Bubble sort
Insertion sort
Selection sort
} $O(n^2)$

Merge Sort $O(n \log n)$

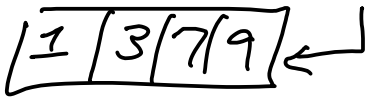
Recursive sort

Algorithm calls itself (on smaller data)

(mathematical induction)

$n = sA + sB$

Merge is O(n)



Function MERGE(A, B, sA, sB)

$O(1)+O(1)+O(1)+O(n)+O(n)+O(n) = O(n)$

O(1) R ← new array of size sA+sB

O(1) left ← 0

O(1) right ← 0

O(n) {

While left<sA and right<sB:

O(1) If A[left] < B[right]:

O(1) R[left+right] ← A[left]

O(1) left ← left+1

Else:

O(1) R[left+right] ← B[right]

O(1) right ← right+1

End If

End While

}

While left<sA:

O(n) R[left+right] ← A[left]

left ← left+1

EndWhile

While right<sB:

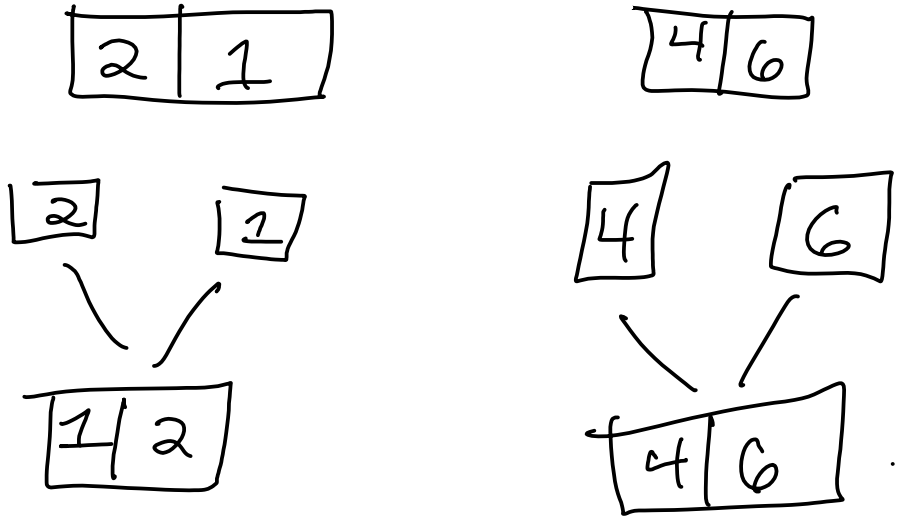
O(n) R[left+right] ← B[right]

right ← right+1

End While

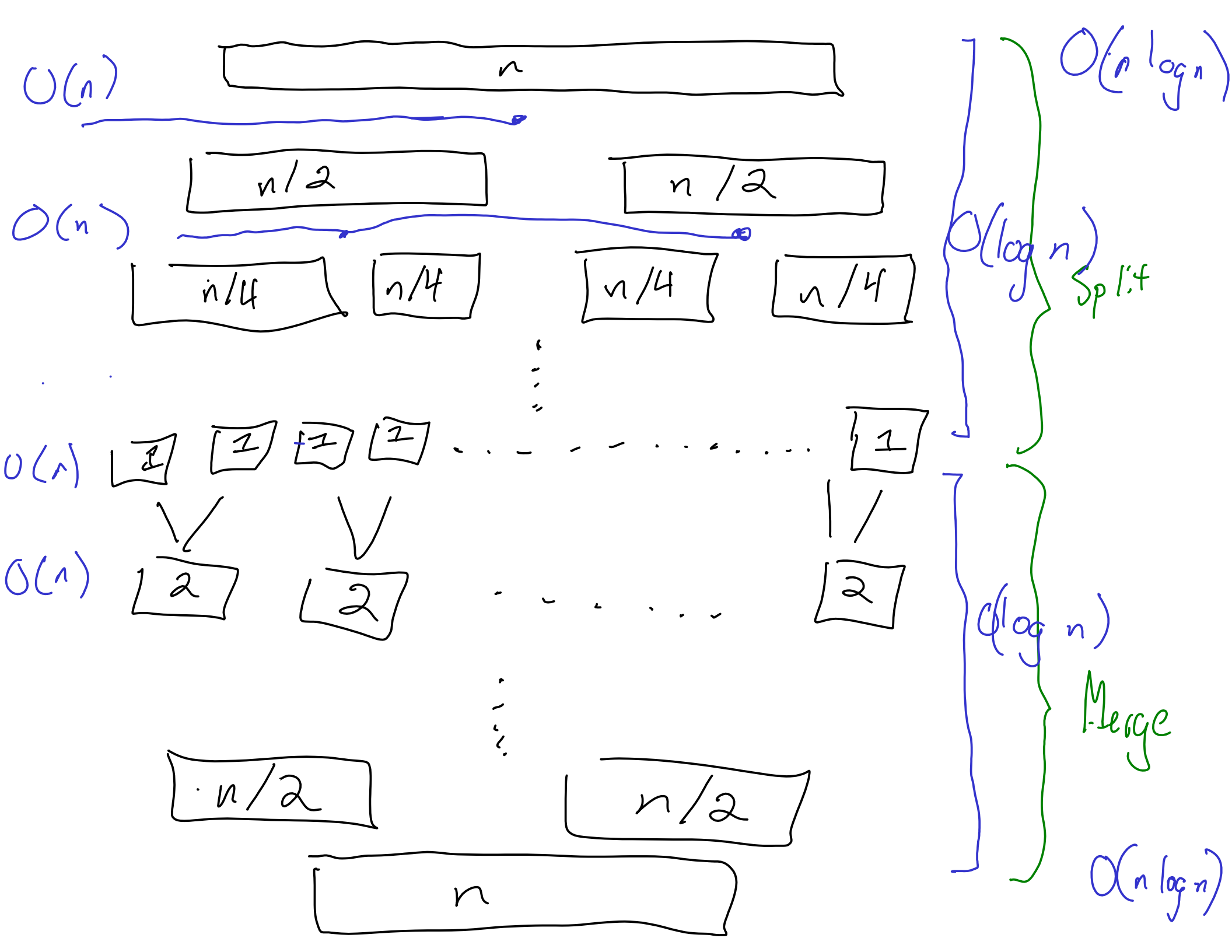
Return R

End Function



Function SPLIT(A, n):
B ← new array, size $\lceil n/2 \rceil$
C ← new array size $\lfloor n/2 \rfloor$
copy first $\lceil n/2 \rceil$ from A into B
copy last $\lfloor n/2 \rfloor$ from A into C
Return B, C
EndFunction

Function MERGESORT(A, n):
O(1) { If n<2:
Return A !!
EndIf }
O(n) { ? ? } B, C ← SPLIT(A, n)
B' ← MERGE SORT(B, $\lceil n/2 \rceil$)
C' ← MERGE SORT(C, $\lfloor n/2 \rfloor$)
O(n) A' ← MERGE(B', C', $\lceil n/2 \rceil$, $\lfloor n/2 \rfloor$)
Return A'
EndFunction


O(n)
n
O(n log n)
n/2
n/2
O(n)
O(log n)
Split
n/4
n/4
n/4
n/4
1
1
1
1
1
O(1)
2
2
2
O(1)
O(log n)
Merge
n/2
n/2
n
O(n log n)