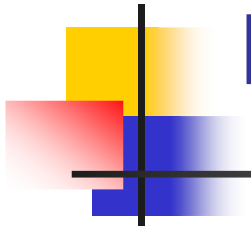
Priority queues

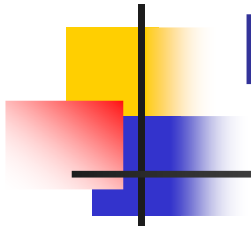
Paolo Camurati

Dip. Automatica e Informatica
Politecnico di Torino



Priority queues

- Heaps have many applications beyond heap-sort
- A priority queue is a data structure to store elements including a priority field such that all main operations are based on such a field
- Priority queues have several applications
  - Job scheduling
  - Etc.



Priority queues

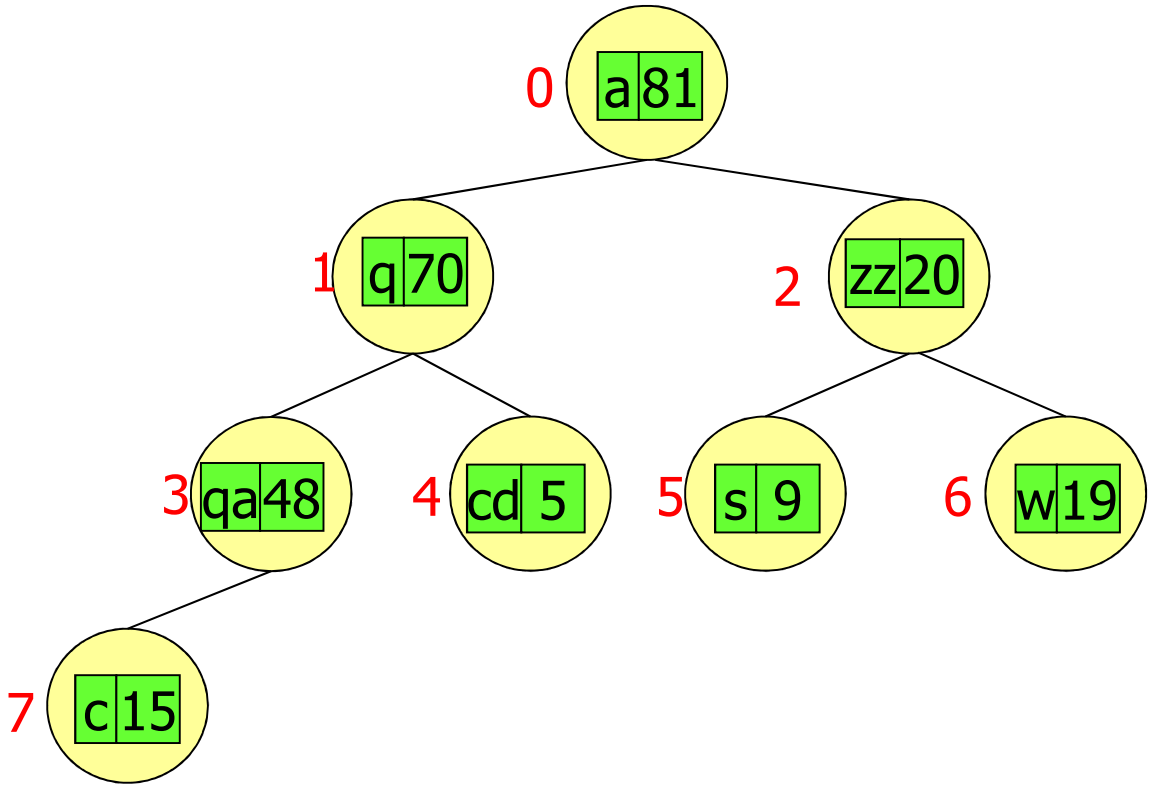
- It is possible to implement
  - Min-priority queues
  - Max-priority queues
- Main operations
  - Insert, extract maximum, read maximum, change priority
- Possible alternative data structure implementations
  - Unordered array/list
  - Ordered array/list

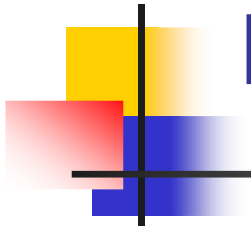
Example

pq->A

a	q	zz	qa	cd	s	w	c		
81	70	20	48	5	9	19	15		
0	1	2	3	4	5	6	7	8	9

pq->heapsize 8





PQinsert function

- Add a leaf to the tree
  - It grows level-by-level from left to right satisfying the structural property
- From current node up (initially the newest leaf) up to the root
  - Compare the parent's key with the new node's key, moving the parent's data from the parent to the child when the key to insert is larger
  - Otherwise insert the data into the current node
- Complexity
  - $T(n) = O(\lg n)$

