make history.

THE UNIVERSITY of ADELAIDE

Merge sort

Dr. Anna Kalenkova

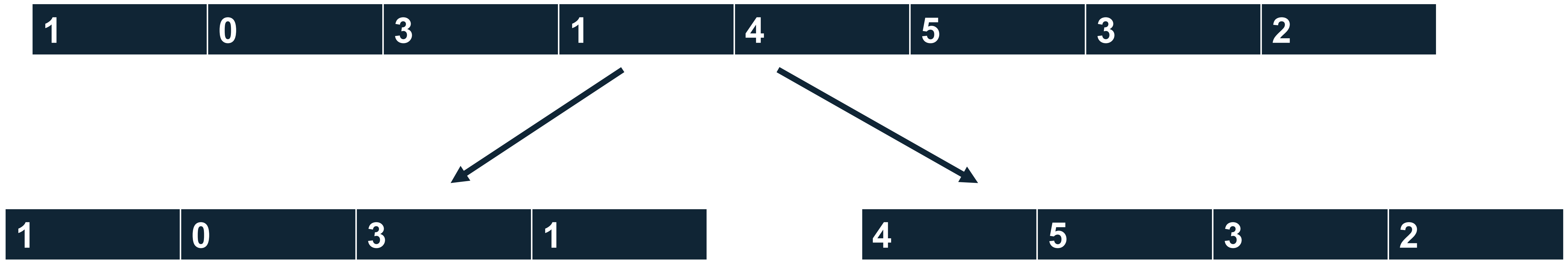
Merge sort

```
vector<int> MergeSort::sort(vector<int> array) {
     if(array.size() == 1) {
          return array;
     }

     // Sort left and right subarrays
     int mid = array.size()/2;
        …
     vector<int> sortedRightArray = sort(rightArray);
     vector<int> sortedLeftArray = sort(leftArray);

     // Merge left and right subarrays
     vector<int> result;
     int l=0, r=0;
     while (l < sortedLeftArray.size()
               && r < sortedRightArray.size()) {
          if(sortedLeftArray.at(l) < sortedRightArray.at(r)) {
               result.push_back(sortedLeftArray.at(l));
               l++;
          } else {…}}
          //
     // Add remaining elements from sortedLeftArray or sortedRightArray
     …
     return result;
}
```

