

# 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2版 5月7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	ソートアルゴリズムのまとめ 2クラス合同小テスト (教室は23・221)
6月	5日	(水)	第8回	グラフアルゴリズム1 (グラフとそのデータ構造)
	12日	(水)	第9回	グラフアルゴリズム2 (重み付きグラフ, 最短経路探索)
	19日	(水) 2EP2穴水	第10回	総合演習 (2EP2) / アルゴリズム設計手法 (2EP3)
	<del>26日</del>	<del>(水) 2EP3穴水</del>	<del>第11回</del>	<del>アルゴリズム設計手法 (2EP2) / 総合演習 (2EP3)</del>
	28日	(金) 4限	第11回	アルゴリズム設計手法 (2EP2) @23. 320 / 総合演習 (2EP3)
7月	3日	(水)	第12回	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

## ■ データの探索

- 線形探索法
- 2分探索法
- ハッシュ法
  - データの格納
    - 衝突
    - オープンアドレス法
    - チェイン法
  - データの探索

# 今回の内容

## ■ ソートアルゴリズム

- 選択ソート
- 挿入ソート
- ヒープソート
  - ヒープ(データ構造)
  - ヒープへのデータ追加
  - ヒープからの最大値取り出し
  - ヒープソート

# 選択ソート

## ■ アルゴリズム

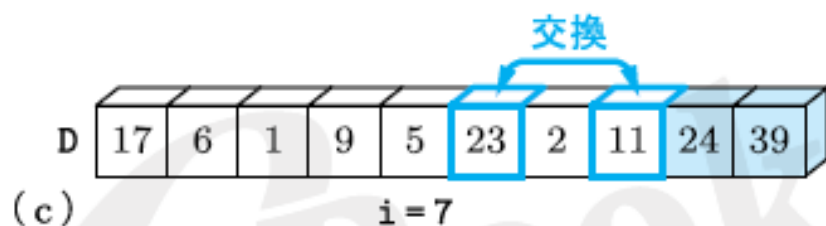
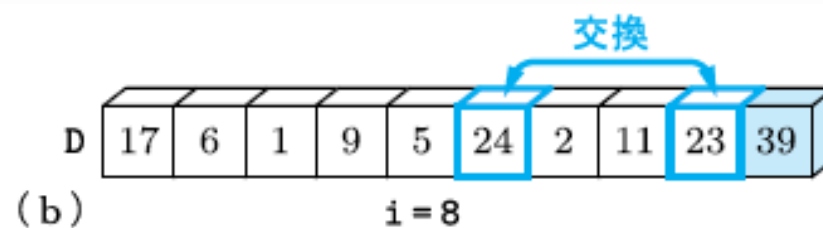
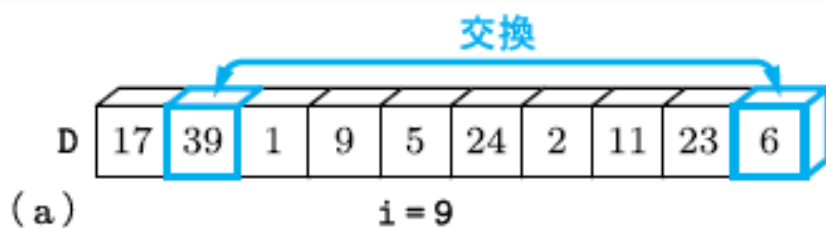
- 前提:  $n$ 個のデータが入力されている
- ① 入力データから最大値を見つける
- ② 見つけた最大値のデータを対象から外す
- ③ ①②の操作を  $n-1$  回繰り返す

