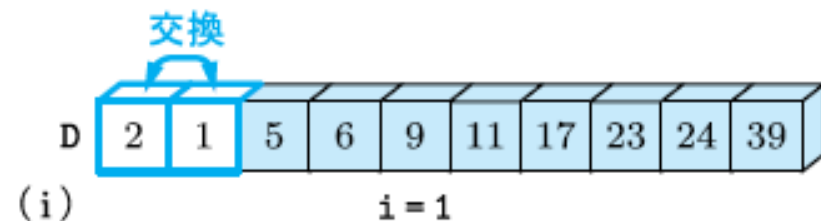
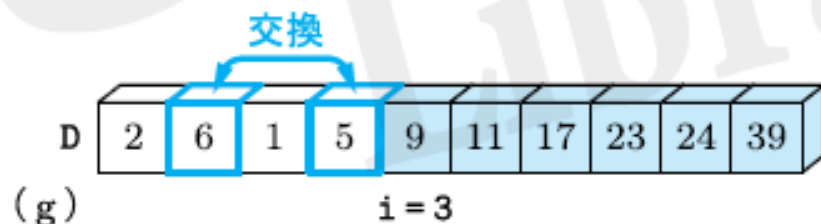
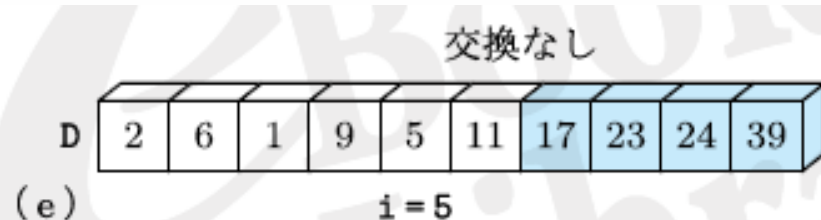
$$\begin{aligned}\sum_{i=1}^{n-1} i \times O(1) &= O(1) \times \frac{n(n-1)}{2} \\ &= O(n^2)\end{aligned}$$

選択ソート(具体例)

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}



選択ソート(具体例)



挿入ソート

■ アルゴリズム

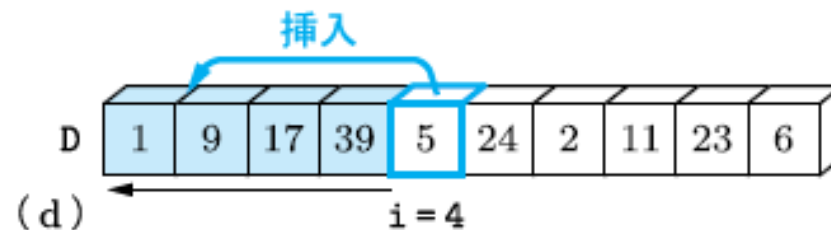
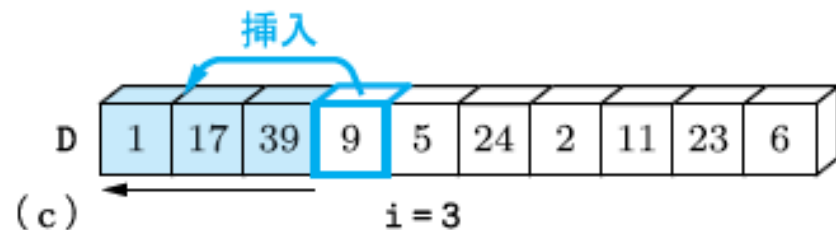
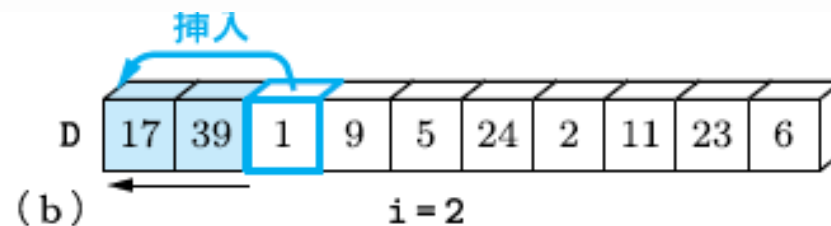
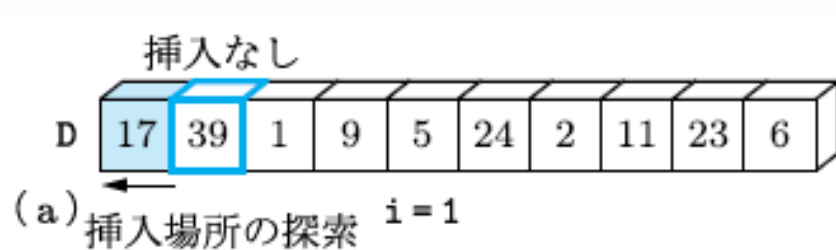
- 前提: n 個のデータをソートの対象とする
- ① 最初のデータを左端に置く
- ② 次のデータは元のデータに合わせて昇順に並べる
- ③ ①②の操作を $n-1$ 回繰り返す

■ 時間計算量

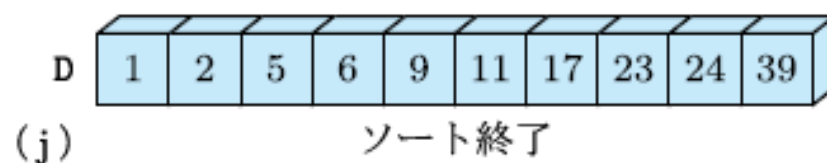
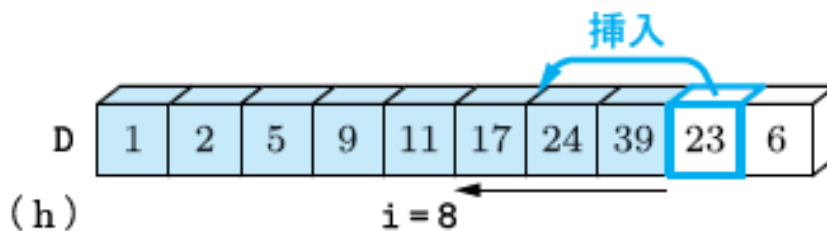
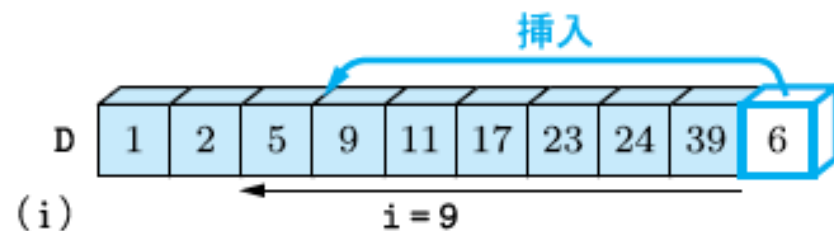
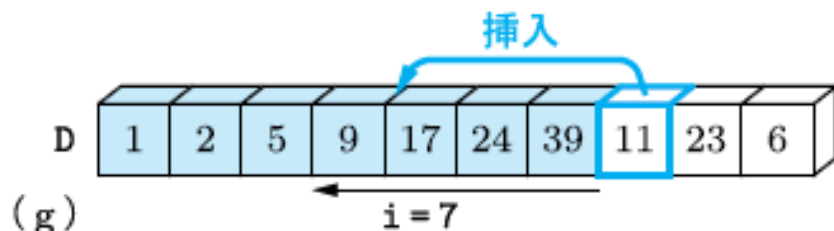
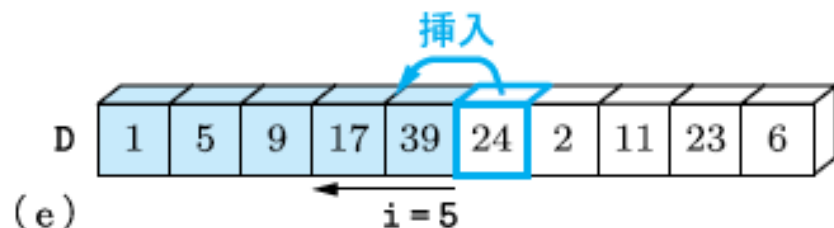
- 最良時間計算量: ソート済の場合 $O(n)$
- 最悪時間計算量: 選択ソートと同じ $O(n^2)$
- 平均時間計算量: $O(n^2)$