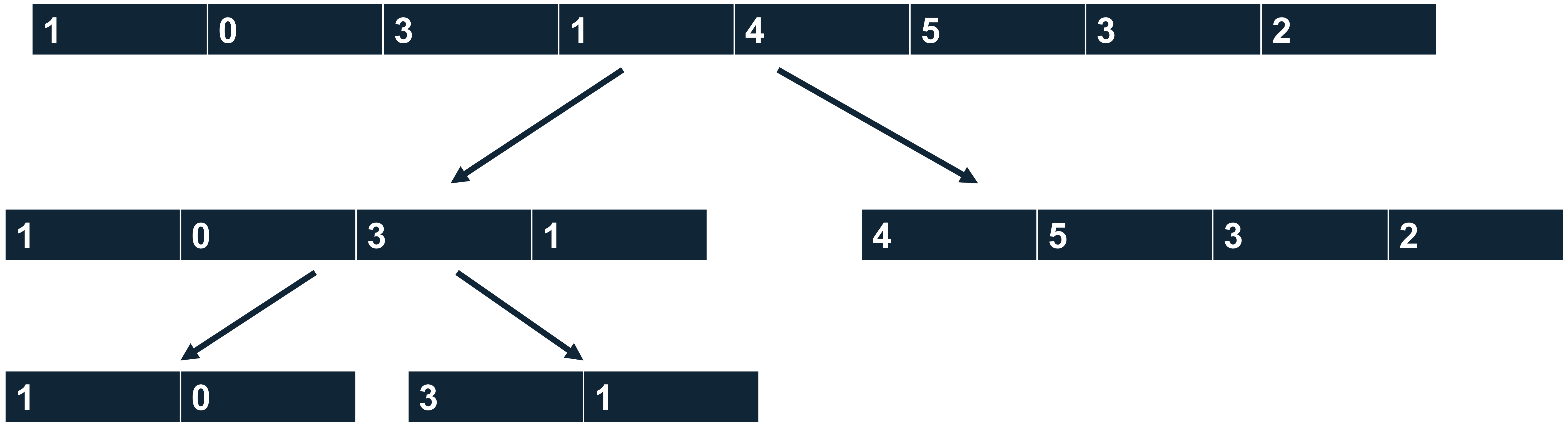
Merge sort

1	0	3	1	4	5	3	2

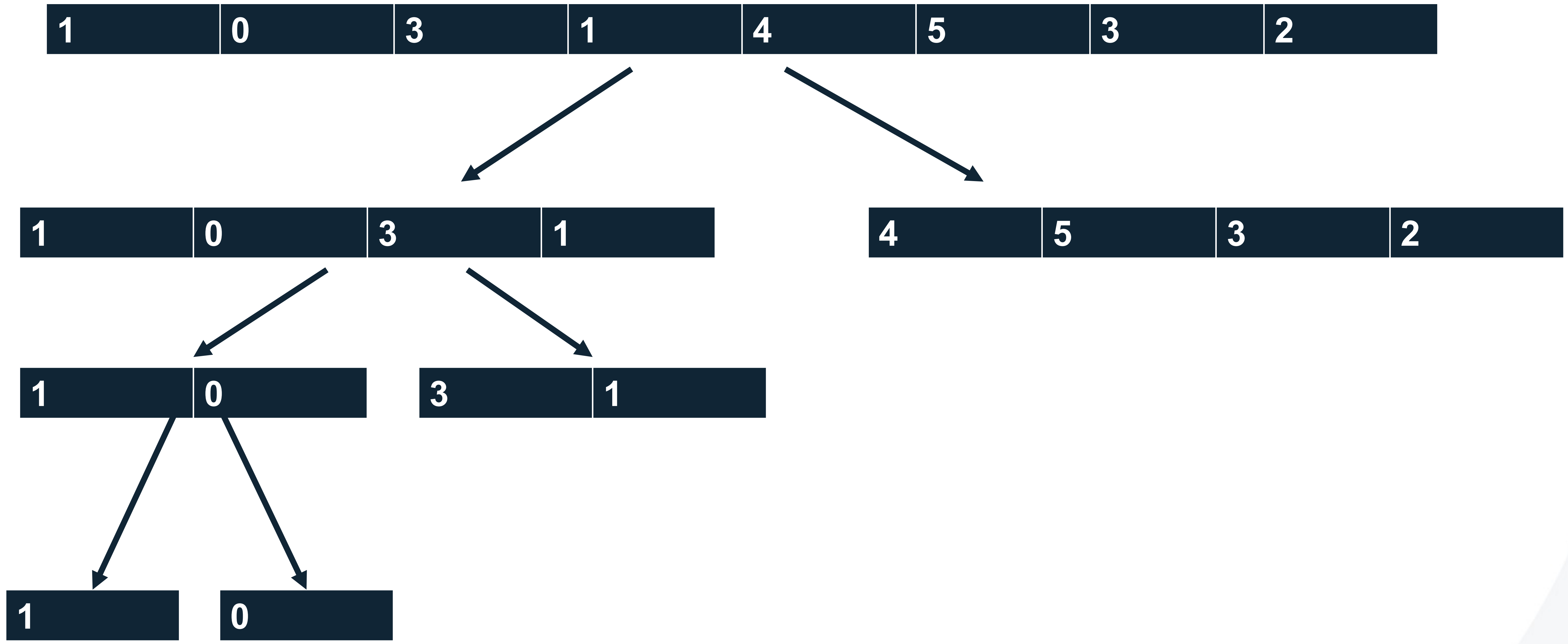
Merge sort



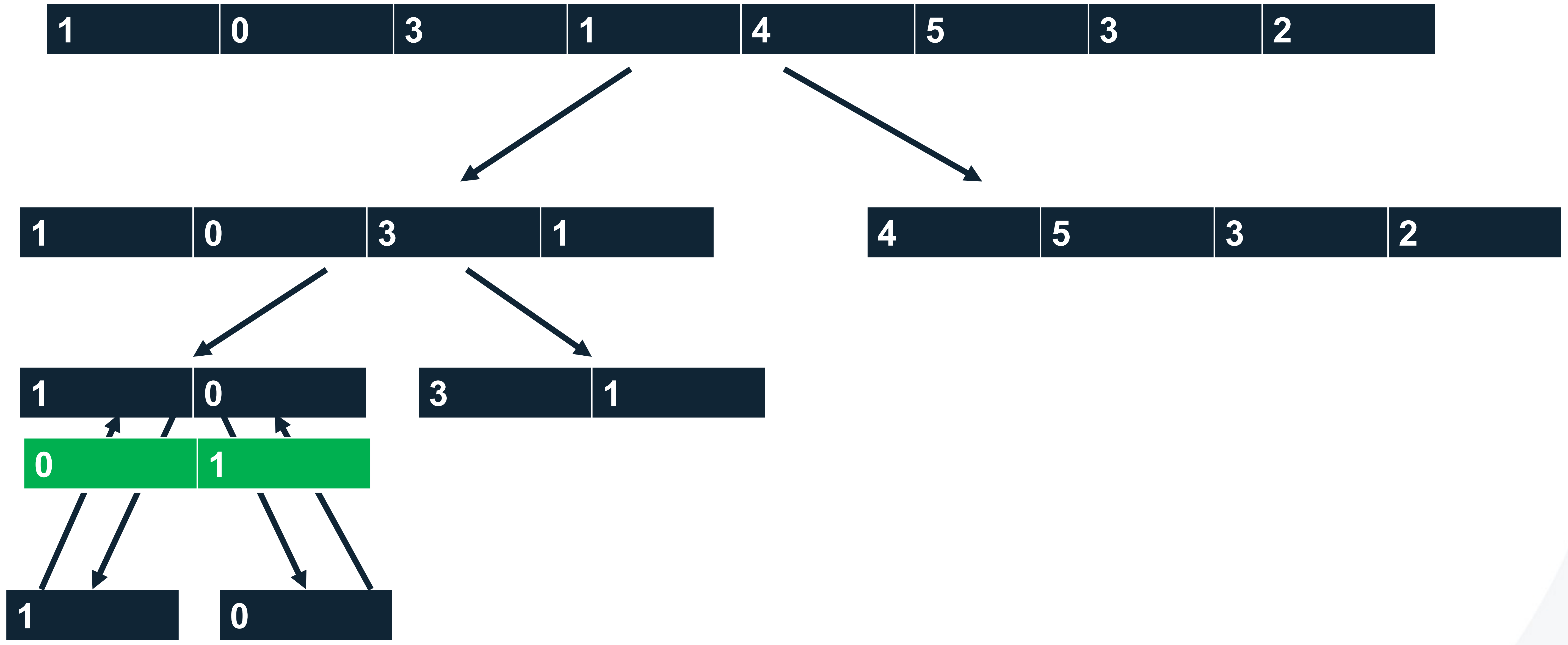