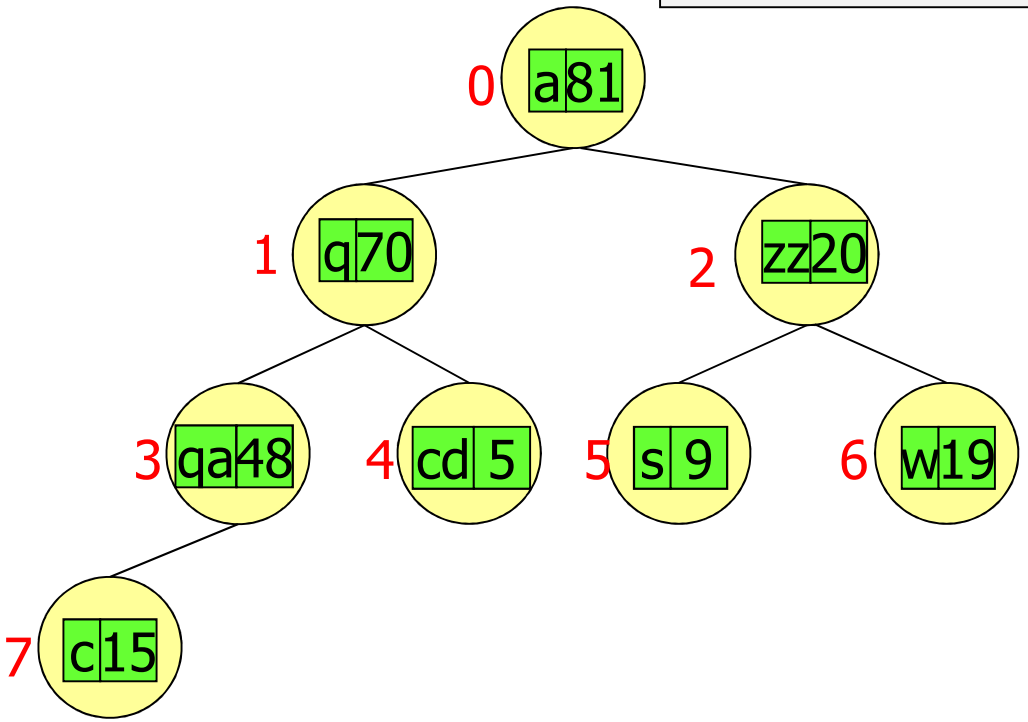
Example

Insertion of r 75

pq->A

a	q	zz	qa	cd	s	w	c		
81	70	20	48	5	9	19	15		
0	1	2	3	4	5	6	7	8	9

pq->heapsize 8

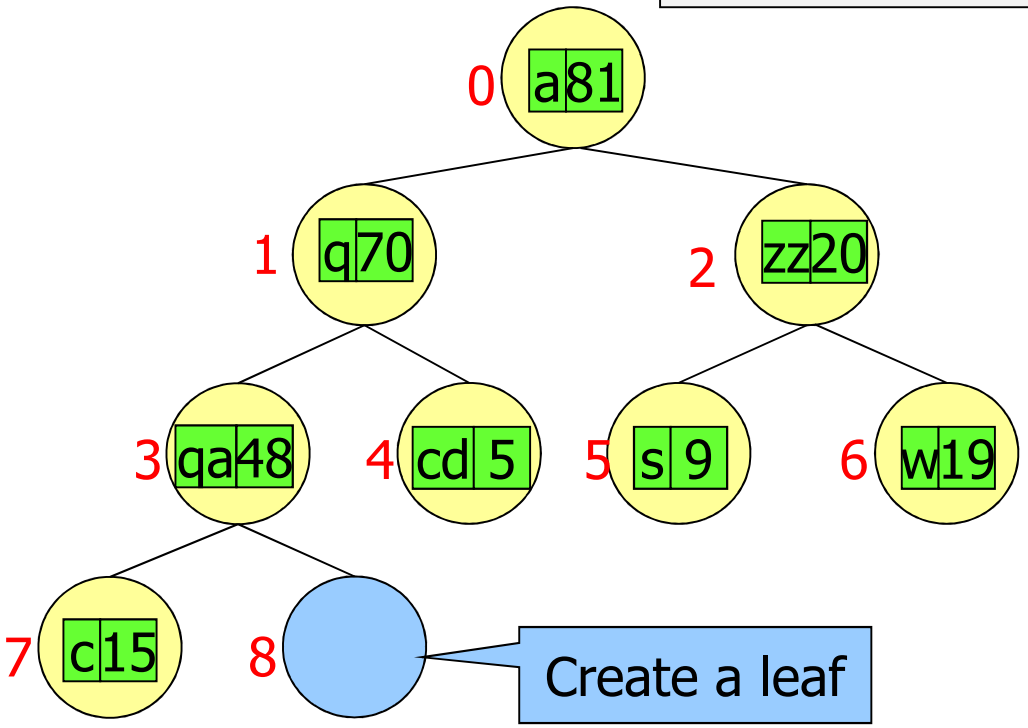


Example

pq->A

a	q	zz	qa	cd	s	w	c		
81	70	20	48	5	9	19	15		
0	1	2	3	4	5	6	7	8	9

pq->heapsize 9



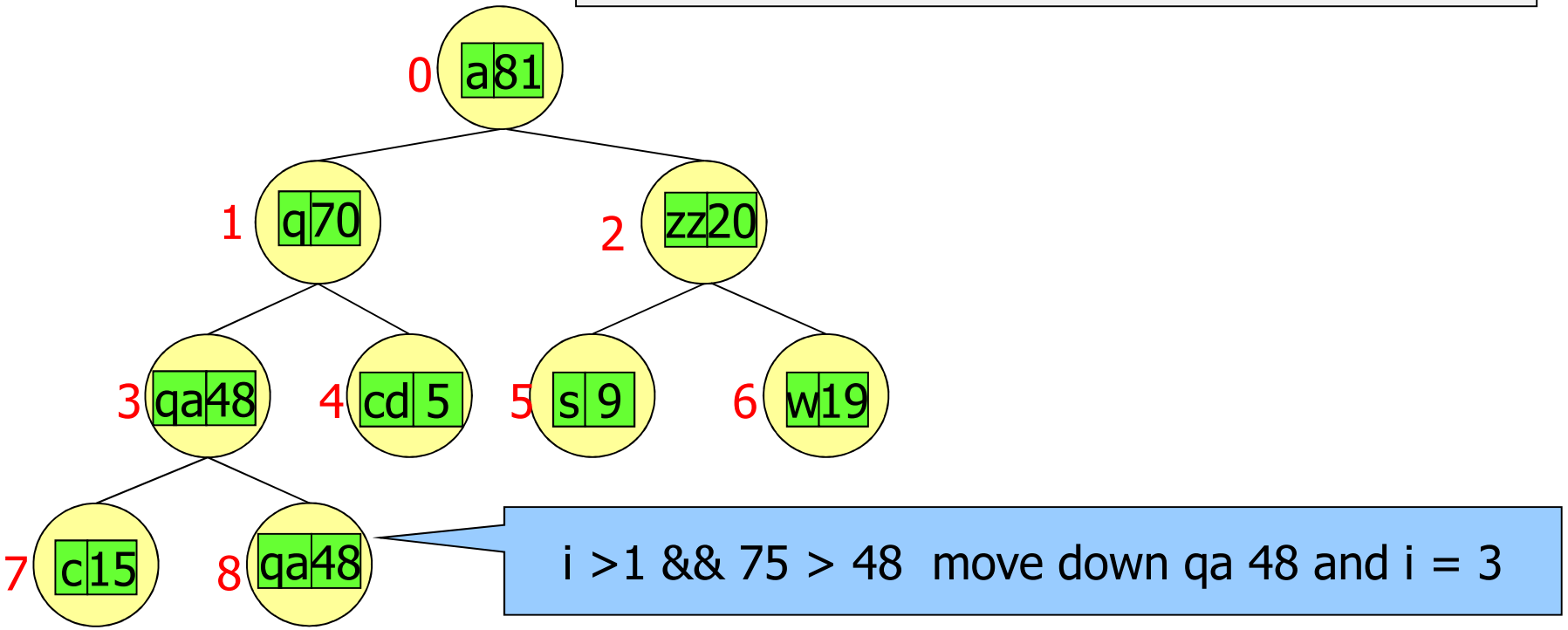
i = 9

Example

pq->A

a	q	zz	qa	cd	s	w	c	qa	
81	70	20	48	5	9	19	15	48	
0	1	2	3	4	5	6	7	8	9

pq->heapsize 9



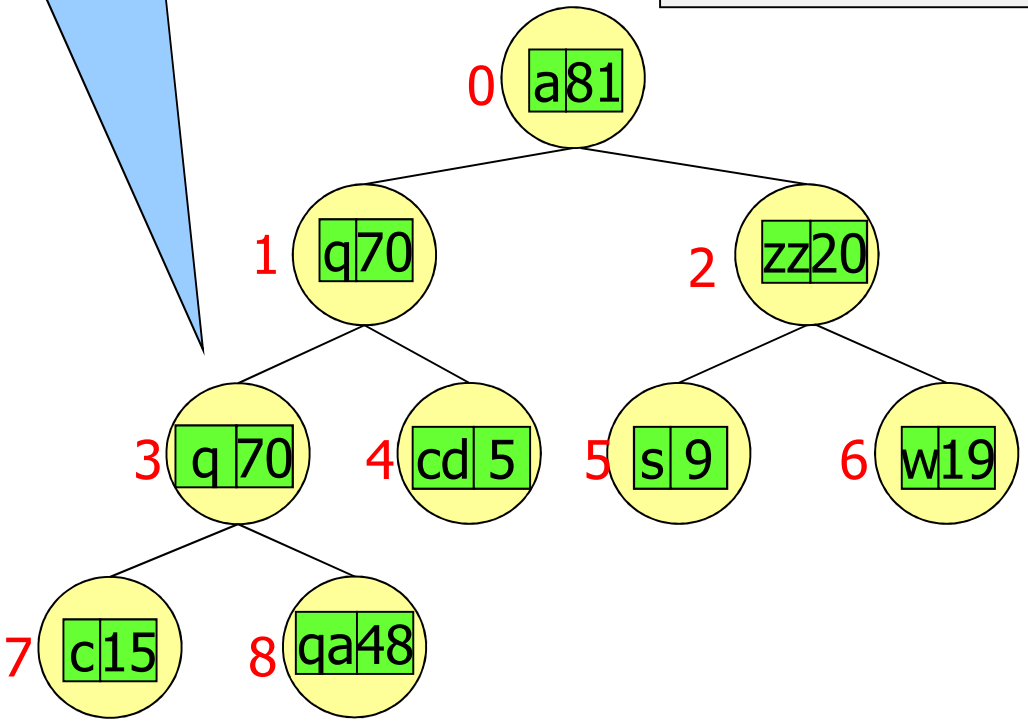
Example

pq->A

a	q	zz	q	cd	s	w	c	qa	
81	70	20	70	5	9	19	15	48	
0	1	2	3	4	5	6	7	8	9