挿入ソート(具体例)

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}



挿入ソート(具体例)



ヒープ

■ ヒープの定義

- 2分木で必ず左詰め
 - 性質1 2分木の最大のレベルを lm とすると, $0 \leq k \leq lm-1$ を満たす各レベル k には 2^k 個の節点が存在し, レベル lm に存在する葉はそのレベルに左詰めされている.
- 親は子よりも必ず大きい
 - 性質2 各節点に保存されるデータは, その子に保存されるデータより大きい

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

17
0

ヒープ

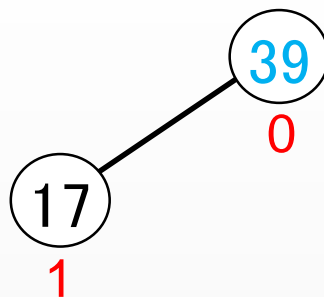
ヒープを
表す配列

17									
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



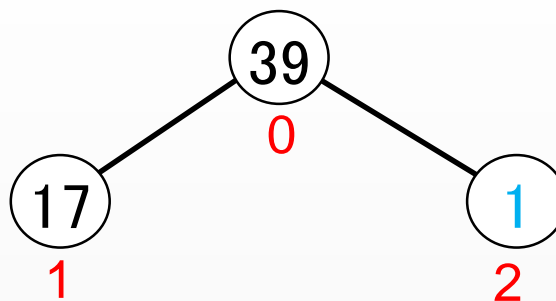
ヒープを
表す配列

39	17								
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



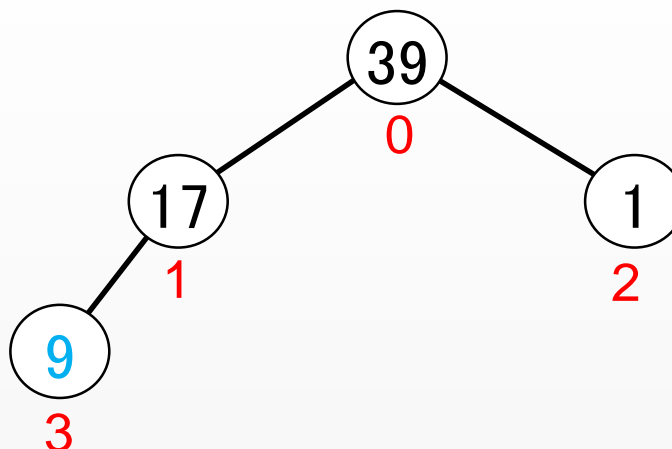
ヒープを
表す配列

39	17	1							
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



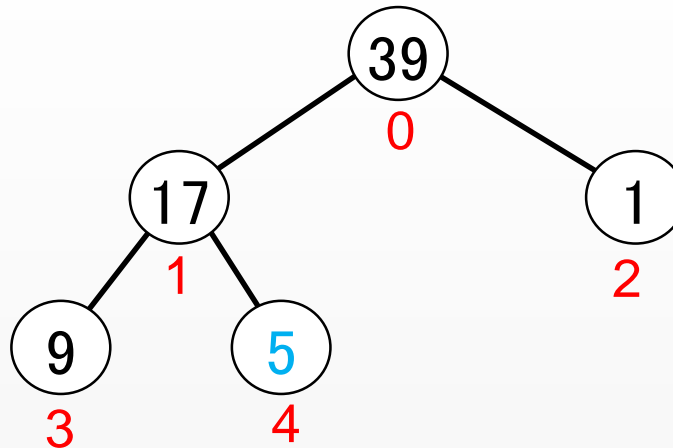
ヒープを
表す配列

39	17	1	9						
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



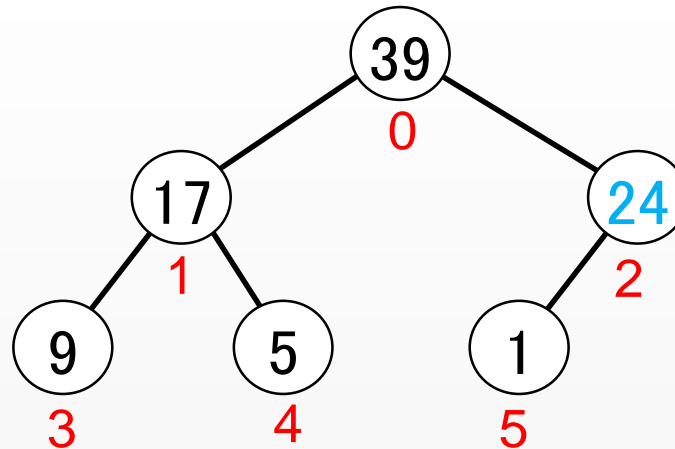
ヒープを
表す配列

39	17	1	9	5					
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



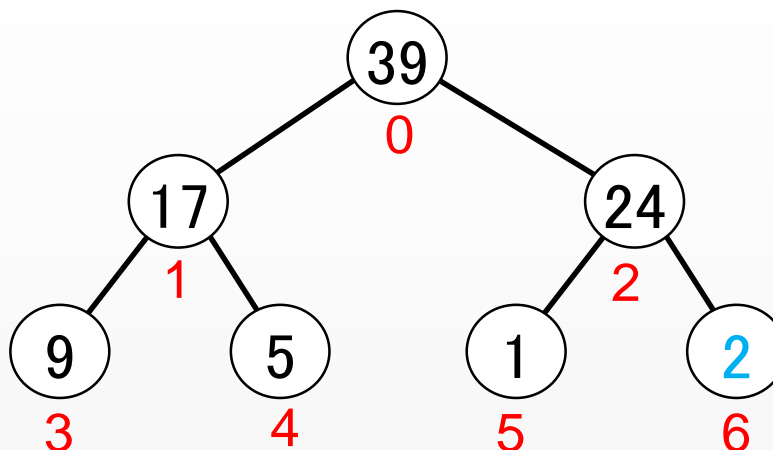
ヒープを
表す配列

39	17	24	9	5	1				
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



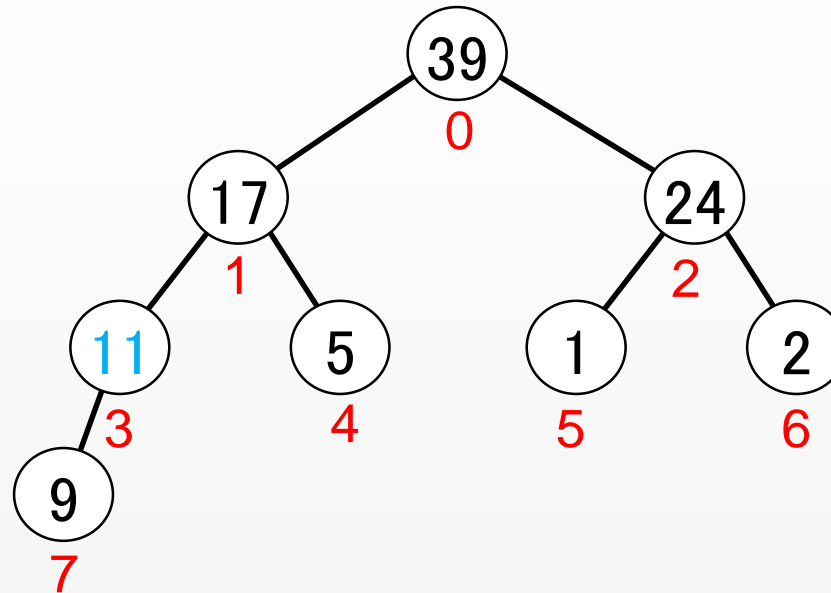
ヒープを
表す配列

39	17	24	9	5	1	2			