Merge sort



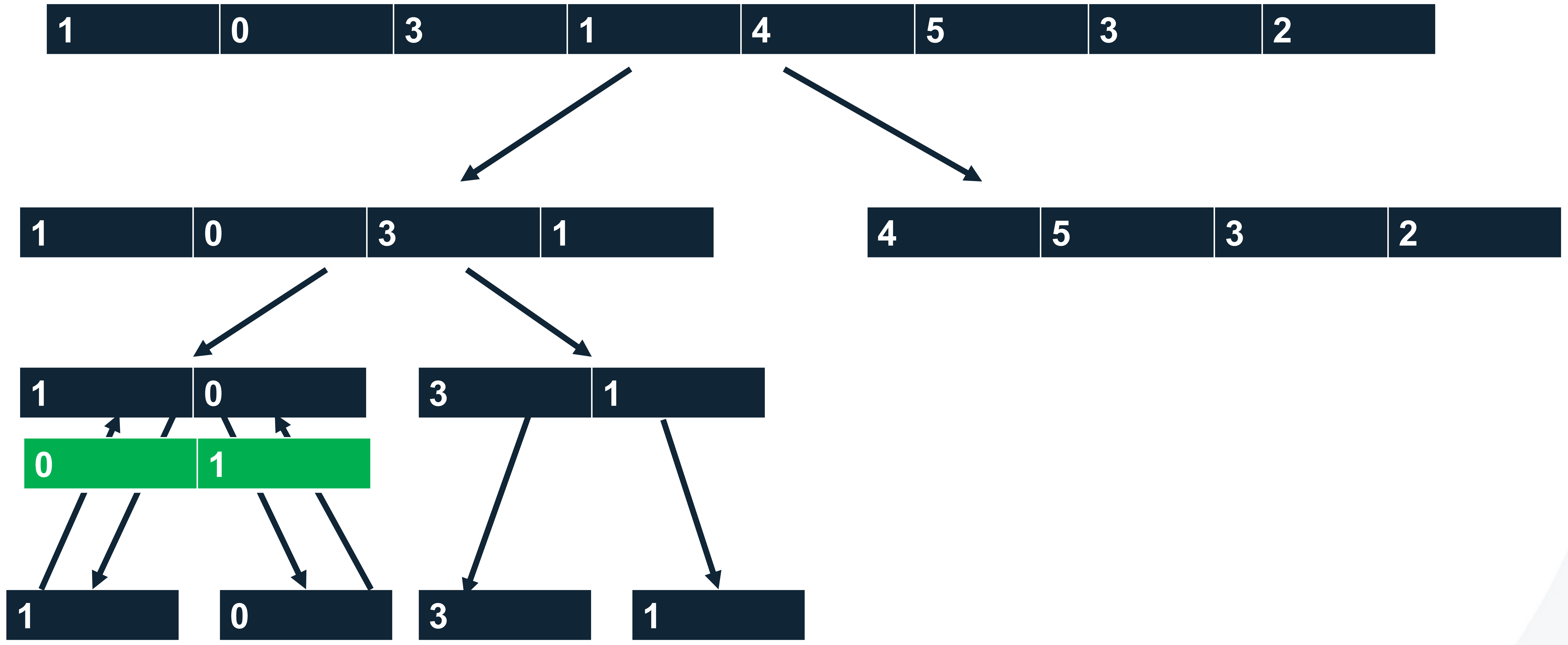
Merge sort



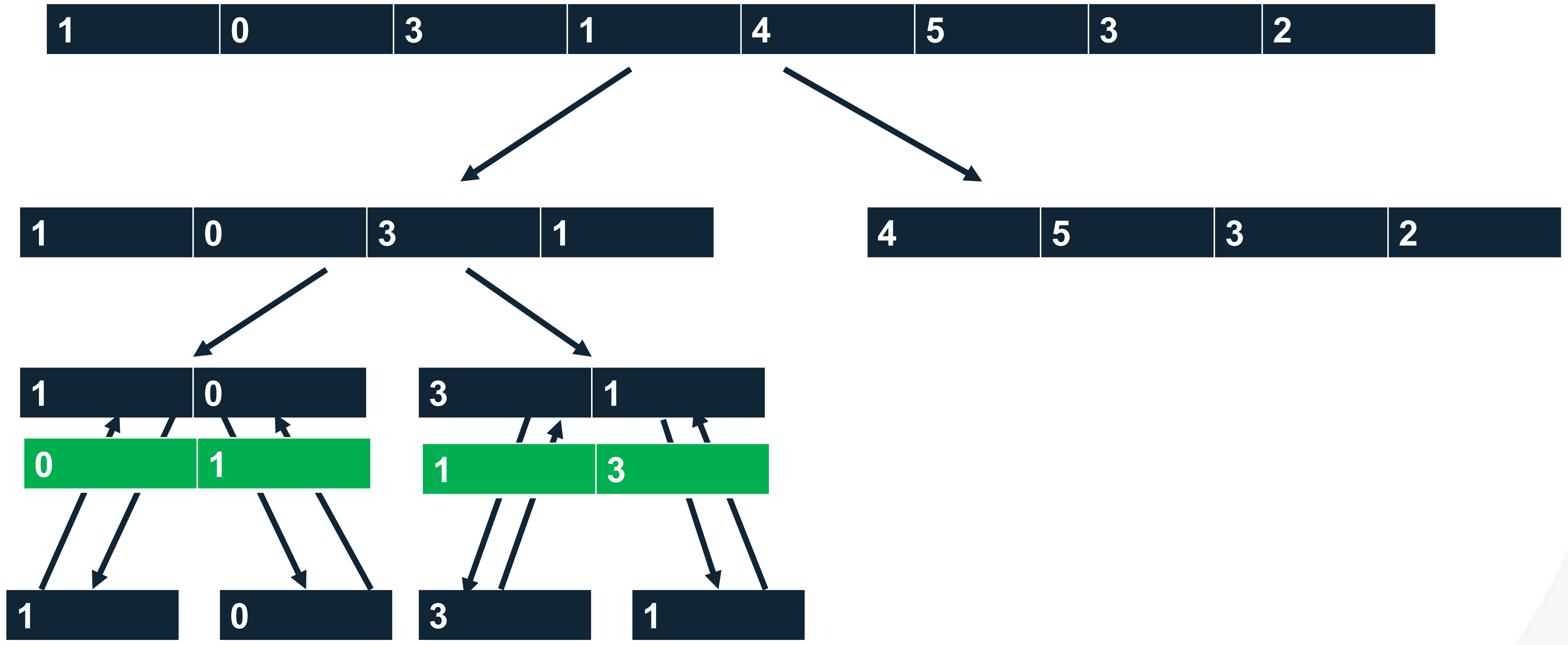
Merge sort