pq->heapsize 9

i >1 && 75 > 70
move down
q 70 and i = 1



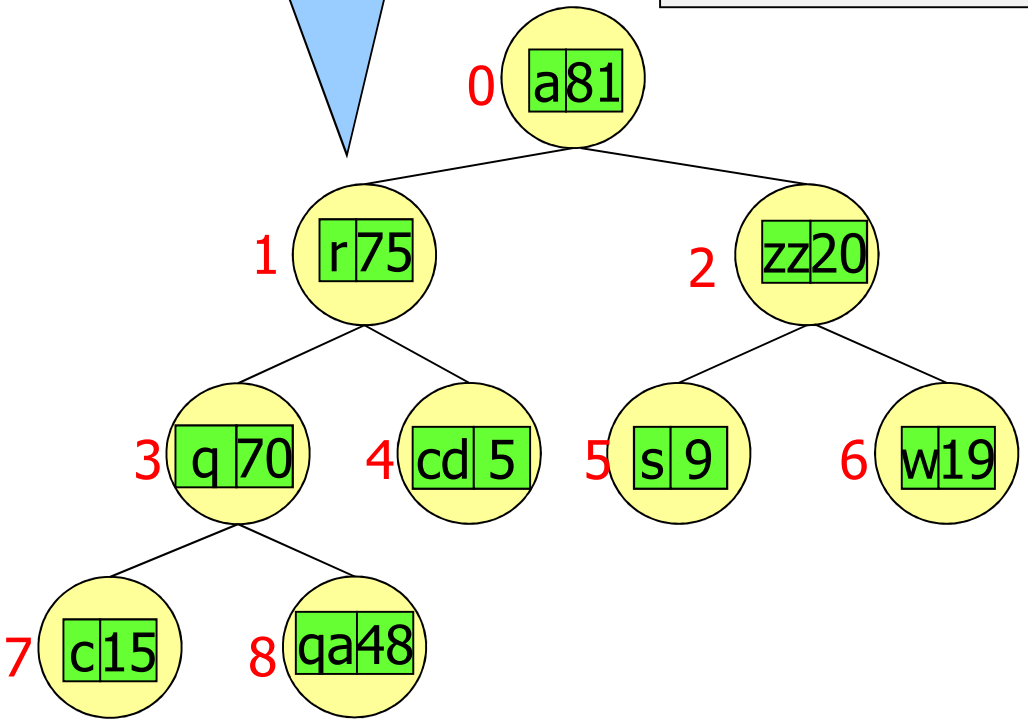
Example

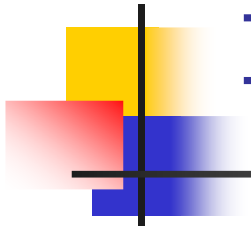
pq->A

a	r	zz	q	cd	s	w	c	qa	
81	75	20	70	5	9	19	15	48	
0	1	2	3	4	5	6	7	8	9

pq->heapsize 9

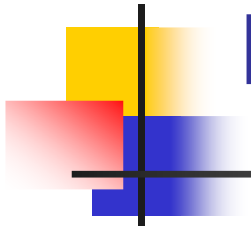
i >1 && 75 < 81
insert r 75





Implementation

```
void PQinsert (PQ pq, Item item) {
  int i;
  i  = pq->heapsize++;
  while( (i>=1) &&
         (ITEMless(pq->A[PARENT(i)], item)) )
    pq->A[i] = pq->A[PARENT(i)];
    i = PARENT (i);
  }
  pq->A[i] = item;
  return;
}
```