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



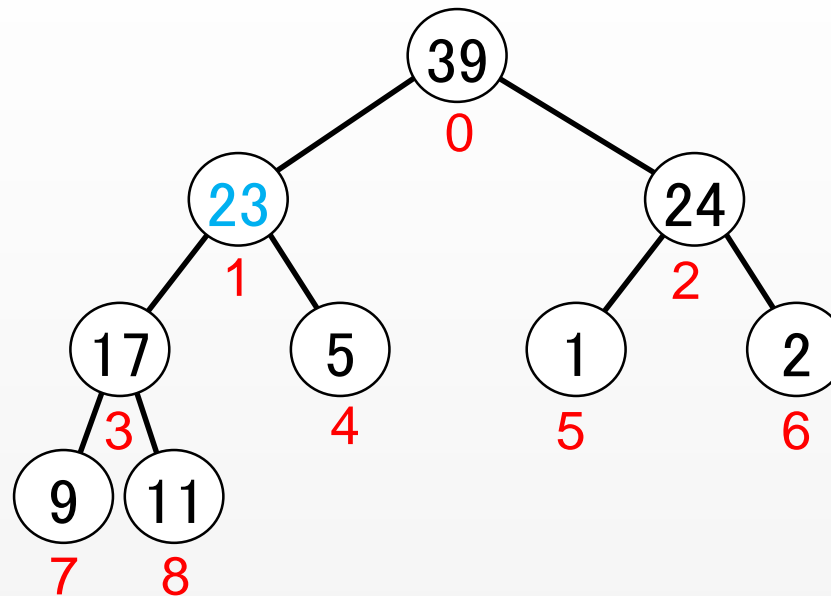
ヒープを
表す配列

39	17	24	11	5	1	2	9		
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



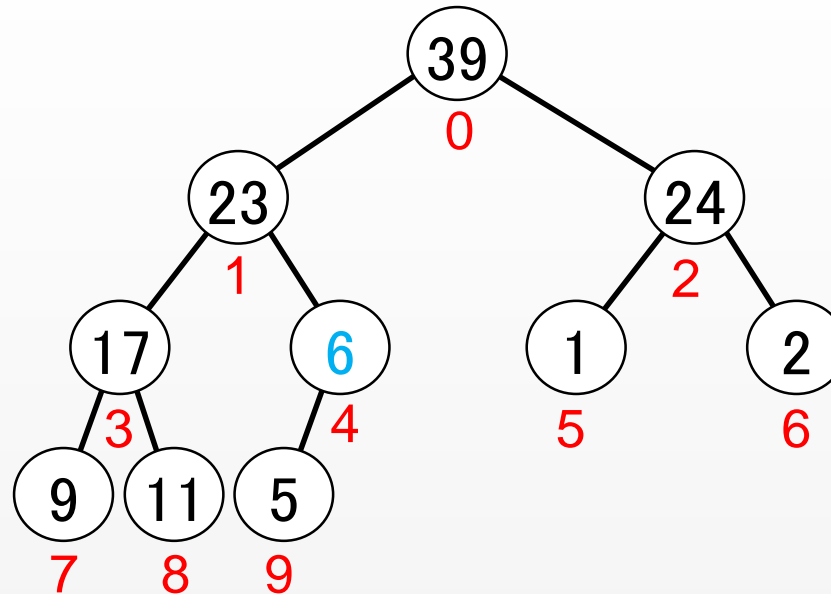
ヒープを
表す配列

39	23	24	17	5	1	2	9	11	
0	1	2	3	4	5	6	7	8	9

ヒープへのデータ構造の作成

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



ヒープを
表す配列

39	23	24	17	6	1	2	9	11	5
0	1	2	3	4	5	6	7	8	9

ヒープへの追加

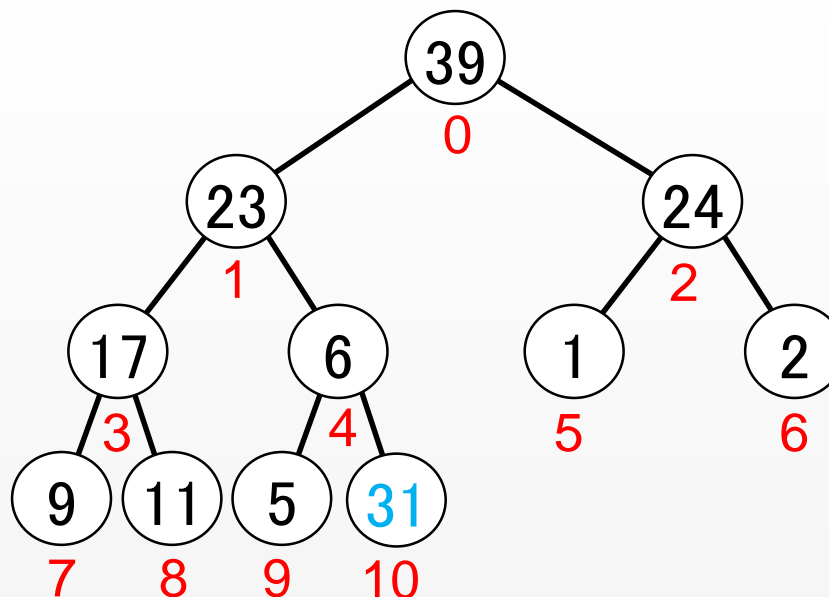
■ 方法

- 2分木の左詰めで追加
- 親と比べて親が小さければ位置を交換
- 親が大きければ終わり

ヒープへの追加

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6} 31

ヒープ



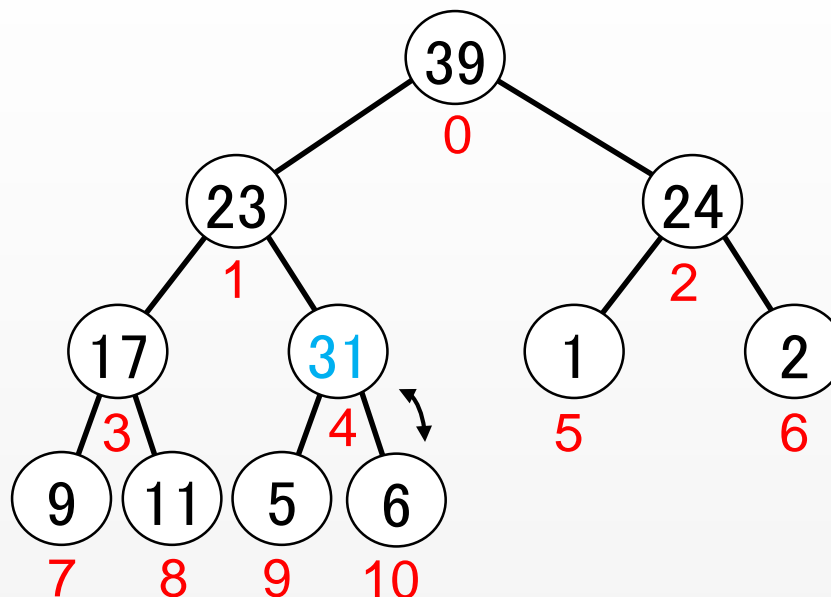
ヒープを
表す配列

39	23	24	17	6	1	2	9	11	5	31
0	1	2	3	4	5	6	7	8	9	10

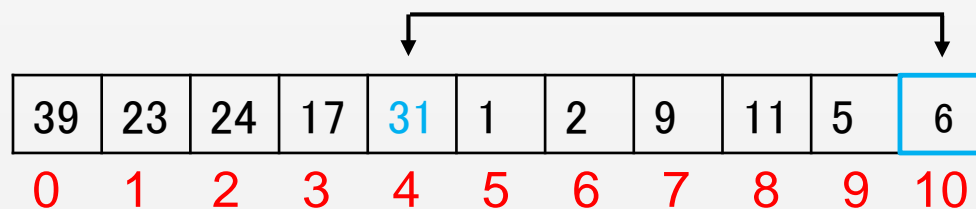
ヒープへの追加

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6} 31

ヒープ



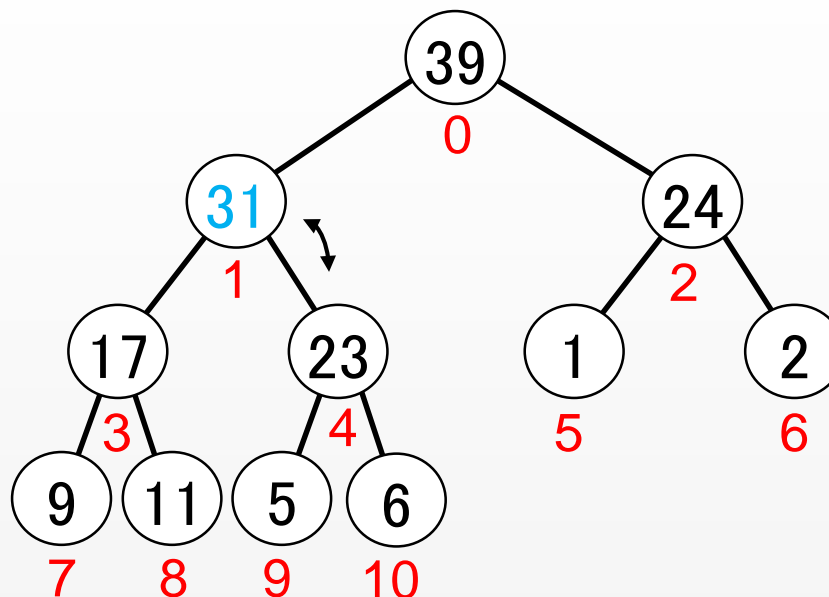
ヒープを
表す配列



ヒープへの追加

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6} 31

ヒープ



ヒープを
表す配列

39	31	24	17	23	1	2	9	11	5	6
0	1	2	3	4	5	6	7	8	9	10

ヒープから最大値の取り出し

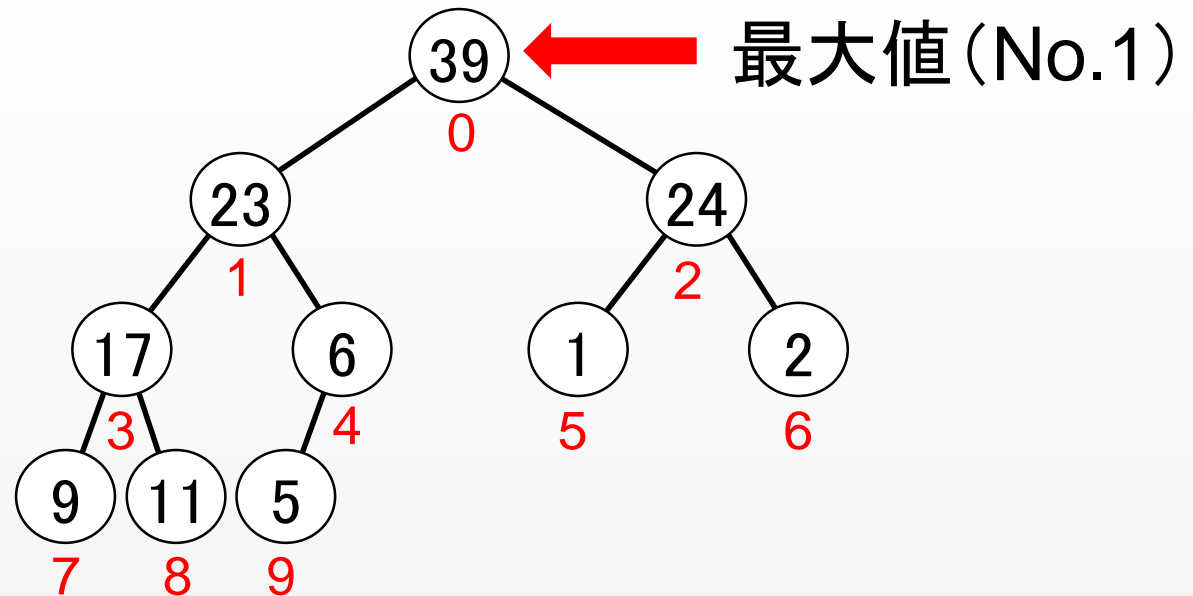
■ 方法

- 2分木の根の値(最大値)を取り出す
- 最後に追加したデータを根に移動する
- 根と根の子の大きい方を交換する
- 上記の操作を葉に向かって繰り返す(ヒープを保つ)