## ■ 時間計算量

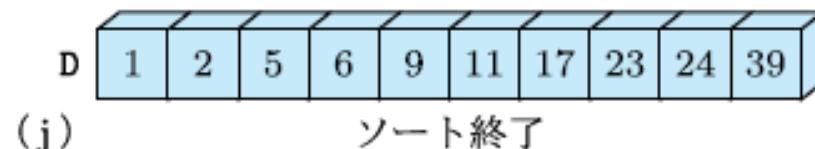
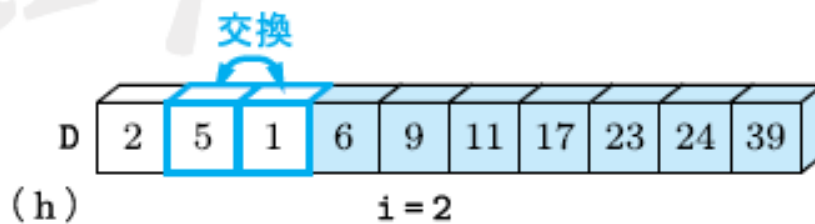
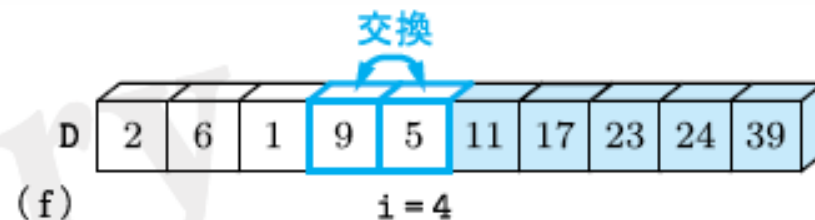
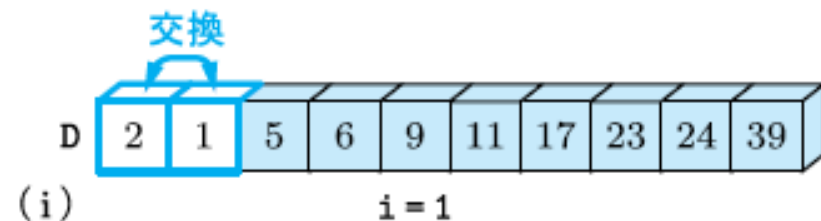
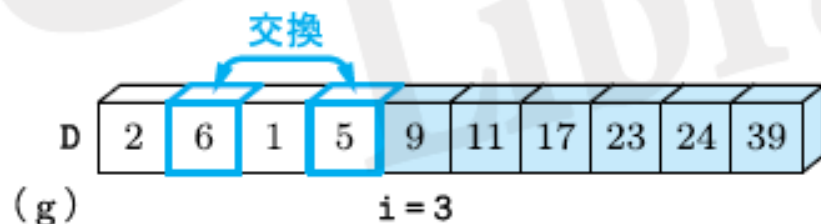
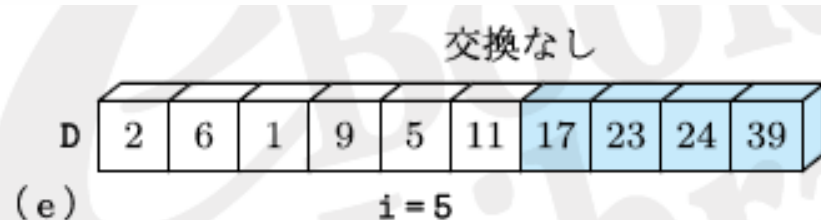
$$\begin{aligned}\sum_{i=1}^{n-1} i \times O(1) &= O(1) \times \frac{n(n-1)}{2} \\ &= O(n^2)\end{aligned}$$

## 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

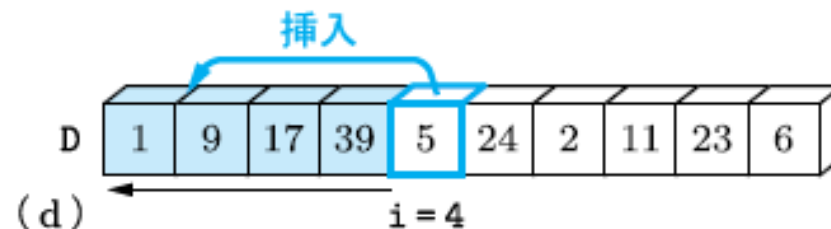
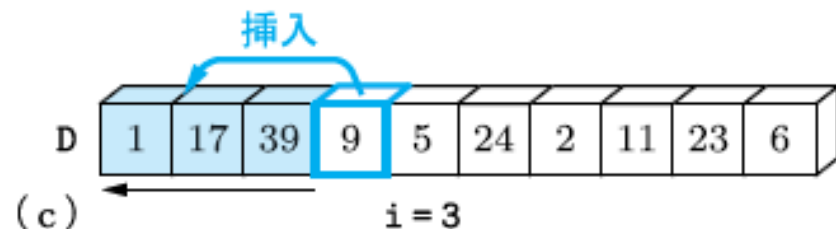
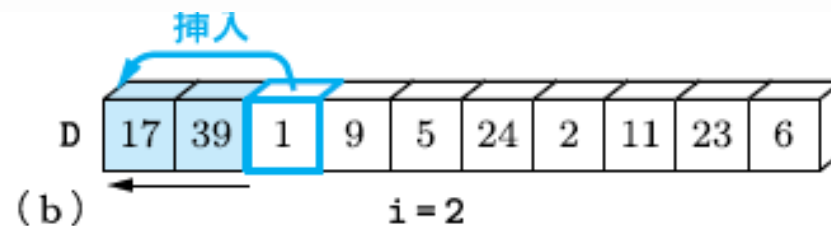
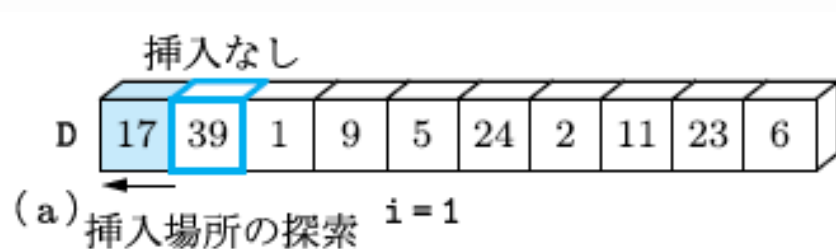
- 前提:  $n$ 個のデータをソートの対象とする
- ① 最初のデータを左端に置く
- ② 次のデータは元のデータに合わせて昇順に並べる
- ③ ①②の操作を  $n-1$  回繰り返す