PQextractMax function

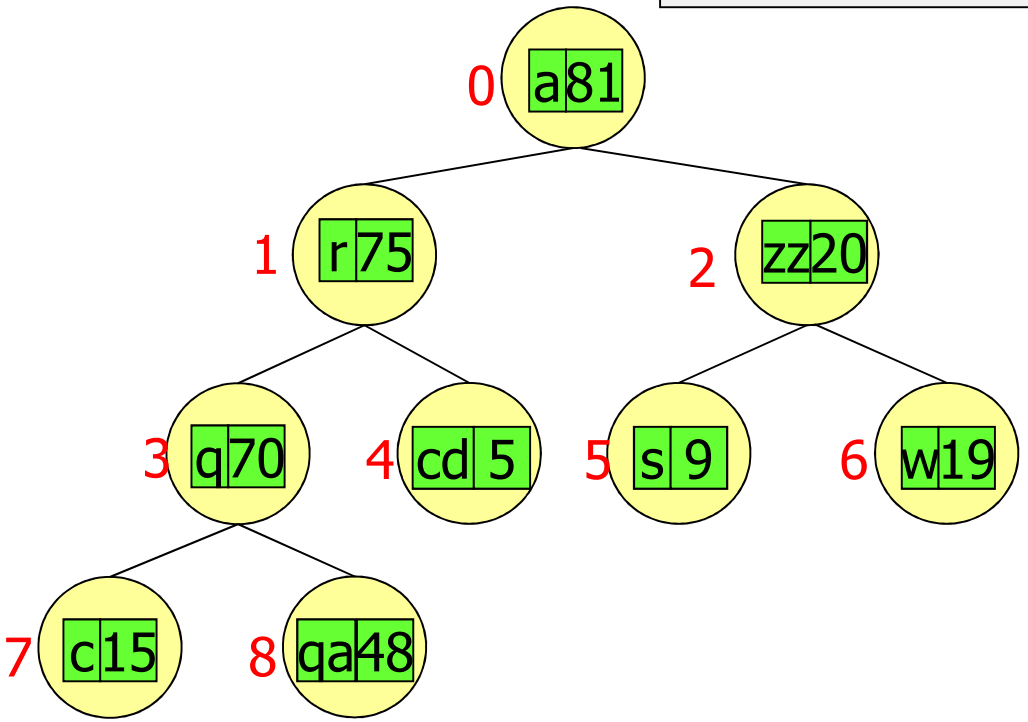
- Modify the head, by extracting the largest value, stored into the root:
  - Swap root with the last leaf (the rightmost onto the last level)
  - Reduce by 1 the heap size
  - Restore the heap property by applying HEAPify
- Complexity
  - $T(n) = O(\lg n)$