ヒープから最大値の取り出し

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



ヒープを
表す配列

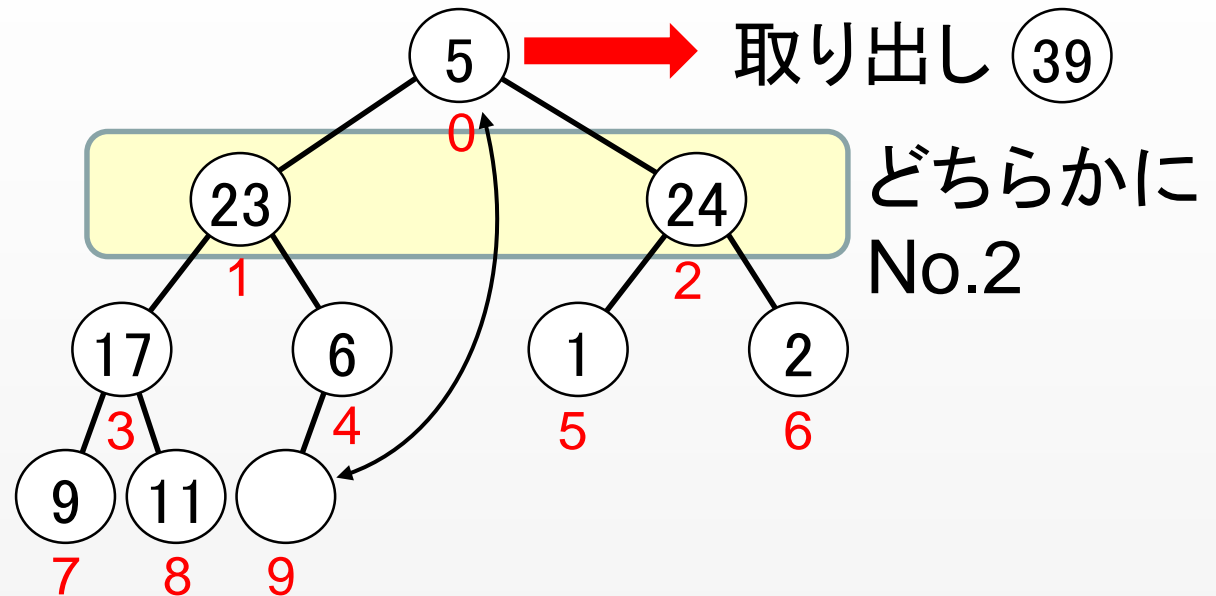


39	23	24	17	6	1	2	9	11	5
0	1	2	3	4	5	6	7	8	9

ヒープから最大値の取り出し

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



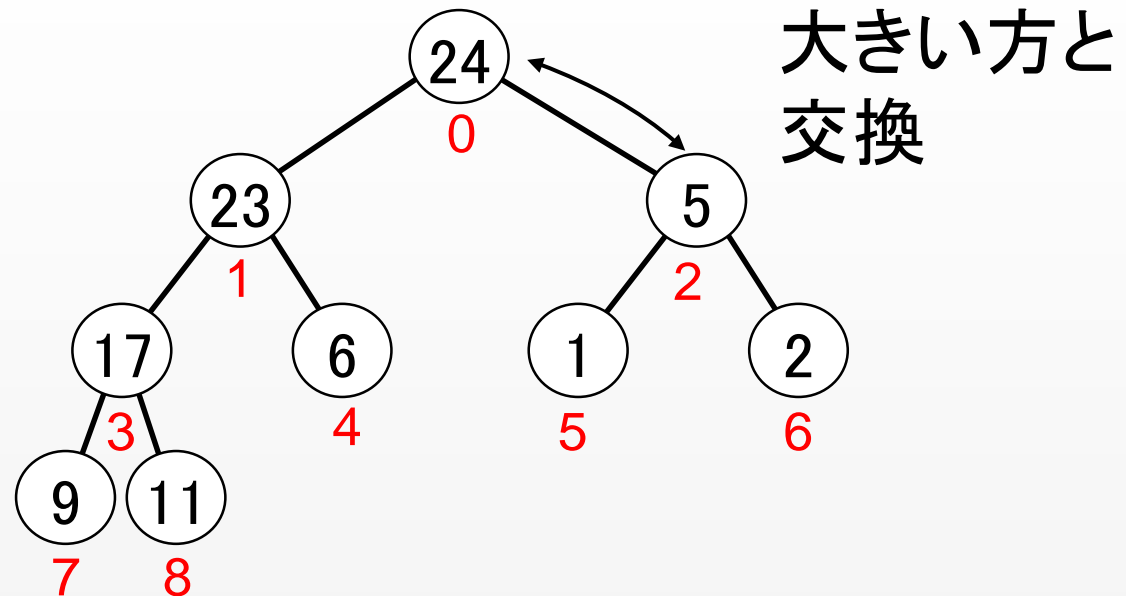
ヒープを
表す配列

5	23	24	17	6	1	2	9	11	
0	1	2	3	4	5	6	7	8	9

ヒープから最大値の取り出し

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

ヒープ



ヒープを
表す配列

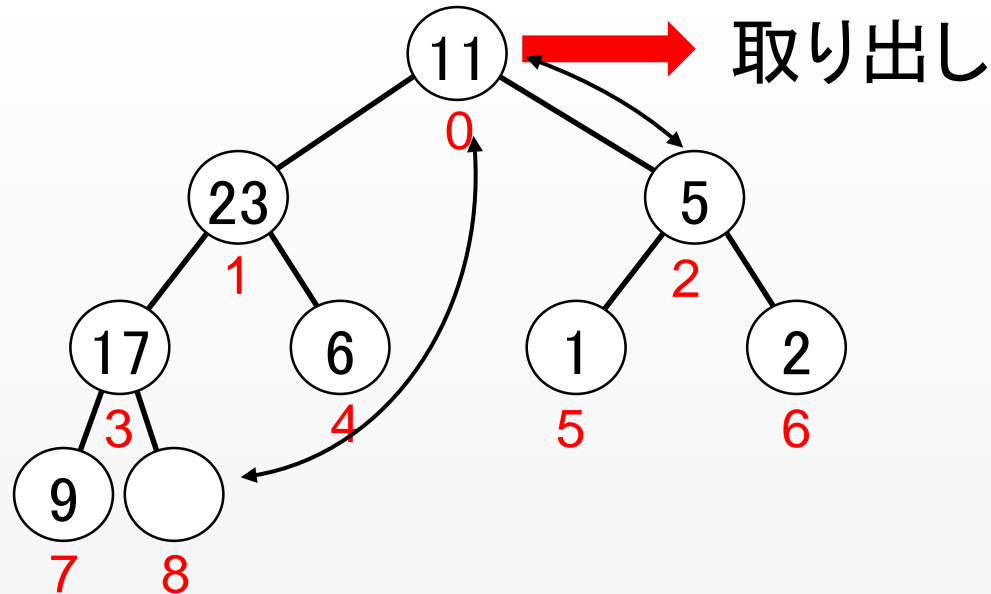
24	23	5	17	6	1	2	9	11	
