优先队列、堆排序

优先队列

- 优先队列
  - 一种关于集合S的数据结构，集合中的元素都有键值key
  - 最大优先队列支持以下操作：
    - **INSERT(S, x)**：把元素x插入集合S，可表示为$S \leftarrow S \cup \{x\}$
    - **MAXIMUM(S)**：返回集合S中具有最大键值的元素
    - **EXTRACT-MAX(S)**：去掉并返回集合S中具有最大键值的元素
    - **INCREASE-KEY(S, x, k)**：将元素x的键值增加到k，这里要求k不能小于x的原键值。
  - 类似的，最小优先队列支持以下操作：
    - **INSERT(S, x)**、**MININUM(S)**
    - **EXTRACT-MIN(S)**、**DECREASE-KEY(S, x, k)**

二叉堆

- 二叉堆是在一个数组上通过下标间关系维护父子结点关系的一棵几乎满的二叉树

PARENT(*i*)

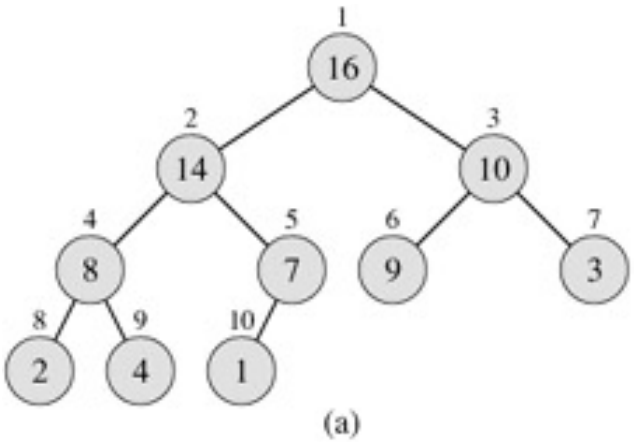
return $\lfloor i/2 \rfloor$

LEFT(*i*)

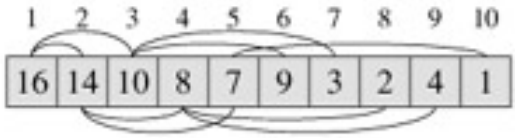
return $2i$

RIGHT(*i*)

return $2i + 1$



(a)



(b)

保持堆的性质

MAX-HEAPIFY(*A*, *i*)

1 *l* ← LEFT(*i*)

2 *r* ← RIGHT(*i*)

3 if *l* ≤ *heap-size*[*A*] and *A*[*l*] > *A*[*i*]

4 then *largest* ← *l*

5 else *largest* ← *i*

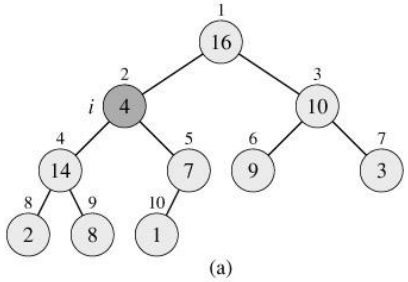
6 if *r* ≤ *heap-size*[*A*] and *A*[*r*] > *A*[*largest*]

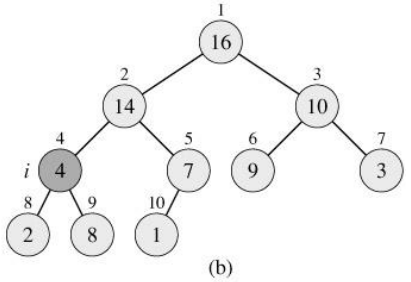
7 then *largest* ← *r*

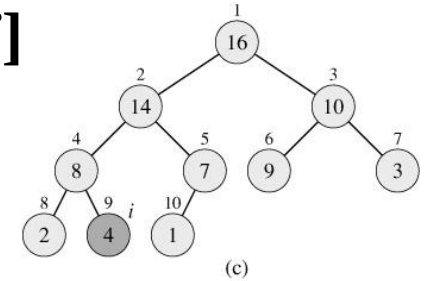
8 if *largest* ≠ *i*

9 then exchange *A*[*i*] ↔ *A*[*largest*]

10 MAX-HEAPIFY(*A*, *largest*)







建堆

BUILD-MAX-HEAP(*A*)