Example

pq->A

a	r	zz	q	cd	s	w	c	qa	
81	75	20	70	5	9	19	15	48	
0	1	2	3	4	5	6	7	8	9

pq->heapsize 9

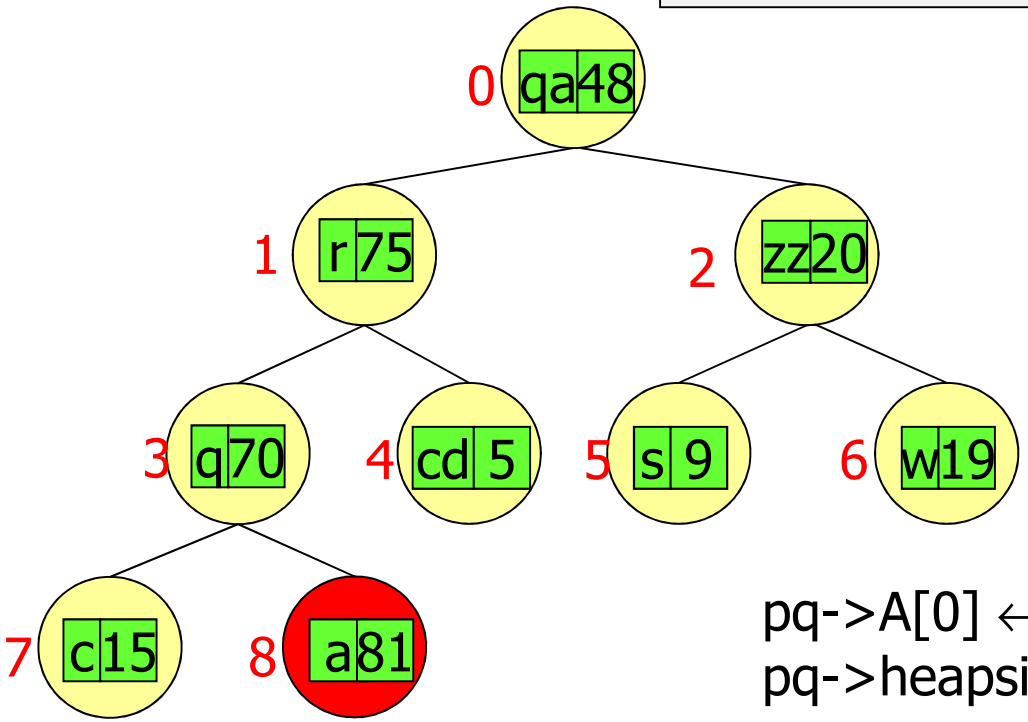


Example

pq->A

qa	r	zz	q	cd	s	w	c	a	
48	75	20	70	5	9	19	15	81	
0	1	2	3	4	5	6	7	8	9

pq->heapsize 8



pq->A[0] ↔ pq->A[pq->heapsize-1]
pq->heapsize--

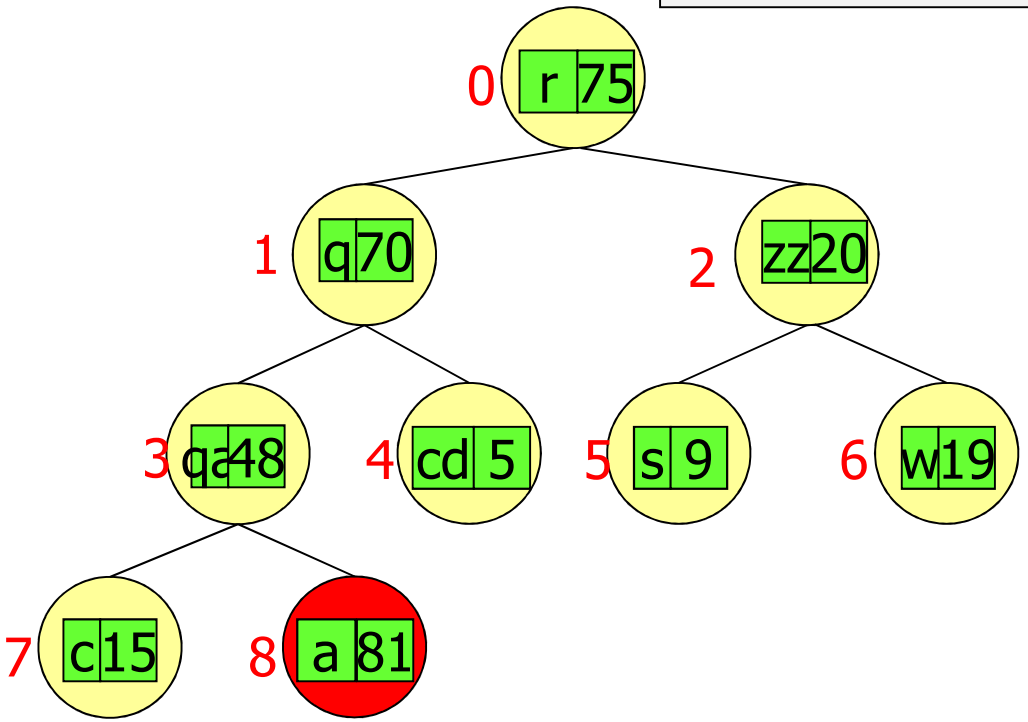
Example

HEAPify(pq->A, 0)

pq->A

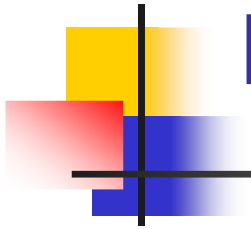
r	q	zz	qa	cd	s	w	c	a	
75	70	20	48	5	9	19	15	81	
0	1	2	3	4	5	6	7	8	9

pq->heapsize 8



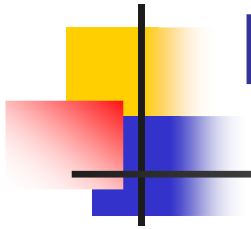
Implementation

```
Item PQextractMax(PQ pq) {
  Item item;
  Swap (pq, 0,pq->heapsize-1);
  item = pq->A[pq->heapsize-1];
  pq->heapsize--;
  HEAPify(pq, 0);
  return item;
}
```



PQchange function

- Modify the property of an element **in a given position** (the value of the heap index is known)
- Moving
  - Up from the given position to the root at most by comparing the parent's key with the modified key
  - Moving down from the parent to the child the largest key, otherwise insert the key into the current node
- Apply HEAPify starting from the given position



PQchange function

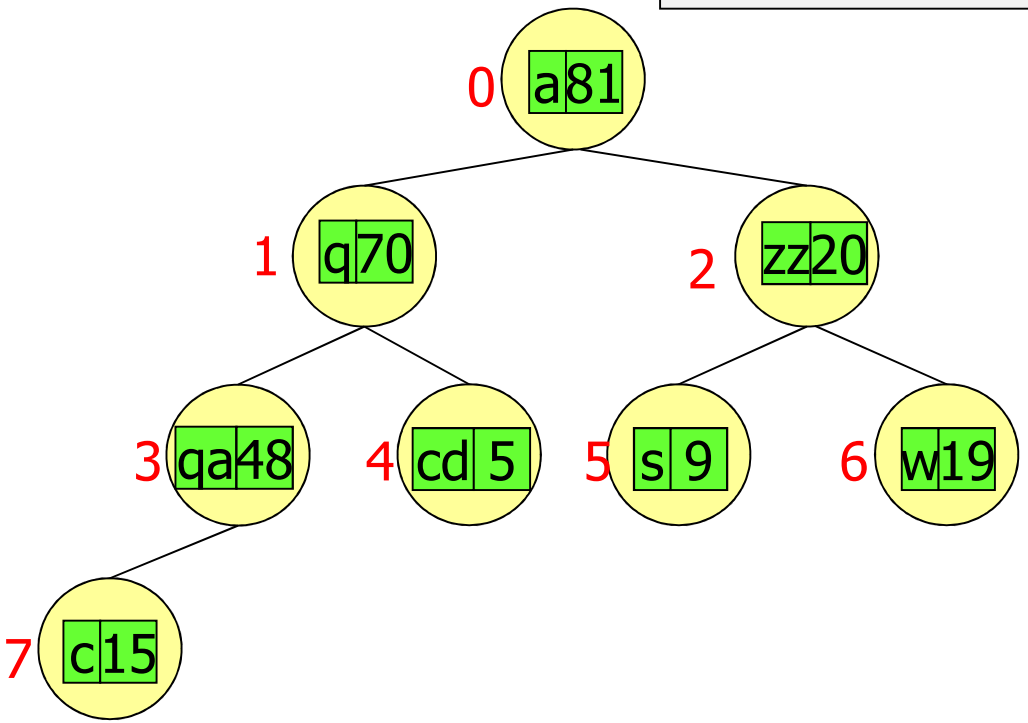
- Can also be implemented as two separate functions
  - Decrease key
  - Increase key
- Complexity
  - T(n) = O(lg n)

Example: Increase key

pq->A

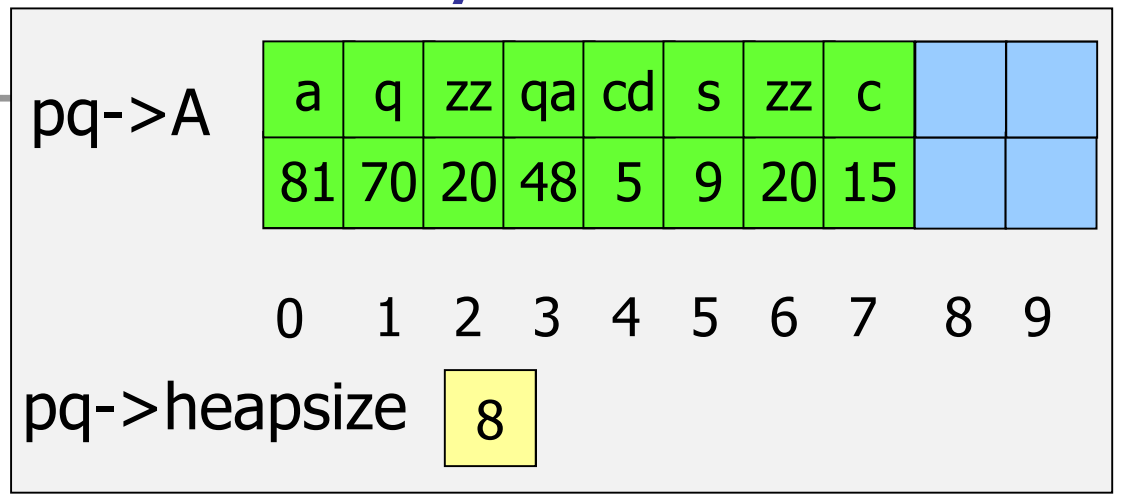
a	q	zz	qa	cd	s	w	c		
81	70	20	48	5	9	19	15		
0	1	2	3	4	5	6	7	8	9