0	1	2	3	4	5	6	7	8	9

ヒープから最大値の取り出し

■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}

39
24

ヒープ



ヒープを
表す配列

11	23	5	17	6	1	2	9		
0	1	2	3	4	5	6	7	8	9

ヒープソート

■ アルゴリズム

- 毎回最大値を取り出したものを順に並べる

■ 時間計算量

- 最悪時間計算量 $2 \times n \times \log n = O(n \log n)$

クイックソート

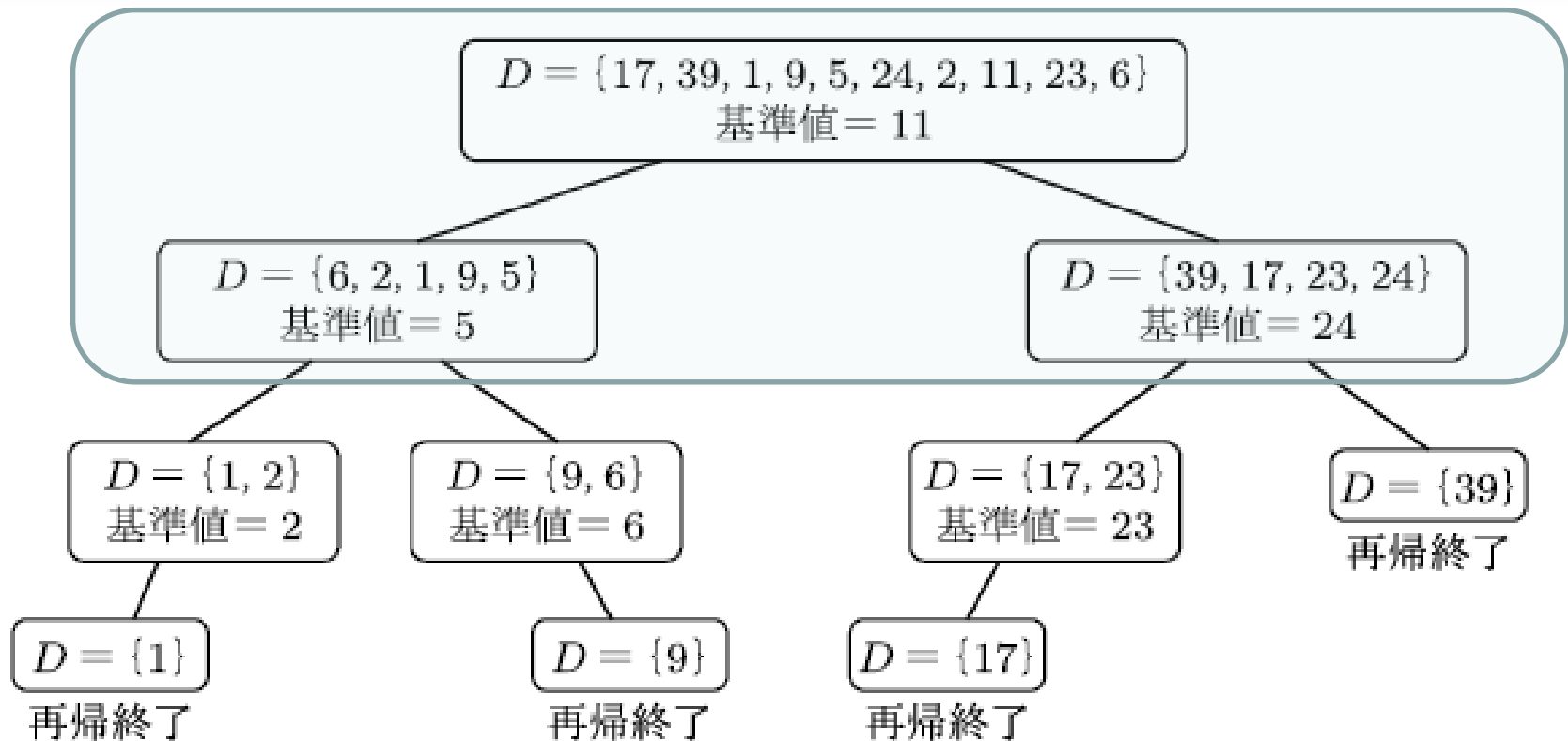
■ アルゴリズム

- ① 適当に基準値を選ぶ
- ② 基準値の大小によってグループに分割
- ③ グループごとに①②を繰り返す
- ④ 最小単位になったら終了

■ 時間計算量

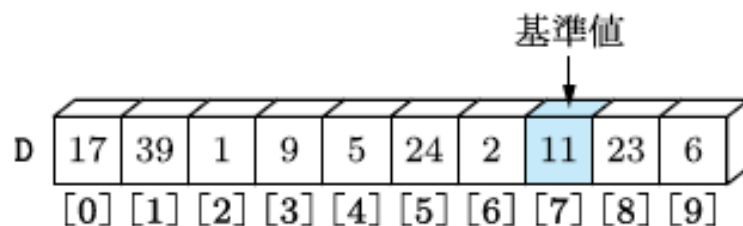
- $O(n \log n)$

クイックソート(考え方)