## ■ 時間計算量

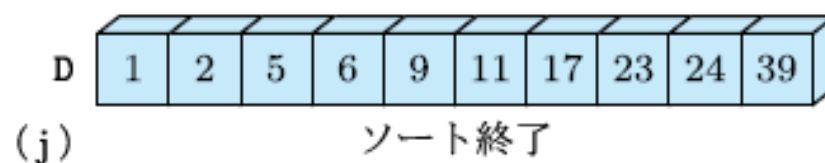
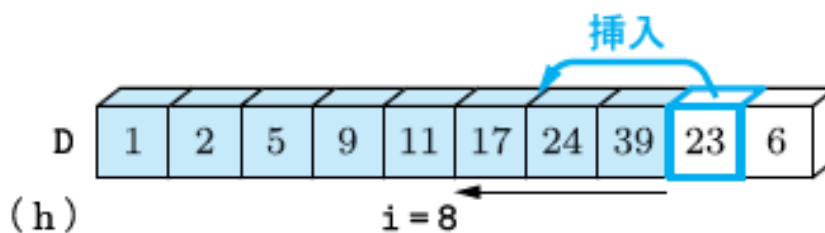
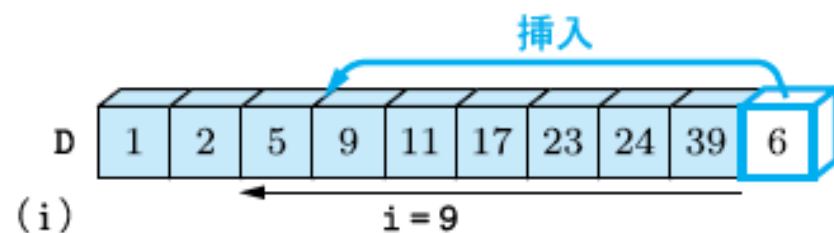
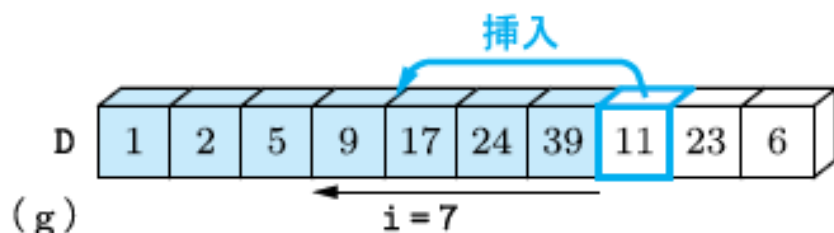
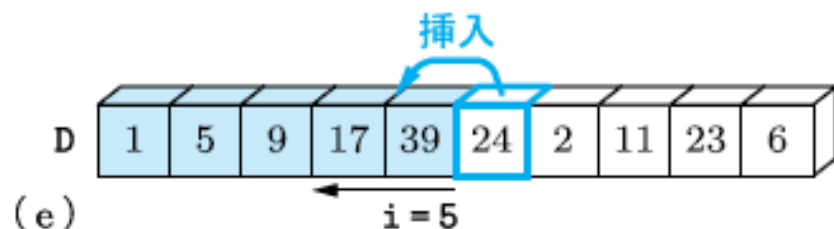
- 最良時間計算量: ソート済の場合  $O(n)$
- 最悪時間計算量: 選択ソートと同じ  $O(n^2)$
- 平均時間計算量:  $O(n^2)$