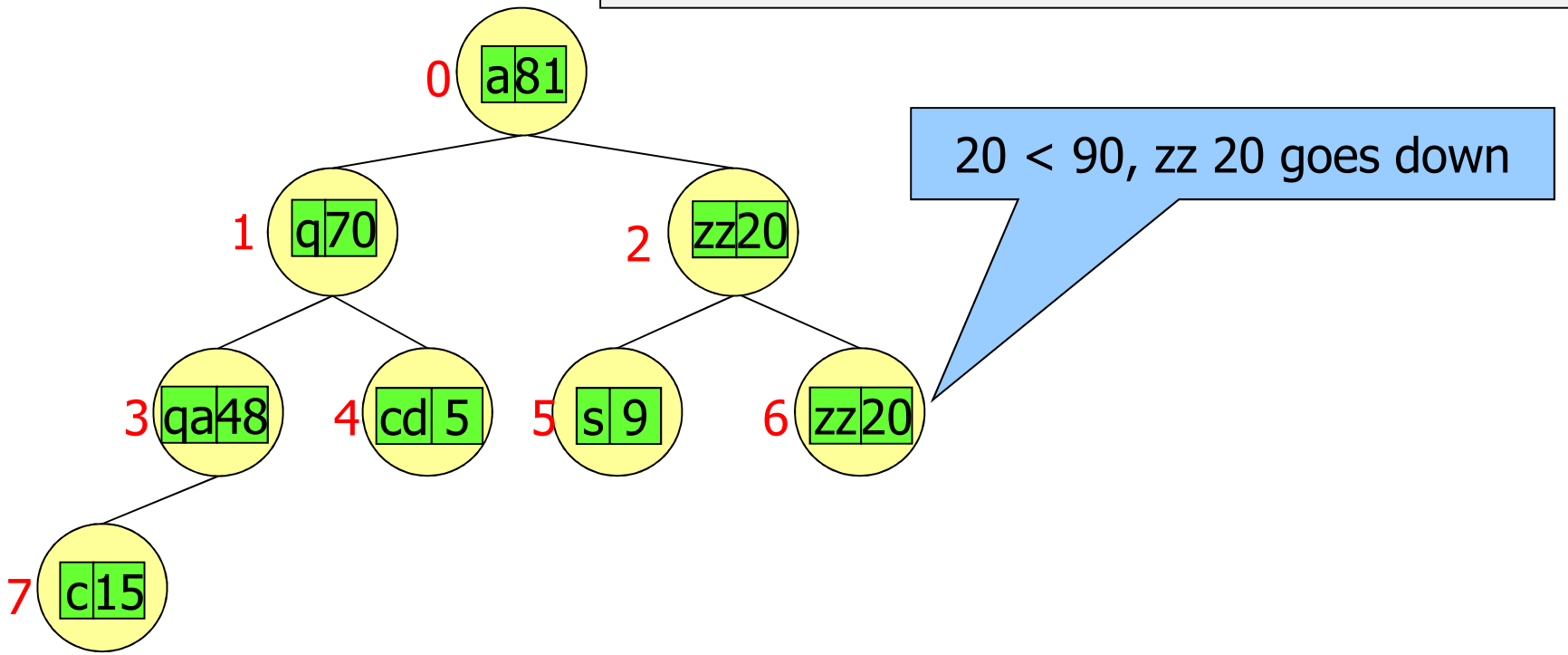
pq->heapsize 8



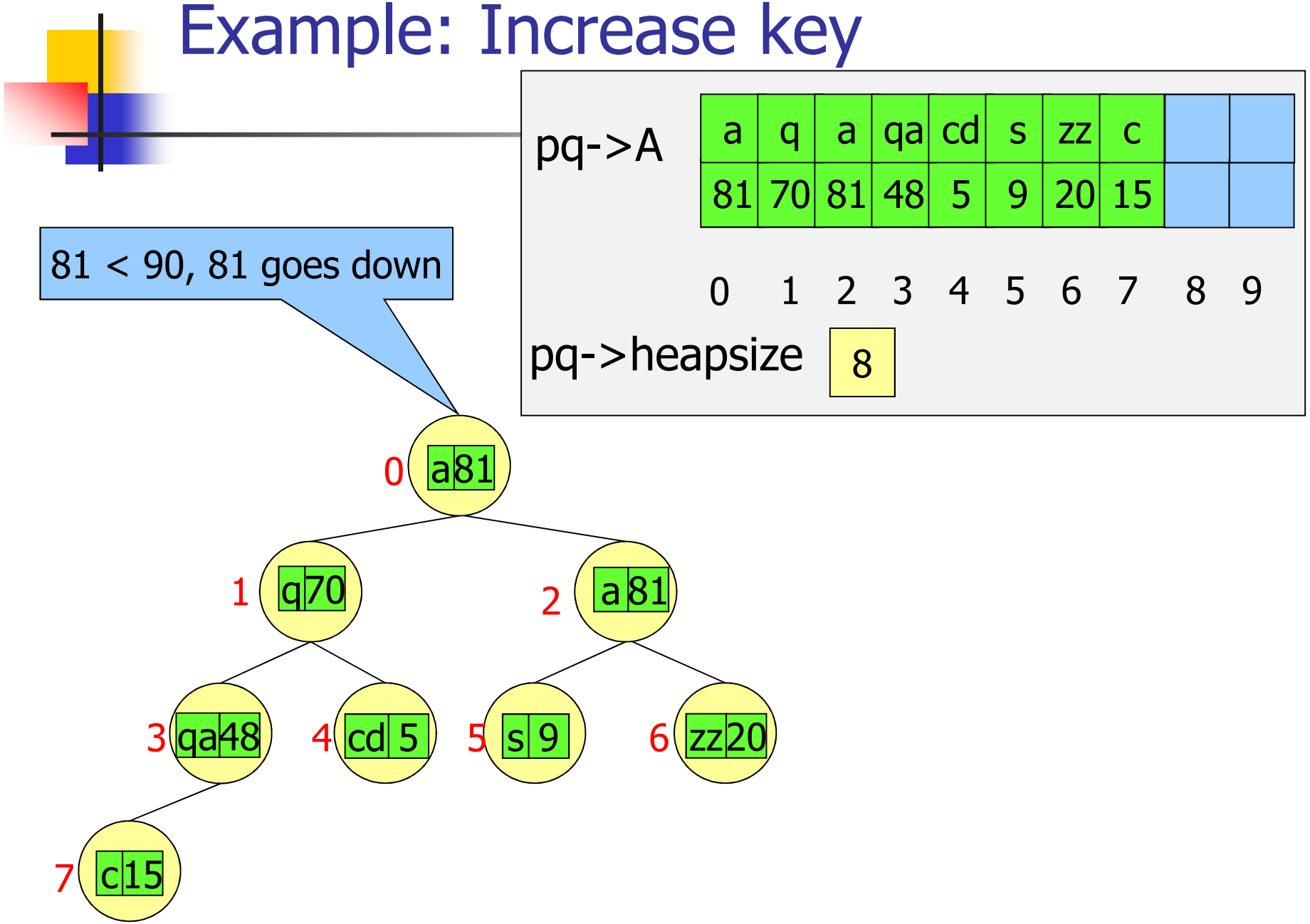
Change priority of element in position 6 from 19 to 90

