NANYANG TECHNOLOGICAL UNIVERSITY

CE2001/ CZ2001: Algorithms

Appendix

(Merge operation in Mergesort)

Ke Yiping, Kelly

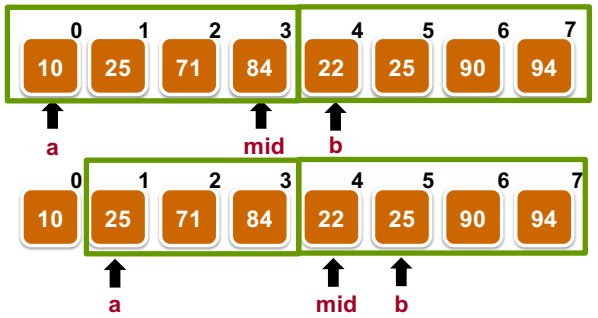
Merge Function

```
void merge(int n, int m)
  {
    int mid = (n+m)/2;
    int a = n, b = mid+1, i, tmp;
    if (m-n <= 0) return;
    while (a <= mid && b <= m) {
        cmp = compare(slot[a], slot[b]);
        if (cmp > 0) { //slot[a] > slot[b]
            tmp = slot[b++];
            for (i = ++mid; i > a; i--)
                slot[i] = slot[i-1];
```


Merge Function

```
            slot[a++] = tmp;
        } else if (cmp < 0) //slot[a] < slot[b]
            a++;
        else {   //slot[a] == slot[b]
            if (a == mid && b == m)
                break;
            tmp = slot[b++];
            a++;
            for (i = ++mid; i > a; i--)
                slot[i] = slot[i-1];
            slot[a++] = tmp;
        }
    } // end of while loop;
  } // end of merge
```


Merge Operation



Parameters for merge:

n:0, m: 7

mid = (0+7)/2 = 3;

a = n; **b** = mid+1;

Comparison:

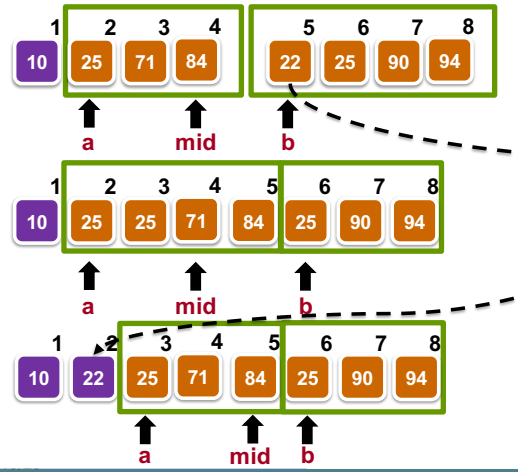
slot[a] < slot[b]

a : the 1st element of the 1st half

mid : the last element of the 1st half

b : the 1st element of the 2nd half

Merge Operation



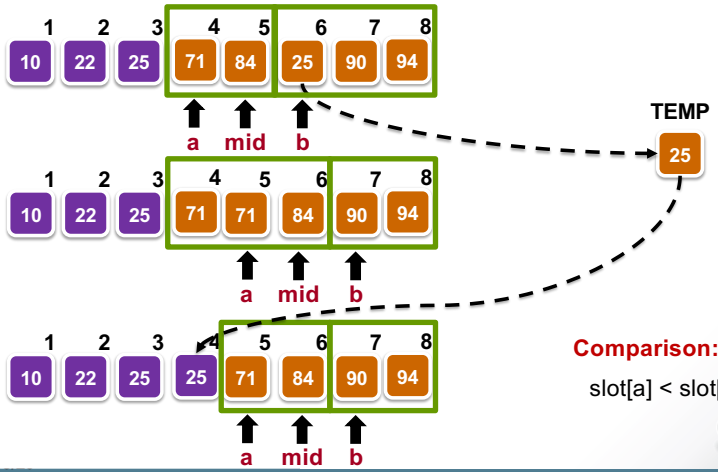
Comparison:

slot[a] > slot[b]

Comparison:

slot[a] == slot[b]

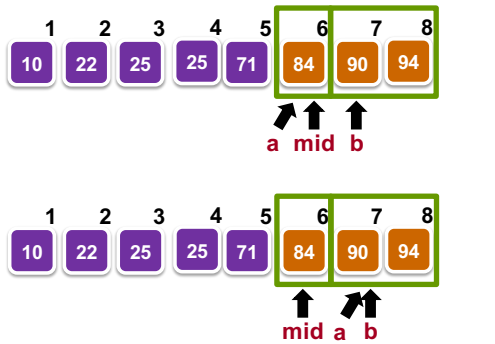
Merge Operation



Comparison:

slot[a] < slot[b]

Merge Operation



Comparison:

slot[a] < slot[b]

1st half empty

Merge operation completed