## 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

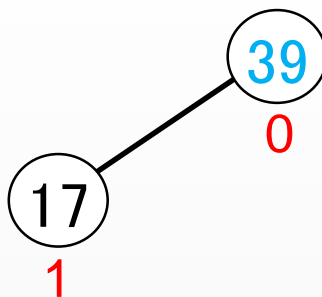
ヒープを  
表す配列

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



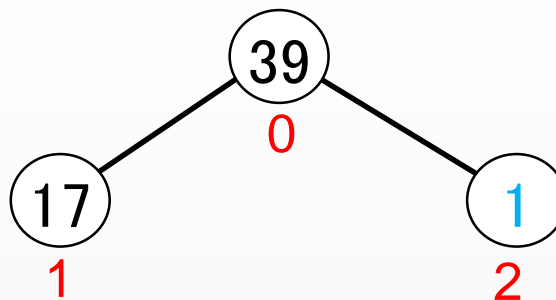
ヒープを  
表す配列

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



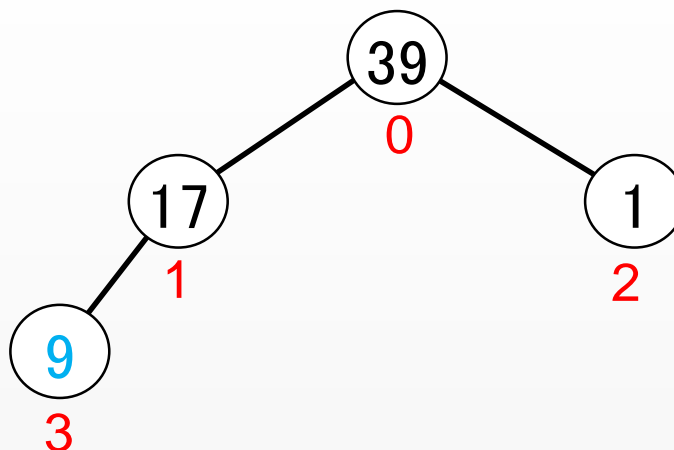
ヒープを  
表す配列

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

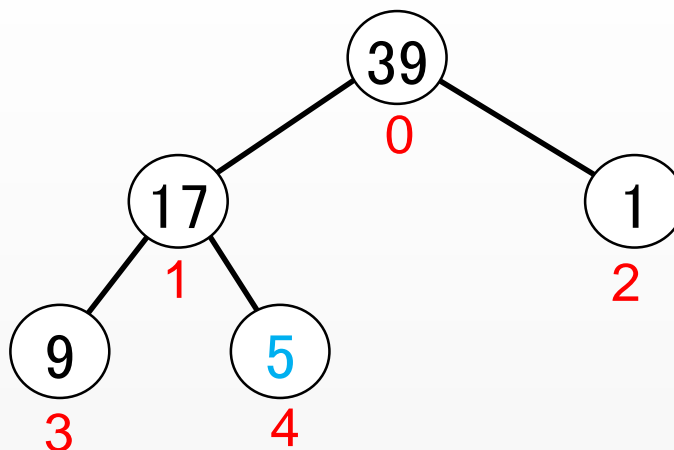