Example: Increase key

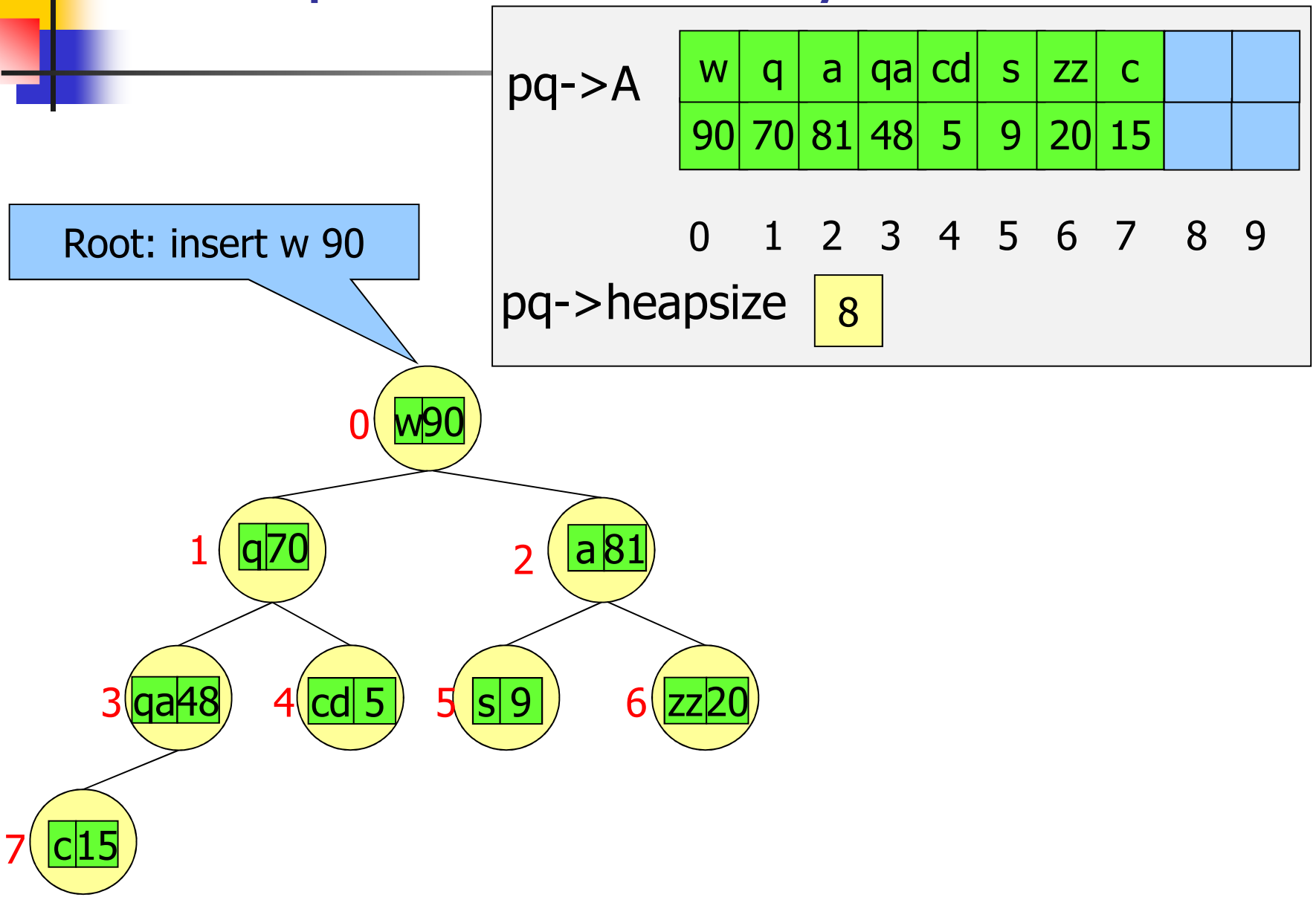




Example: Increase key



Example: Increase key

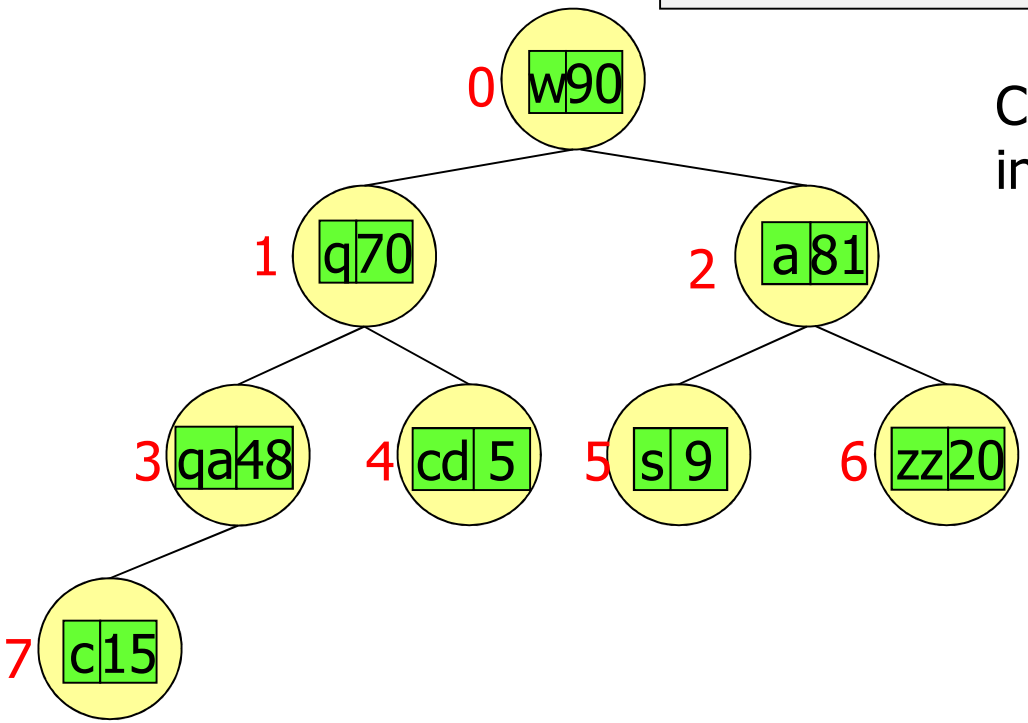


Example: Decrease key

pq->A

w	q	a	qa	cd	s	zz	c		
90	70	81	48	5	9	20	15		
0	1	2	3	4	5	6	7	8	9

pq->heapsize 8



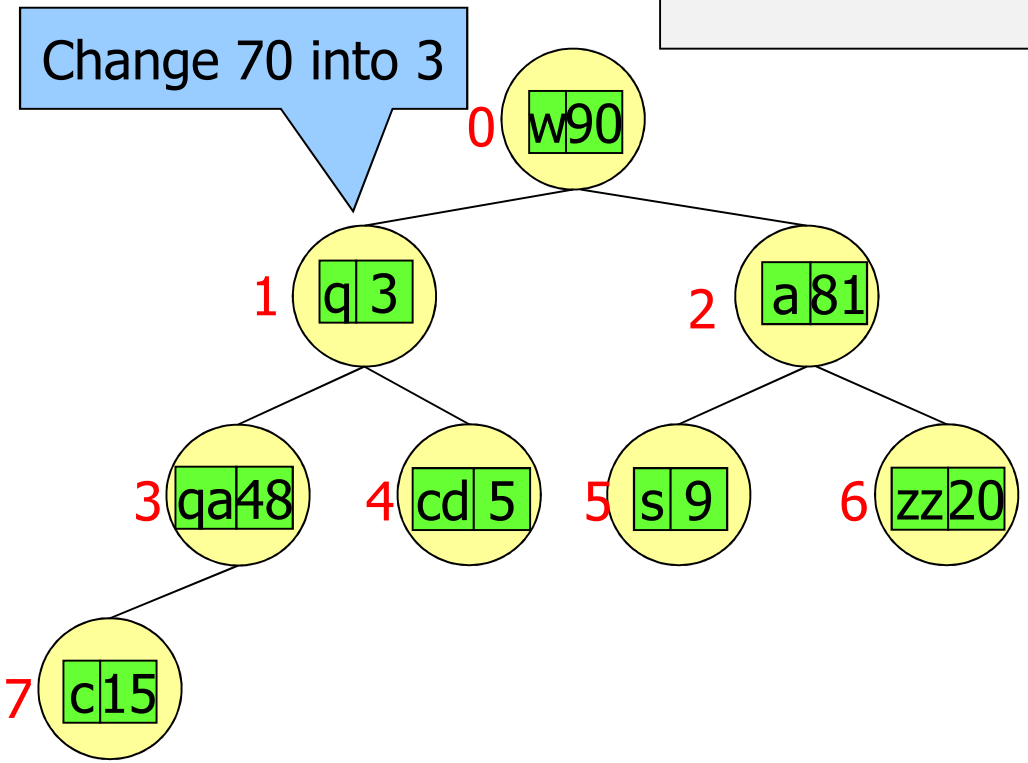
Change priority element
in position 1 from 70 to 3

Example: Decrease key

pq->A

w	q	a	qa	cd	s	zz	c		
90	3	81	48	5	9	20	15		
0	1	2	3	4	5	6	7	8	9