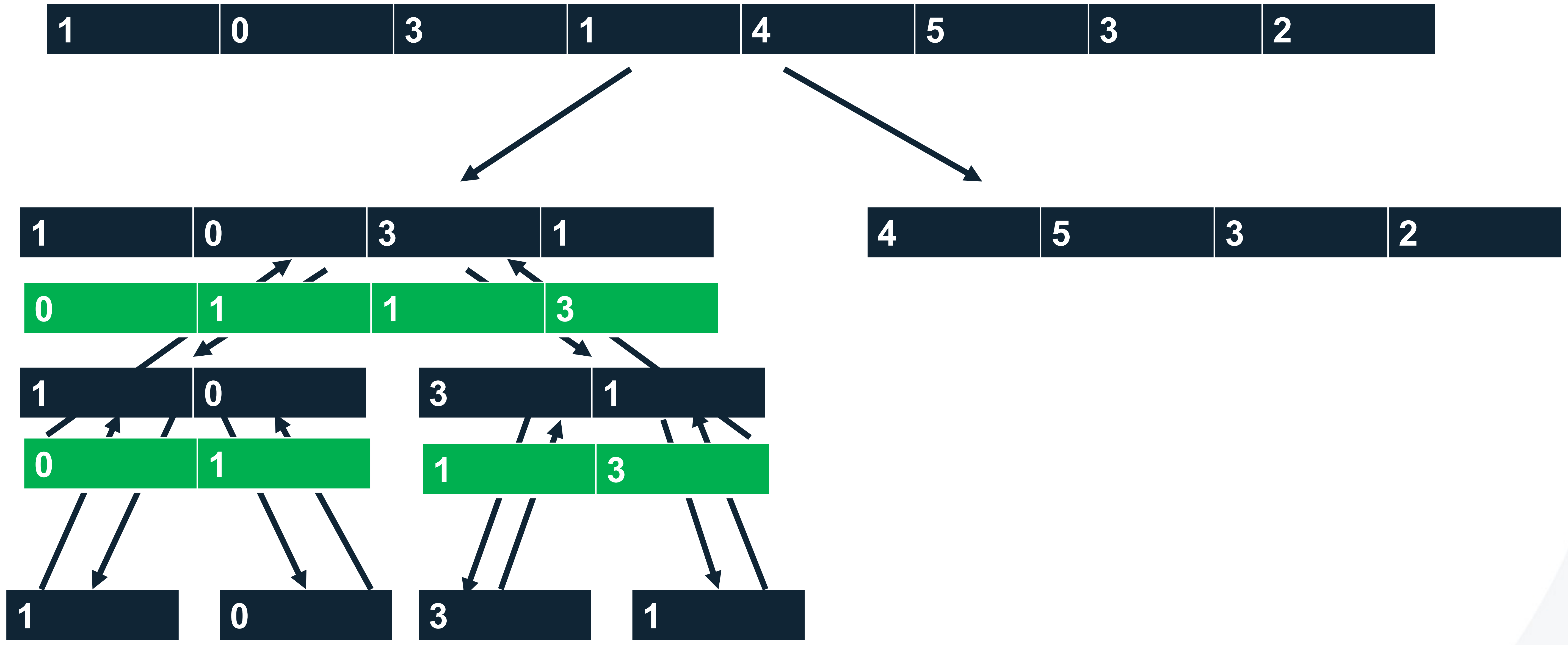
Merge sort



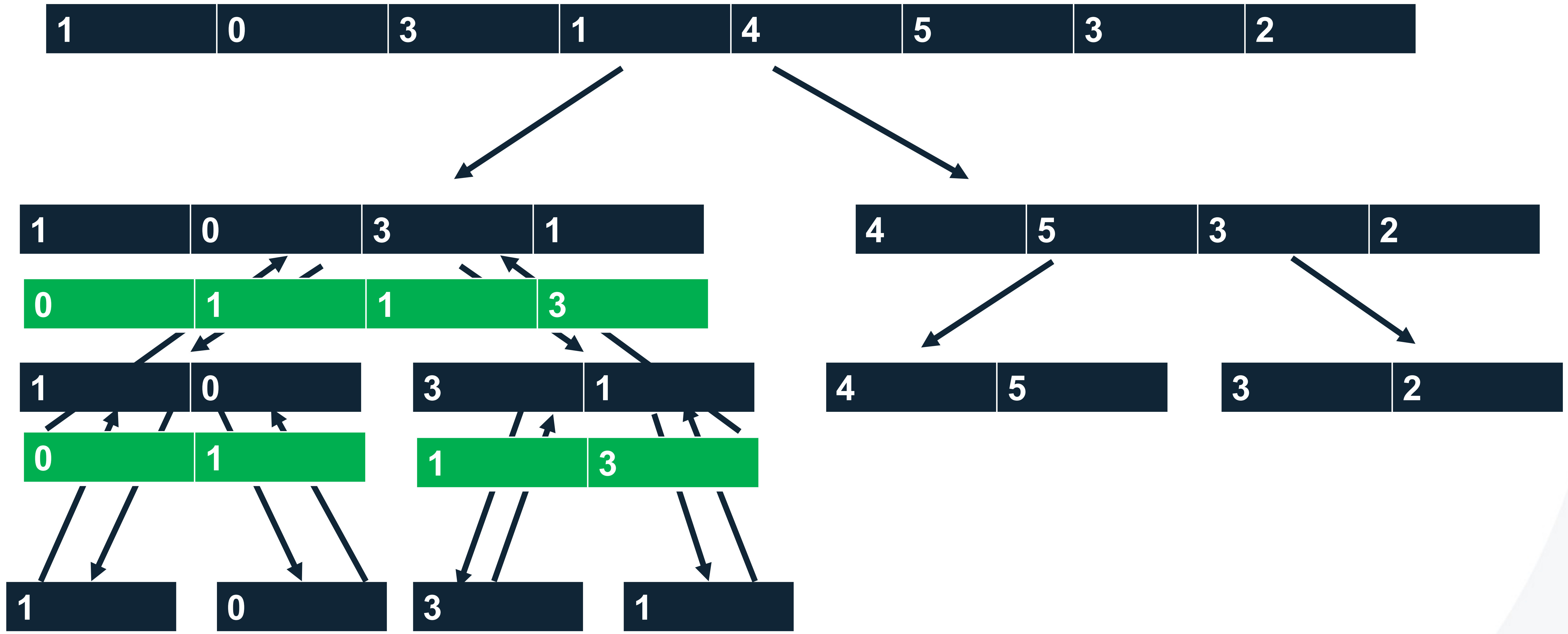
Merge sort



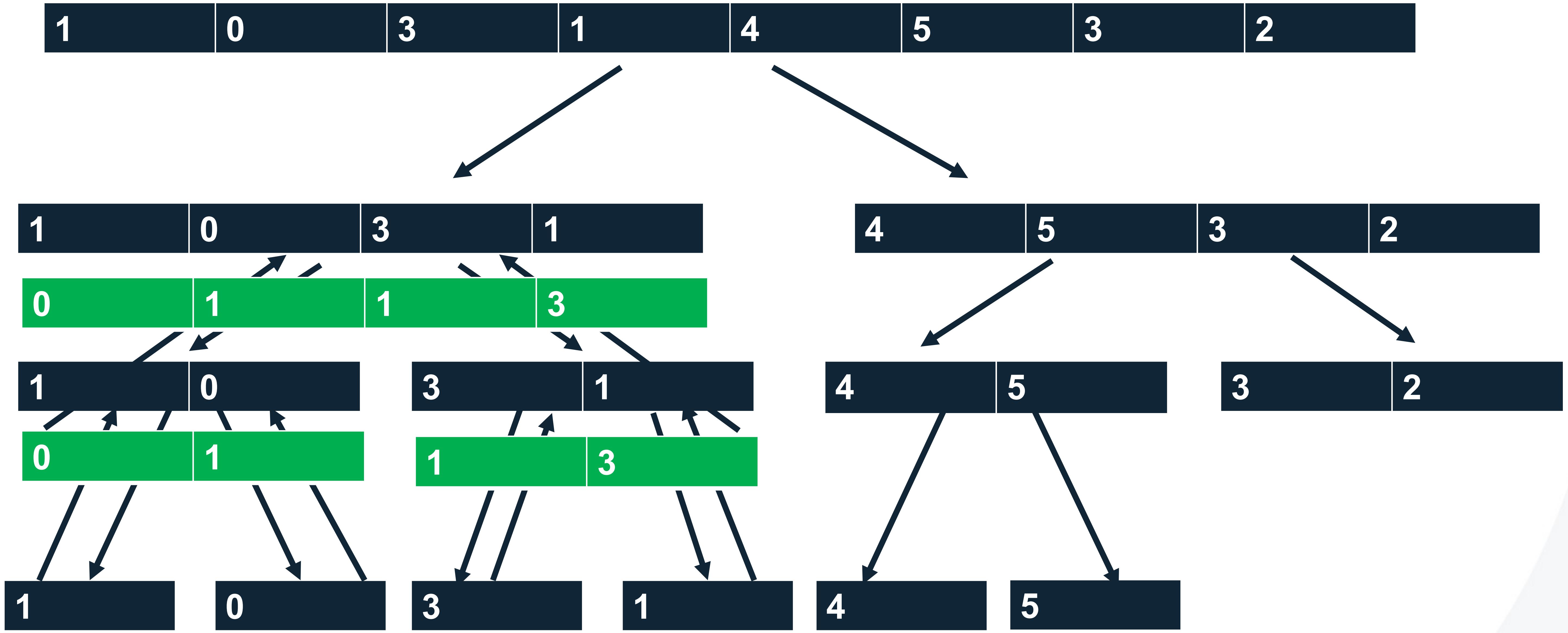
Merge sort