39	17	1	9						
0	1	2	3	4	5	6	7	8	9



# 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



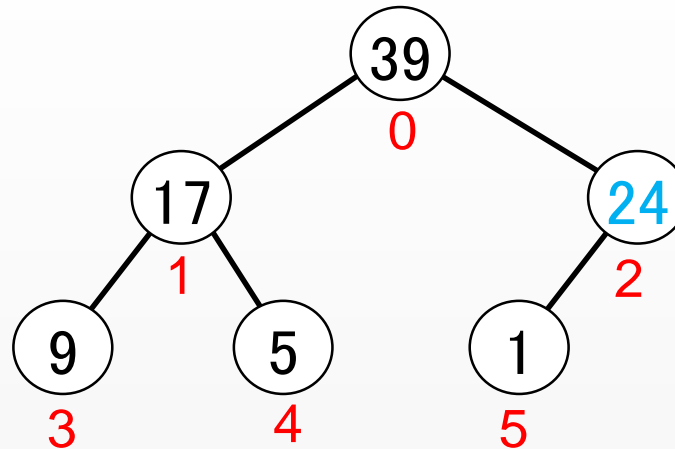
ヒープを  
表す配列

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



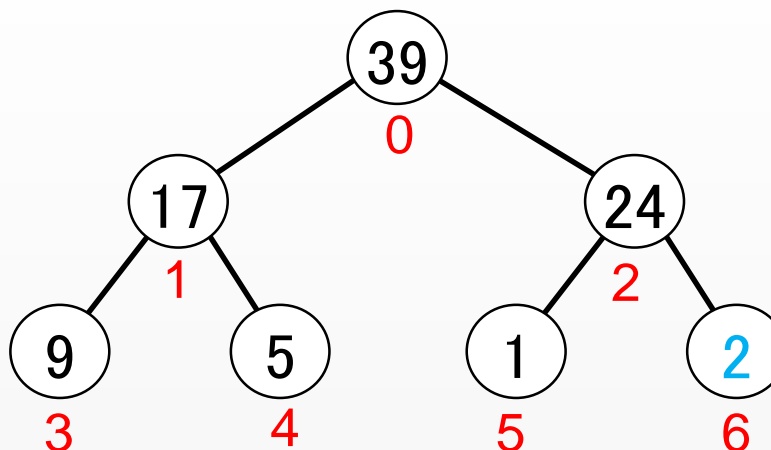
ヒープを  
表す配列

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



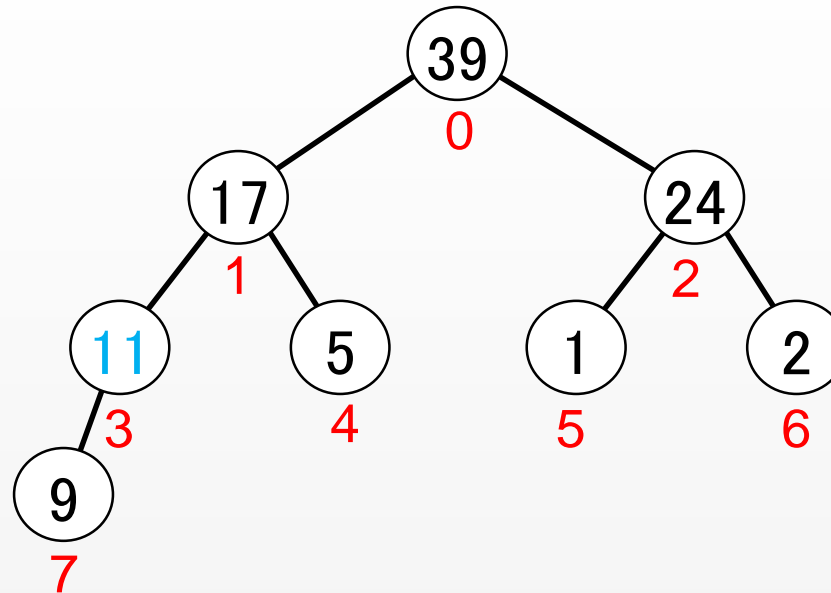
ヒープを  
表す配列

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