pq->heapsize 8



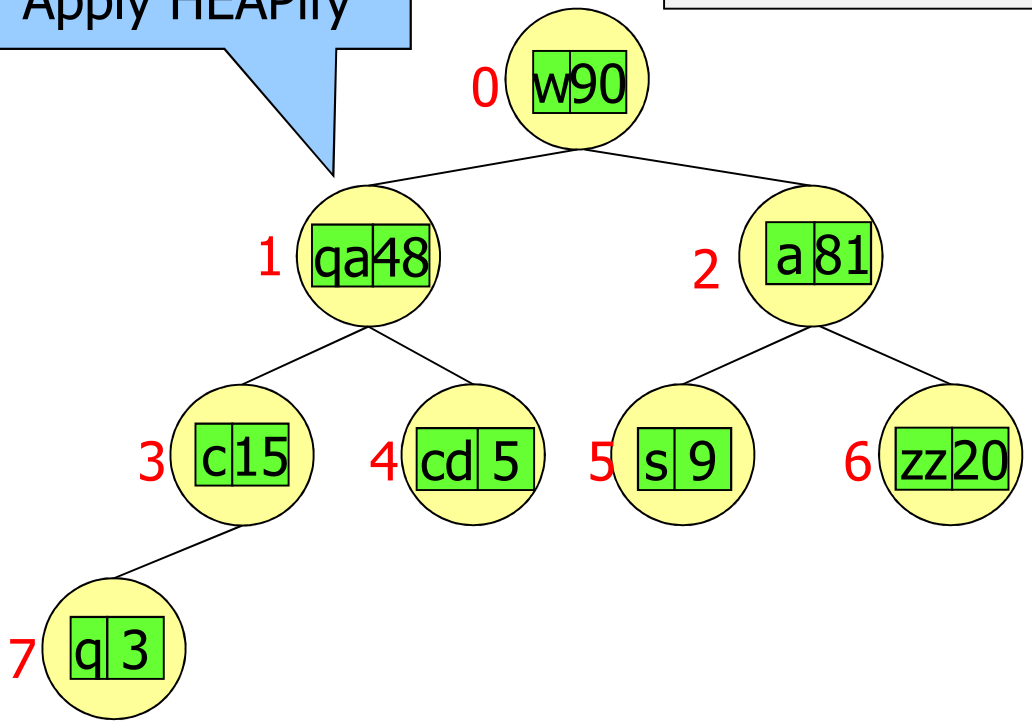
Example: Decrease key

pq->A

w	qa	a	c	cd	s	zz	q		
90	48	81	15	5	9	20	3		
0	1	2	3	4	5	6	7	8	9

pq->heapsize 8

Apply HEAPify



Implementation

```
void PQchange (PQ pq, int i, Item item) {
  while( (i>=1) &&
         (ITEMless(pq->A[PARENT(i)], item)) ) {
    pq->A[i] = pq->A[PARENT(i)];
    i = PARENT(i);
  }
  pq->A[i] = item;
  HEAPify(pq, i);
  return;
}
```

Increase key

Decrease key