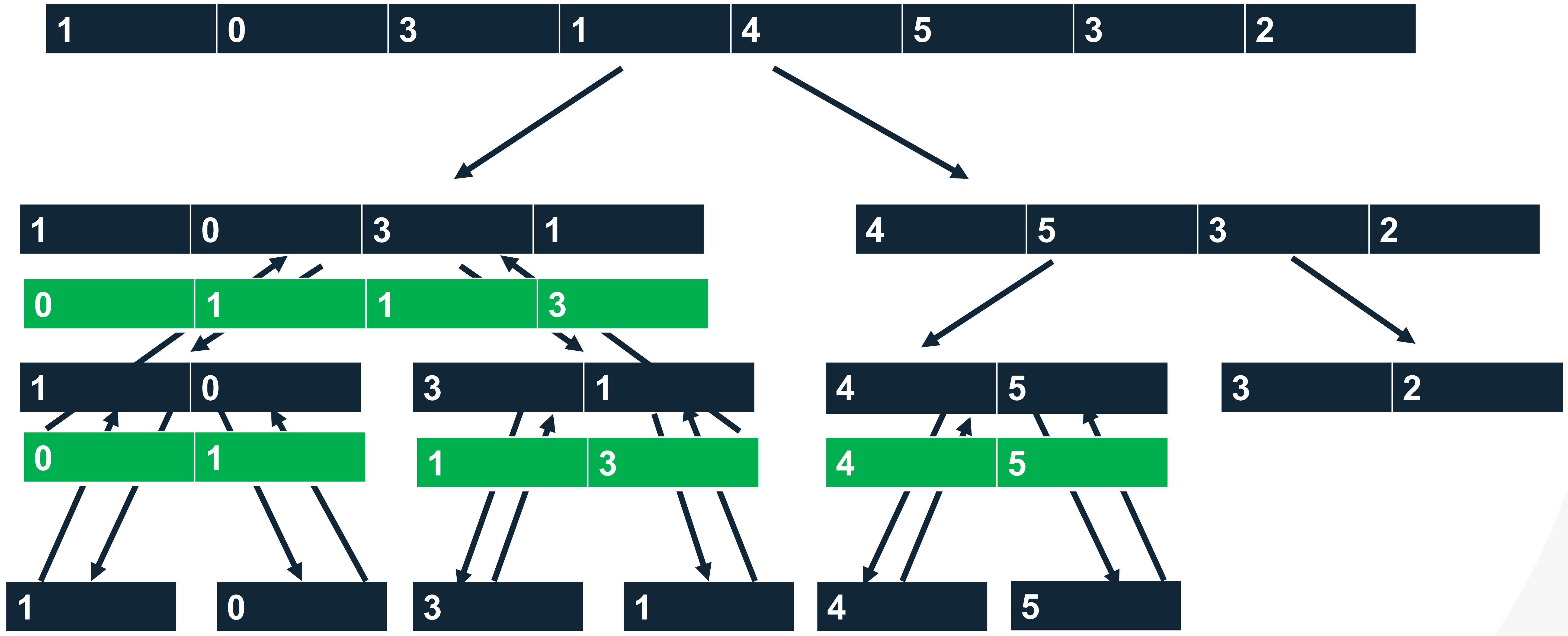
Merge sort



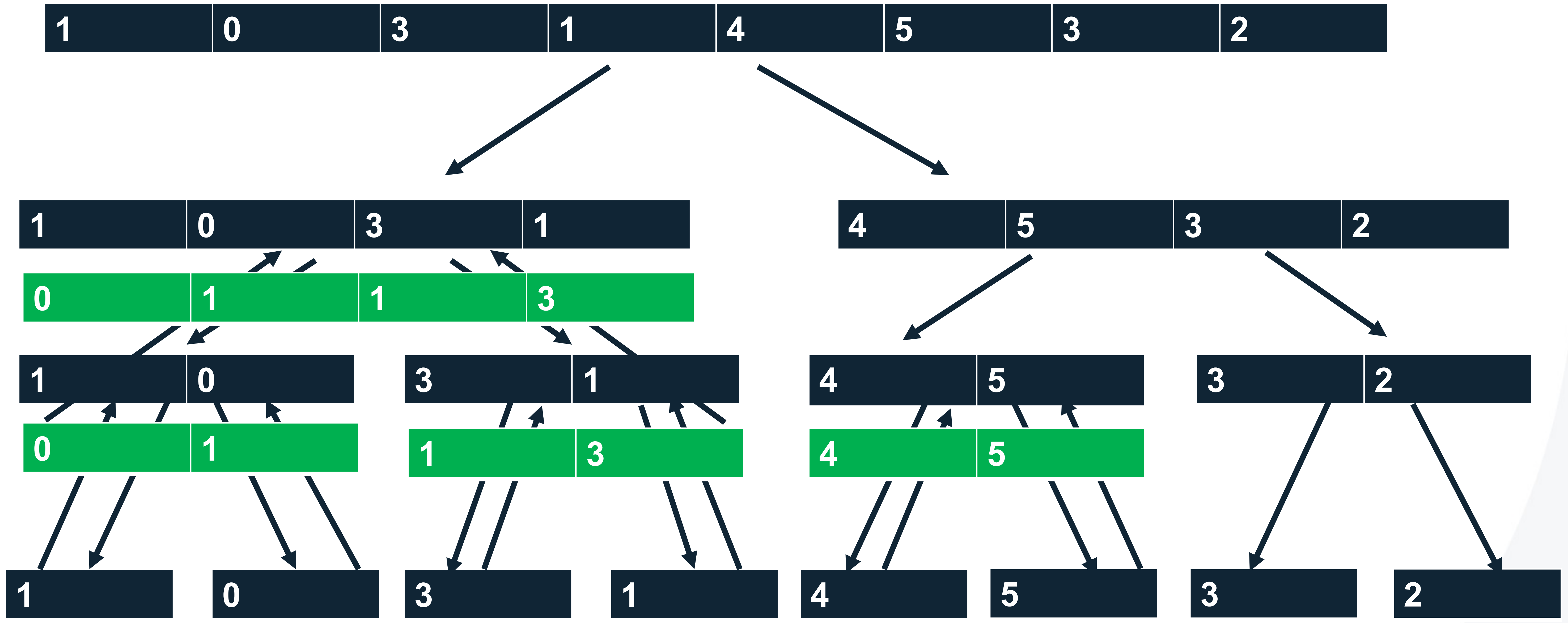
Merge sort



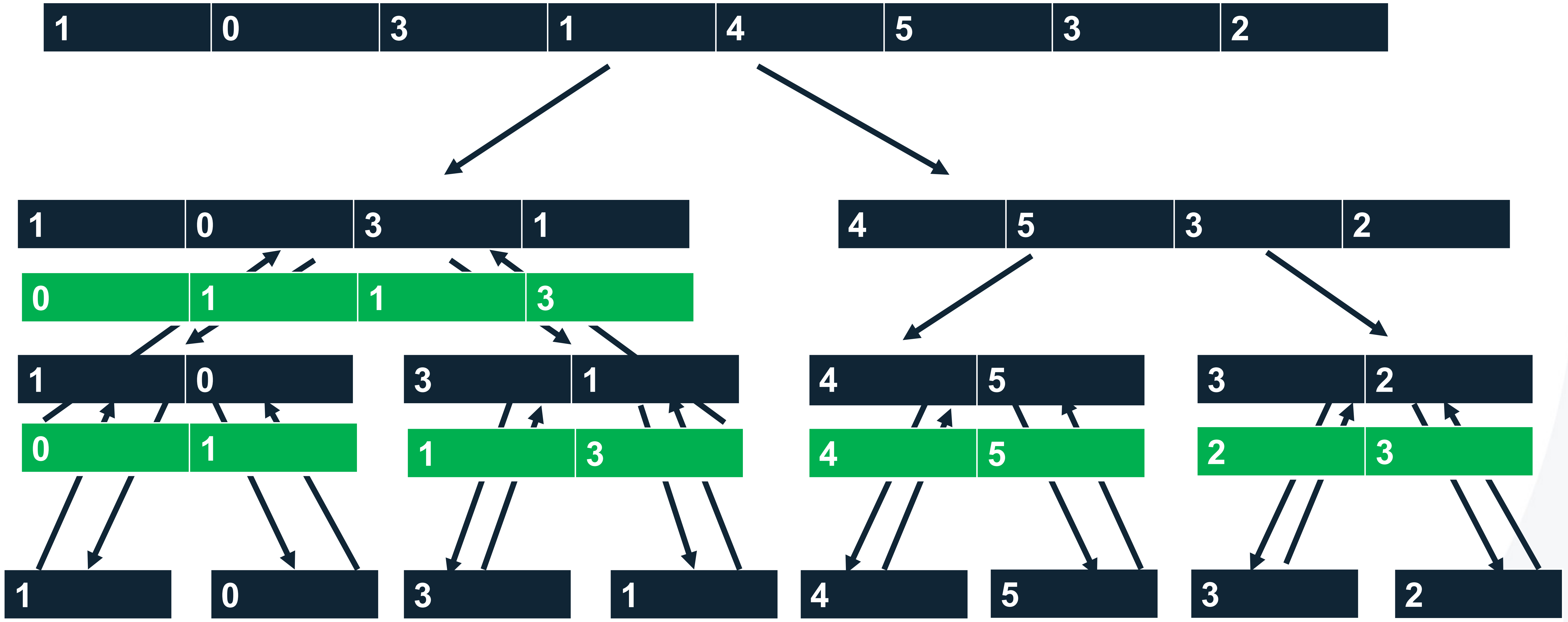