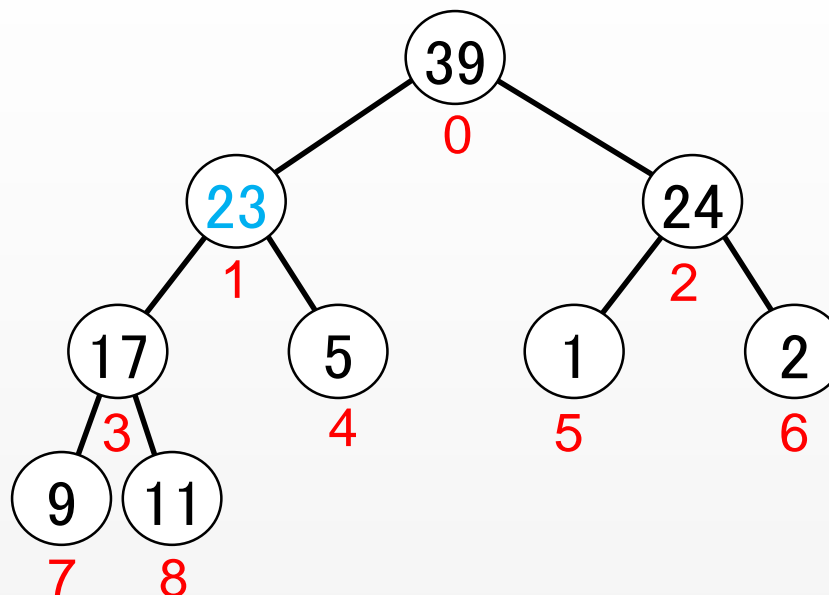
ヒープを  
表す配列

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



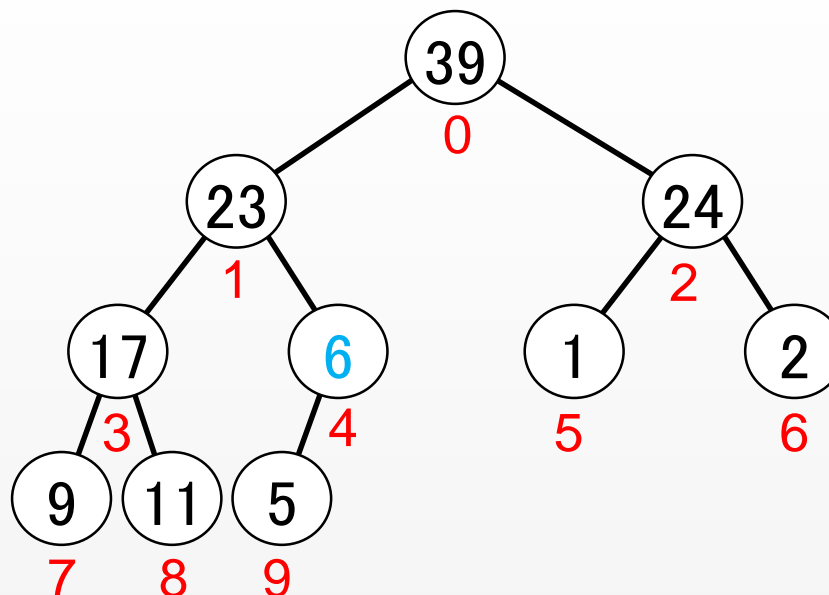
ヒープを  
表す配列

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

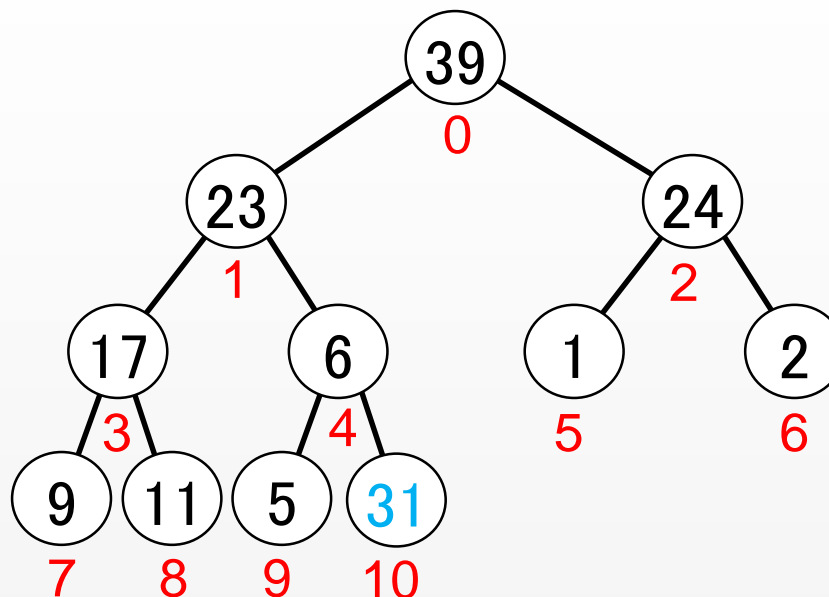
## ■ 方法

- 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

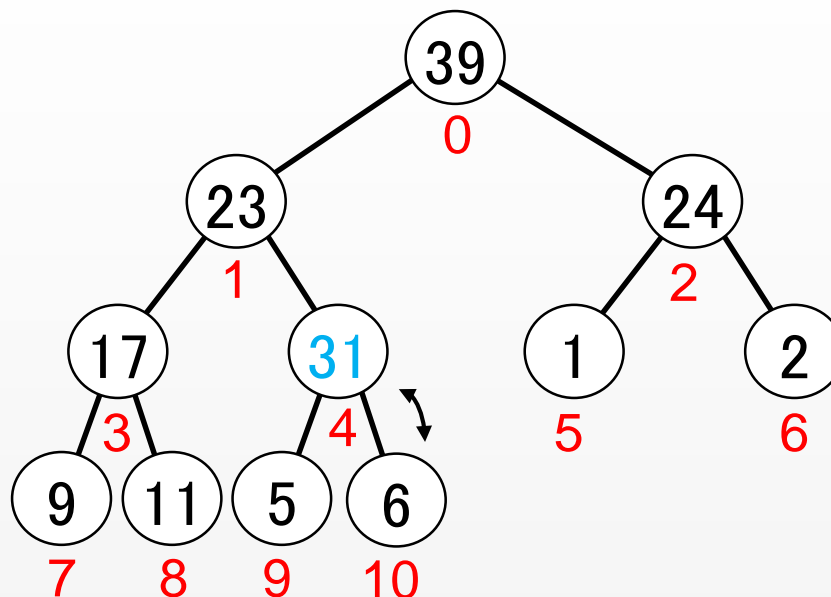
39	23	24	17	6	1	2	9	11	5	31
0	1	2	3	4	5	6	7	8	9	10