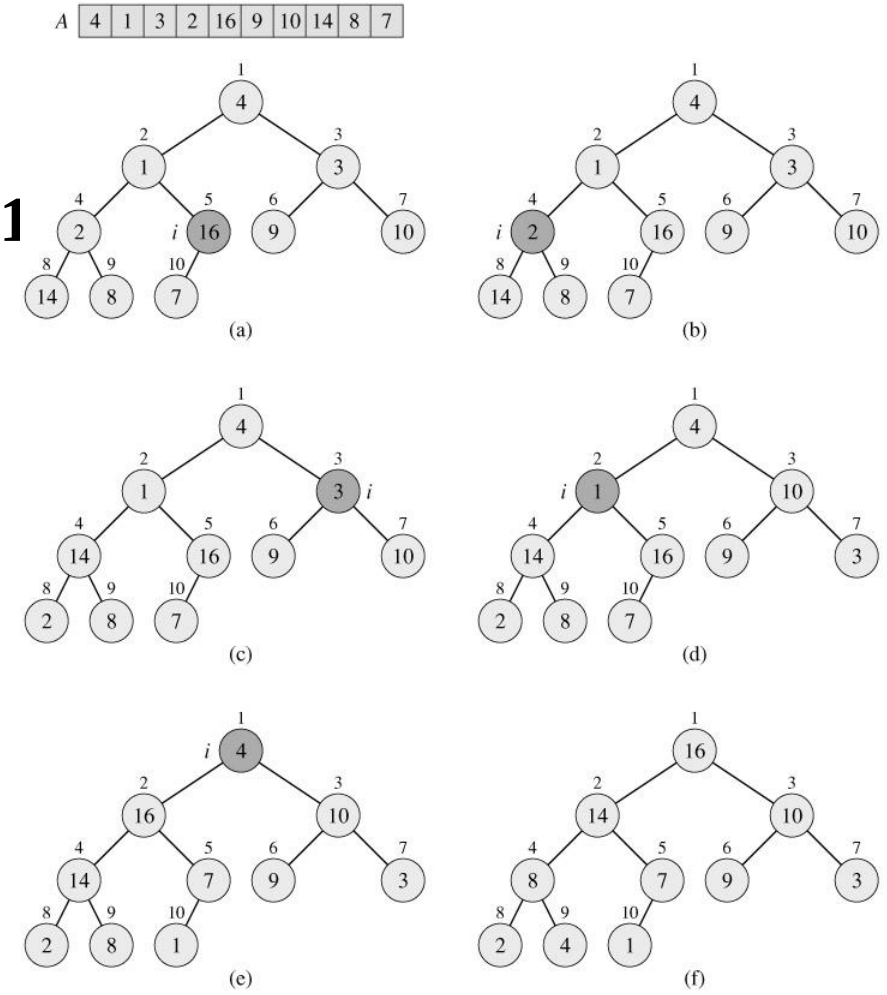
1 *heap-size*[*A*] ← *length*[*A*]

2 for *i* ← [*length*[*A*]/2] downto 1

3 do MAX-HEAPIFY(*A*, *i*)

总运行时间为**O(*n*)**。

合并堆的过程需要用重建堆来实现。



取最大元素 / 取出最大元素

HEAP-MAXIMUM(*A*)

1 return *A*[1]

HEAP-EXTRACT-MAX(*A*)

1 if *heap-size*[*A*] < 1

2 then error "heap underflow"

3 *max* ← *A*[1]

4 *A*[1] ← *A*[*heap-size*[*A*]]

5 *heap-size*[*A*] ← *heap-size*[*A*] - 1

6 MAX-HEAPIFY(*A*, 1)

7 return *max*

增加元素的键值 / 插入新元素

HEAP-INCREASE-KEY(*A*, *i*, *key*)

```
1 if key < A[i]
2     then error "new key is smaller than current key"
3 A[i] ← key
4 while i > 1 and A[PARENT(i)] < A[i]
5     do exchange A[i] ↔ A[PARENT(i)]
6         i ← PARENT(i)
```

MAX-HEAP-INSERT(*A*, *key*)

```
1 heap-size[A] ← heap-size[A] + 1
2 A[heap-size[A]] ← -∞
3 HEAP-INCREASE-KEY(A, heap-size[A], key)
```