Merge sort



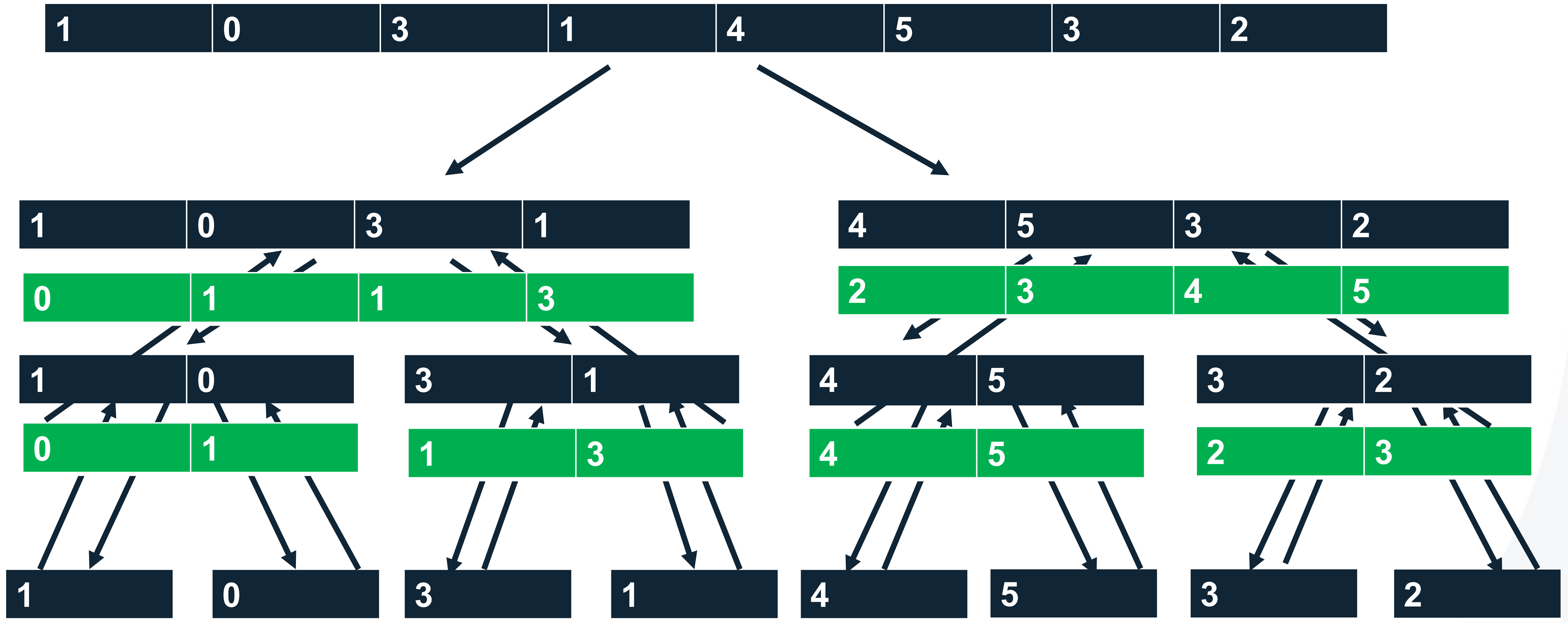
Merge sort



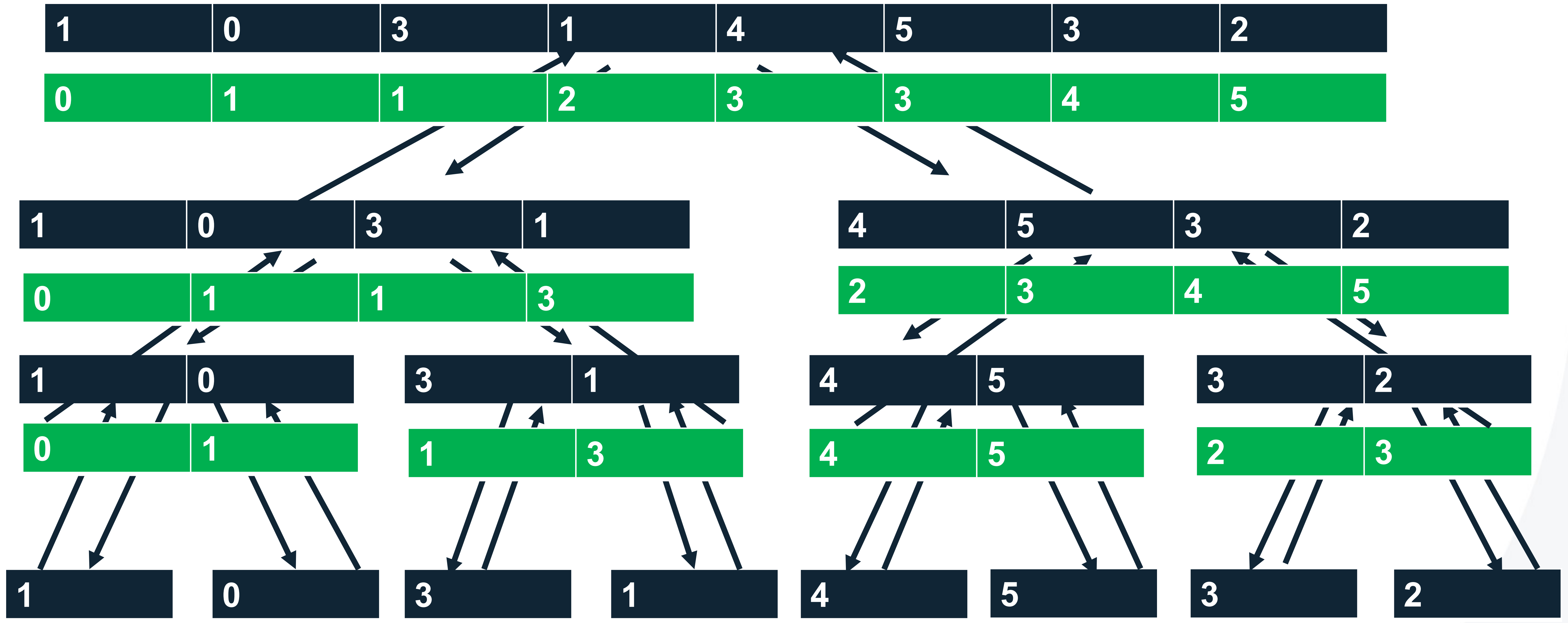
Merge sort



Merge sort



Merge sort



1 0 3 1 4 5 3 2
0 1 1 2 3 3 4 5
1 0 3 1
0 1 1 3
4 5 3 2
2 3 4 5
1 0
0 1
3 1
1 3
4 5
4 5
3 2
2 3
1
0
3
1
4
5
3
2

Merge sort

Applying master theorem with a=2, b=2 and d=1
(the complexity of merging two subarrays is $\Theta(n)$),
we get time complexity $\Theta(n\log(n))$ in best, worst and average cases.