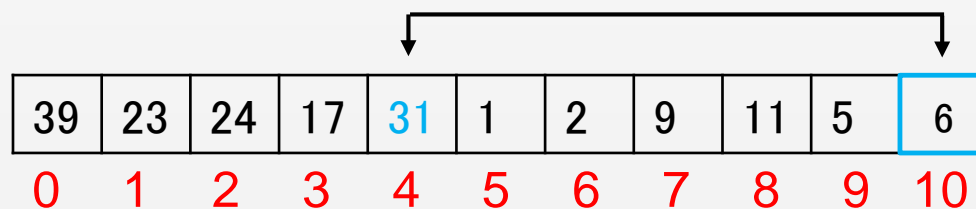
# ヒープへの追加

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

ヒープ



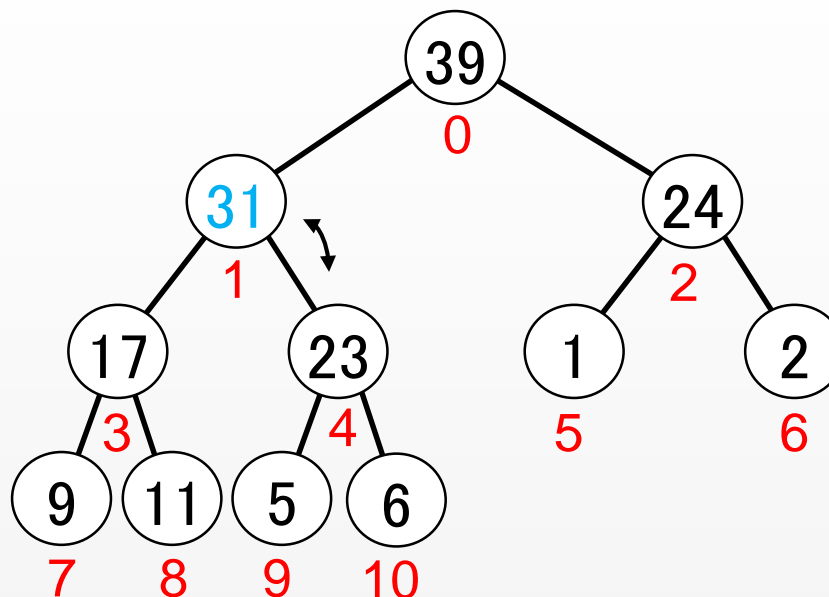
ヒープを  
表す配列



# 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

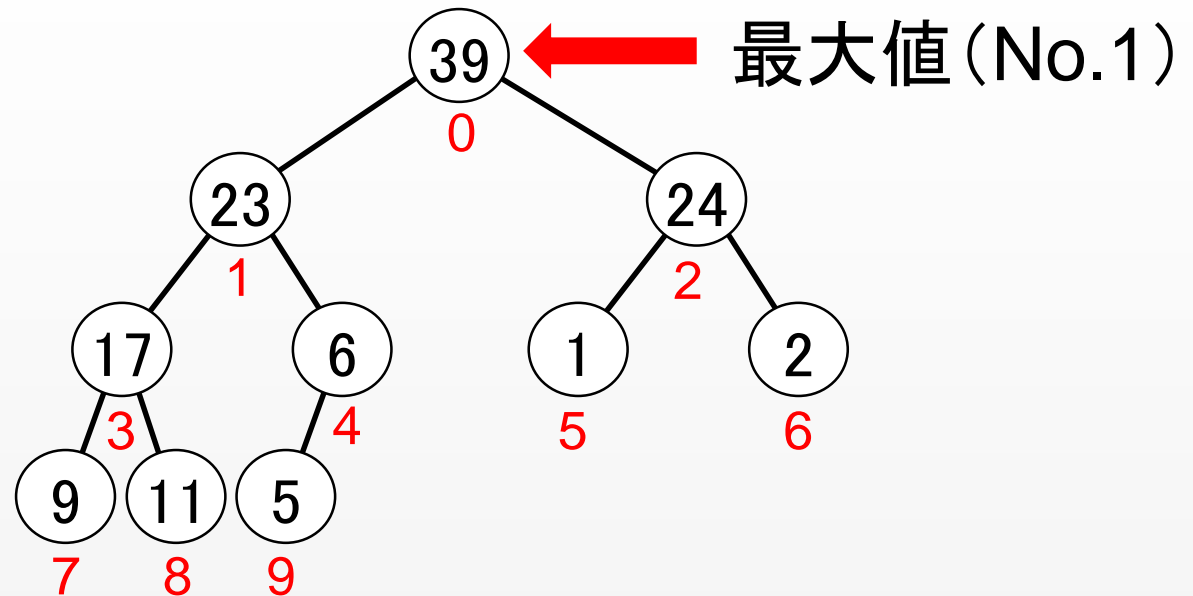
## ■ 方法

- 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

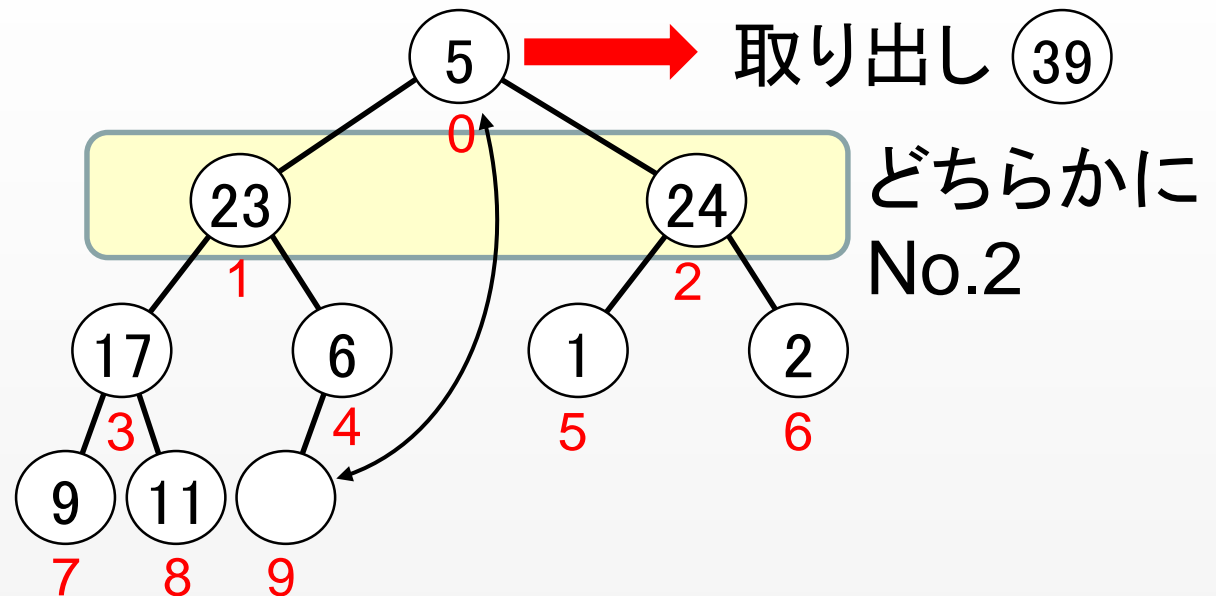


39	23	24	17	6	1	2	9	11	5
0	1	2	3	4	5	6	7	8	9

# 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



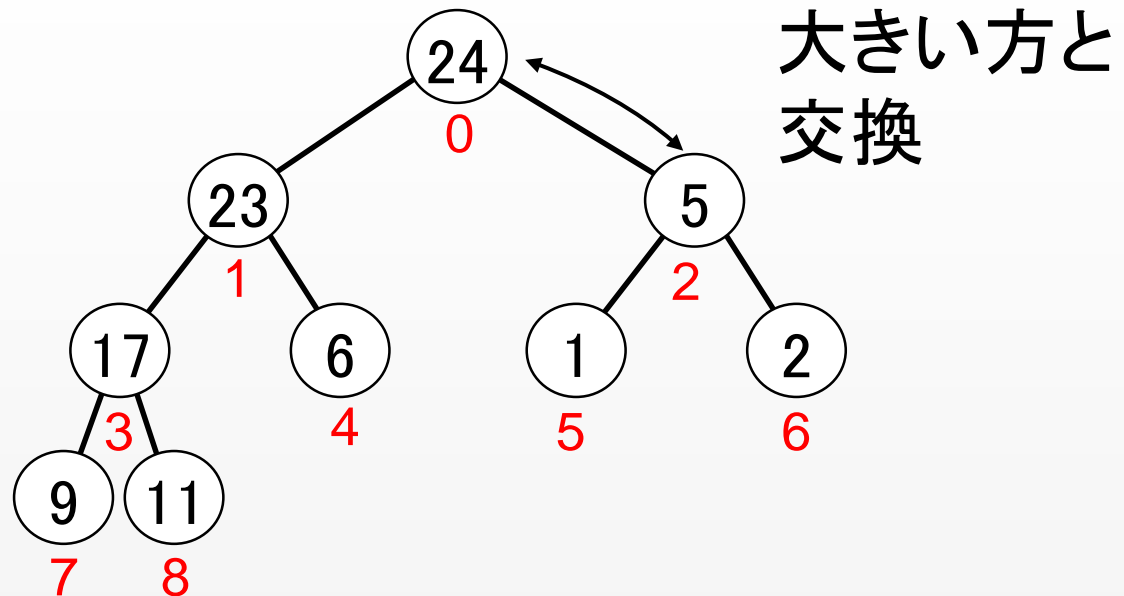
ヒープを  
表す配列

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

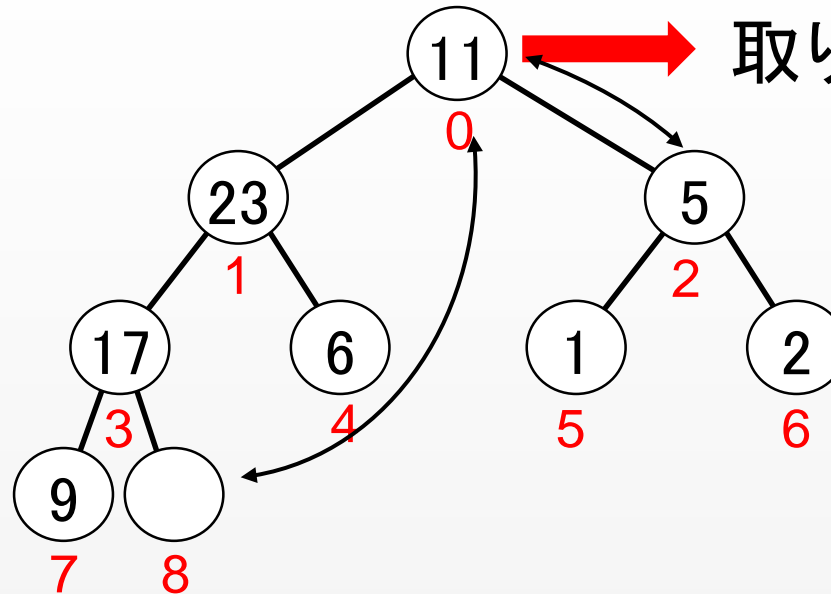
24	23	5	17	6	1	2	9	11	
0	1	2	3	4	5	6	7	8	9

# 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  $2 \times n \times \log n = O(n \log n)$



# 第5週出席課題

【出席課題】 学籍番号：

クラス・番号：

氏名：