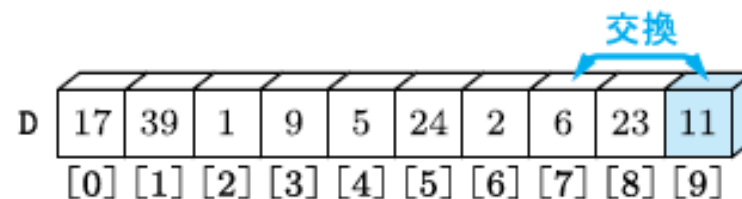


クイックソート(具体例)

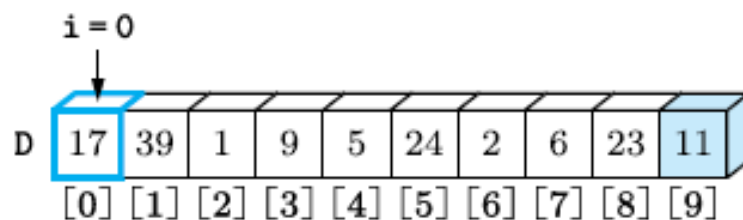
■ 入力: {17, 39, 1, 9, 5, 24, 2, 11, 23, 6}



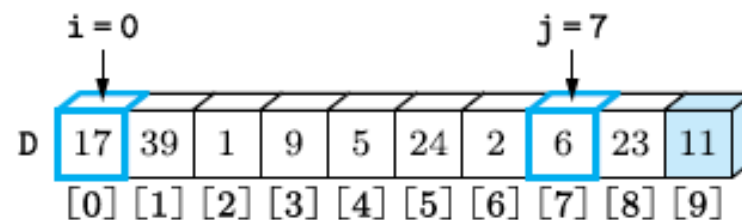
(a)



(b)



(c)



(d)



左端から基準値より大きい値を見つける

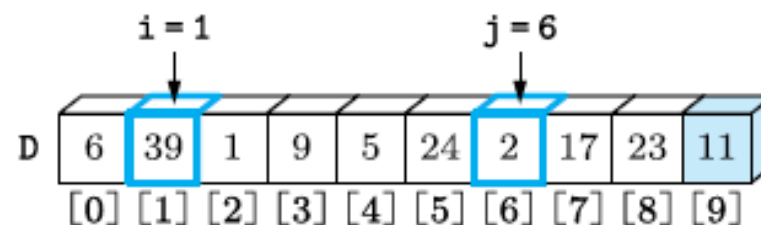


右端から基準値より小さい値を見つける

クイックソート(具体例)



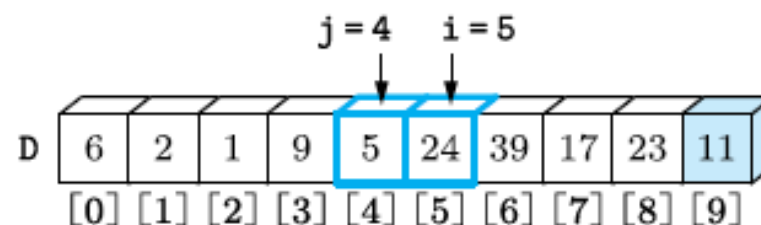
(e)



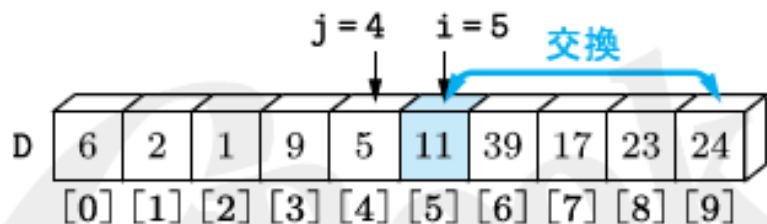
(f)



(g)



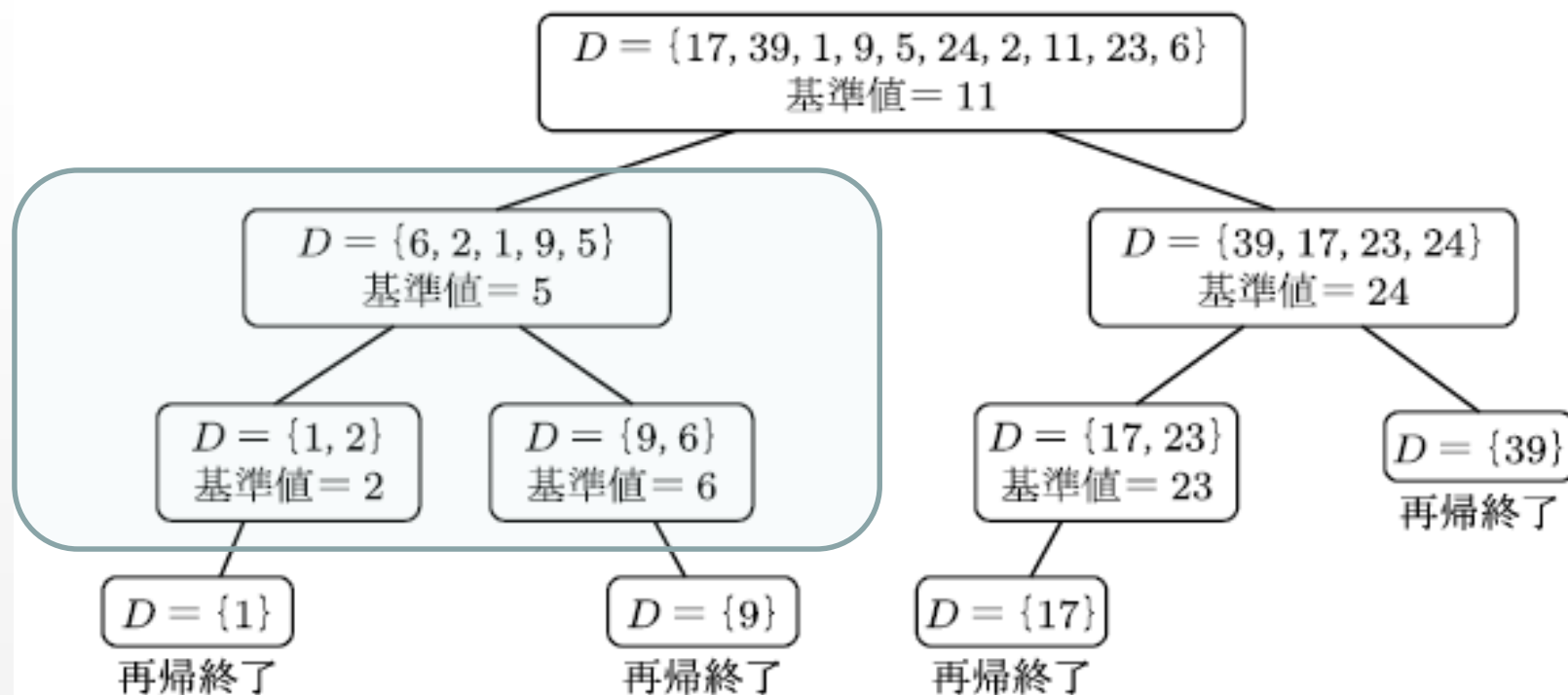
(h)



(i)

大小の交換終了後、基準値と大きい値を交換

クイックソート



クイックソート(具体例)

■ 入力: {6, 2, 1, 9, 5}

(a)

6	2	1	9	5
0	1	2	3	4

(b)

↓		↓		
6	2	1	9	5
0	1	2	3	4

(c)

1	2	6	9	5
0	1	2	3	4

(d)

	↓	↓		
1	2	6	9	5
0	1	2	3	4

(e)