1. 以下の文章の①～⑦について、それぞれ正しい記号を下から選べ。正しい記号が複数存在する場合はすべて列挙せよ。ただし、②～⑦については、もっとも適切なものを1つだけ選ぶこと。

ソートとは、与えられたデータを決められた順番に並べるという操作であるが、ソートの入力は( ① )。

$n$  個のデータを格納しているヒープに対して、データを1つ追加するのに必要な時間計算量は( ② )であり、最大のデータを削除するのに必要な時間計算量は( ③ )である。

サイズが $n$ のデータに対する挿入ソートの最良時間計算量は( ④ )であり、最悪時間計算量は( ⑤ )である。また、同じデータに対するヒープソートの最良時間計算量は( ⑥ )であり、最悪時間計算量は( ⑦ )である。

- ①: a. 整数でなければならない  
b. 全順序関係が成り立たなければならない  
c. 同じ値が存在してはいけない  
d. 昇順に並んでいなければならない
- ②: a.  $O(n^2)$    b.  $O(n)$    c.  $O(\log n)$    d.  $O(1)$
- ③: a.  $O(n^2)$    b.  $O(n)$    c.  $O(\log n)$    d.  $O(1)$
- ④: a.  $O(n^2)$    b.  $O(n \log n)$    c.  $O(n)$    d.  $O(1)$
- ⑤: a.  $O(n^2)$    b.  $O(n \log n)$    c.  $O(n)$    d.  $O(1)$
- ⑥: a.  $O(n^2)$    b.  $O(n \log n)$    c.  $O(n)$    d.  $O(1)$
- ⑦: a.  $O(n^2)$    b.  $O(n \log n)$    c.  $O(n)$    d.  $O(1)$