1	2	5	9	6
0	1	2	3	4

マージソート

■ アルゴリズム

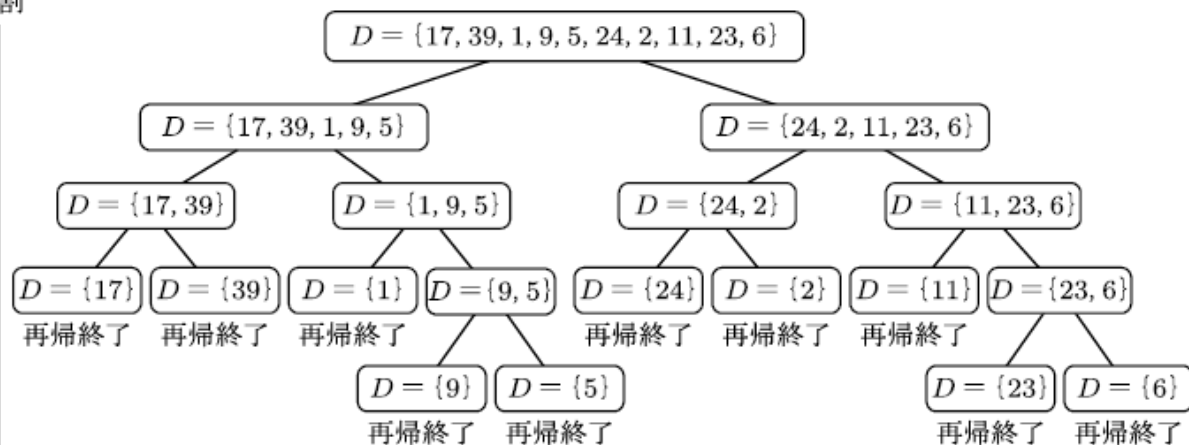
- ① データの集合を2つの集合に分割する
- ② ①を再帰的に繰り返し、最小単位になったら分割の終了
- ③ 分割した集合を深いレベルから順に再帰的にマージする

■ 時間計算量

- $O(n \log n)$

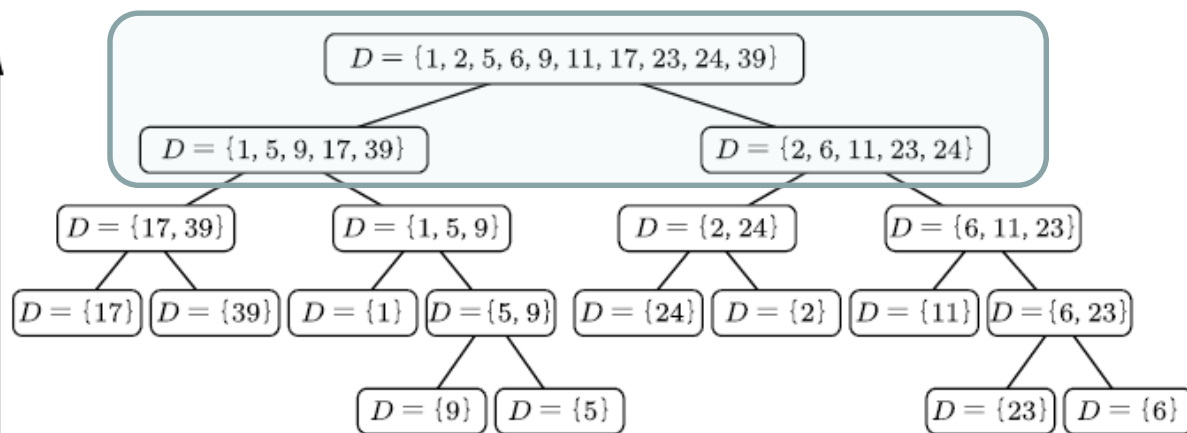
マージソート(考え方)

分割



(a)

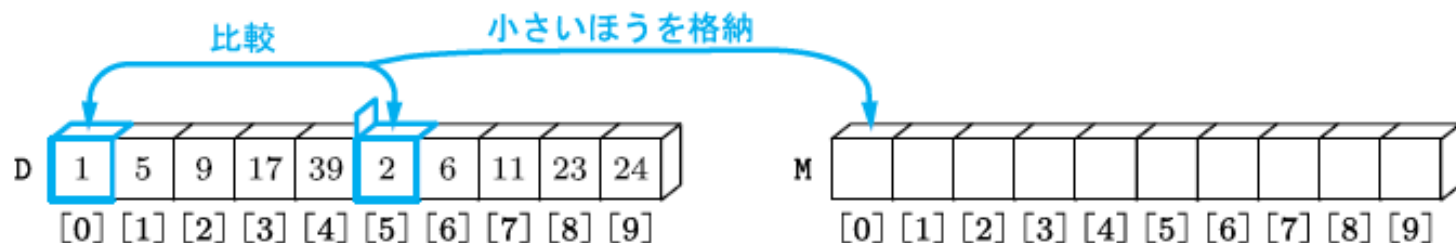
マージ



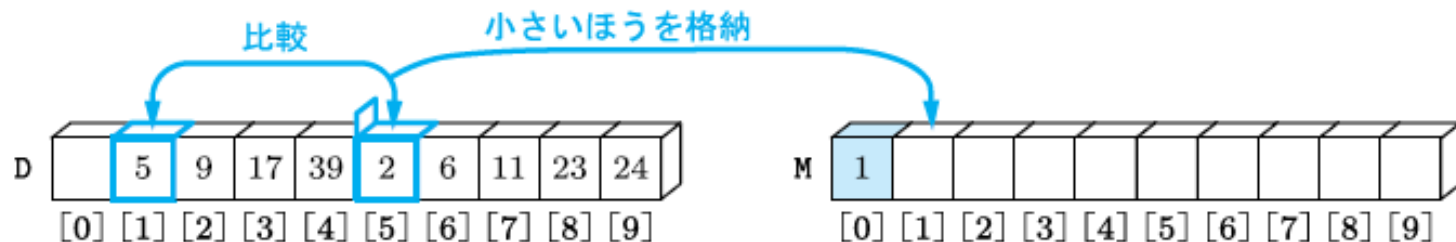
(b)

マージソート(具体例)

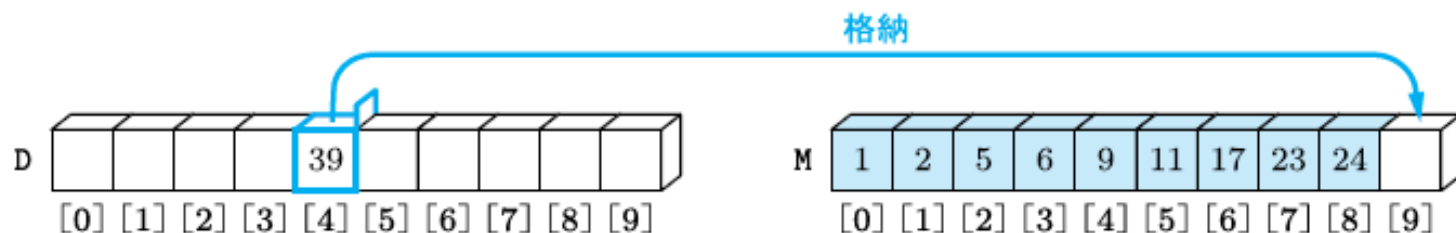
■ 入力: {1, 5, 9, 17, 39} {2, 6, 11, 23, 24}



(a)

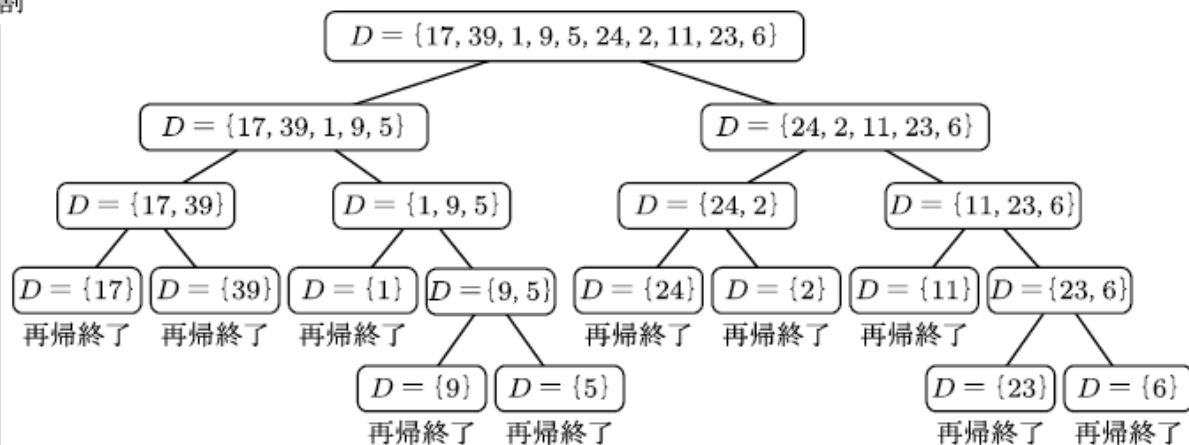


(b)



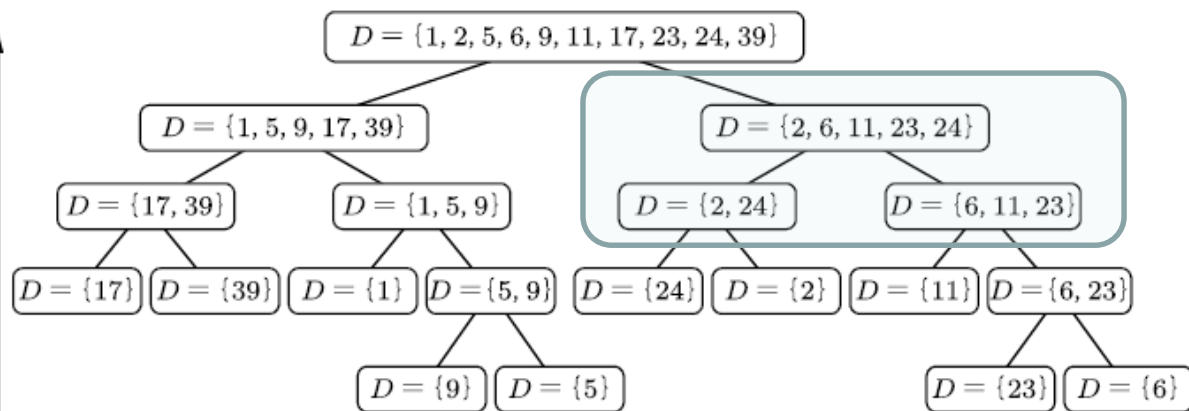
マージソート(考え方)

分割



(a)

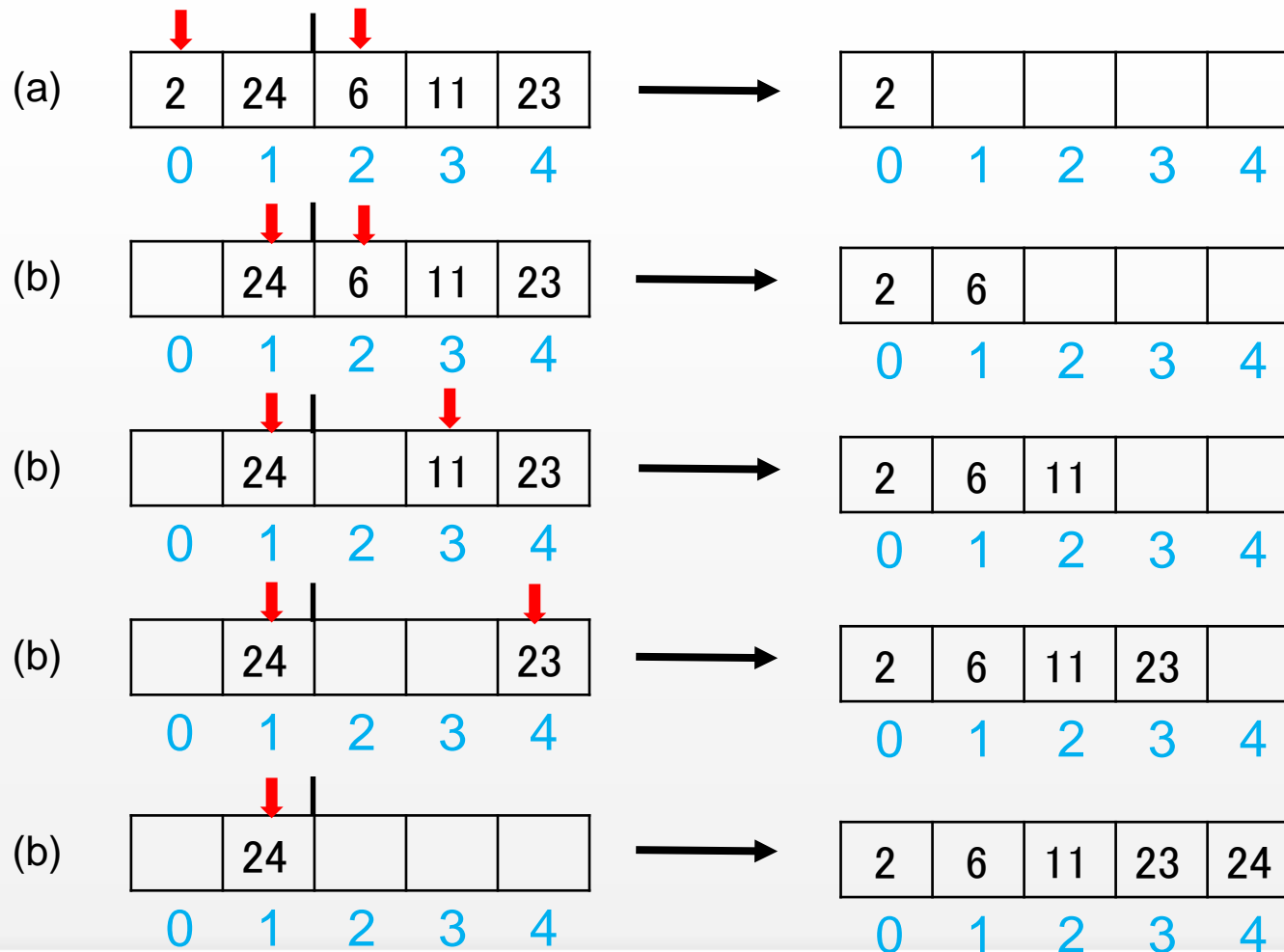
マージ



(b)

マージソート(具体例)

■ 入力: {2, 24} {6, 11, 23}



自然数の集合{35, 21, 4, 49, 55, 19, 12, 32}を入力とする。

1. 選択ソートを行なうステップ(1)～(7)で、配列に適切な数字を入れなさい。
途中でソートが終了した場合は、それまでで良い。

初期

35	21	4	49	55	19	12	32
0	1	2	3	4	5	6	7

(1)

0	1	2	3	4	5	6	7

(2)

0	1	2	3	4	5	6	7

(3)

0	1	2	3	4	5	6	7

(4)

0	1	2	3	4	5	6	7

(5)

0	1	2	3	4	5	6	7

(6)

0	1	2	3	4	5	6	7

(7)

4	12	19	21	32	35	49	55
0	1	2	3	4	5	6	7

2. 挿入ソートを行なうステップ(1)～(7)で、配列に適切な数字を入れなさい。
途中でソートが終了した場合は、それまでで良い。

初期

35	21	4	49	55	19	12	32
----	----	---	----	----	----	----	----

0 1 2 3 4 5 6 7

(1)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(2)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(3)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(4)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(5)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(6)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(7)

4	12	19	21	32	35	49	55
---	----	----	----	----	----	----	----

0 1 2 3 4 5 6 7

3. クイックソートを行なう際、基準値を21とした。下記のステップ(1)～(7)で、配列に適切な数字を入れなさい。2回目以降の基準値はそれぞれのグループの右端値とする。途中でソートが終了した場合は、それまでで良い。

初期

35	21	4	49	55	19	12	32
----	----	---	----	----	----	----	----

0 1 2 3 4 5 6 7

(1)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(2)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(3)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(4)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(5)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(6)

--	--	--	--	--	--	--	--

0 1 2 3 4 5 6 7

(7)

4	12	19	21	32	35	49	55
---	----	----	----	----	----	----	----

0 1 2 3 4 5 6 7

4. マージソートで、分割とマージのステップに当てはまる数字入れなさい。

