CompSci 201, L15: Mergesort and Binary Search

Interested in computer science research and graduate school?

- Duke CS Student Union is hosting a panel of students applying to graduate school and currently in graduate school.
- Pizza and wings provided!
- 7-8:30 pm **tonight**, Monday 3/6.
- Social Sciences 136.
- RSVP at duke.is/2sv2y

H""'+."*$,$"#&:(I+,)"1(.-

- !"#$%&'(")#$%'*+,
  - !"#$%&'(!)*(+,-(./012%3(/04'(5"#$%&'6(37%
- -.#)./#$%'*+0
  - -!8(9(.4#"'01:(5"#;/%<46(37%
- 123#$%'*+45
  - =0//(#7'('>"#$%!%$&%'()*+'$%(',%-
  - .*(#"$)+$$"*/0(%/1*-($2"'/3(4'%567
- -.#)./#$%'*+66
  - ?03'%"<(@A(/012%3(/04'('>"#3B<(A#('"%%4

Today's Agenda

1. Finish mergesort
2. Binary search algorithm
3. Introduce Stack, Queue, PriorityQueue

Mergesort recursive method

- Should sort everything in **`ar`** starting at index **`l`** and up to (but not including) index **`r`**.

```
34    public static void mergeHelper(int[] ar, int l, int r) {
35        int diff = r-l;
36        if (diff < 2) { return; }
37        int mid = l + diff/2;
38        mergeHelper(ar, l, mid);
39        mergeHelper(ar, mid, r);
40        merge(ar, l, mid, r);
41    }
```

Base case, if 0 or 1 elements, nothing to do

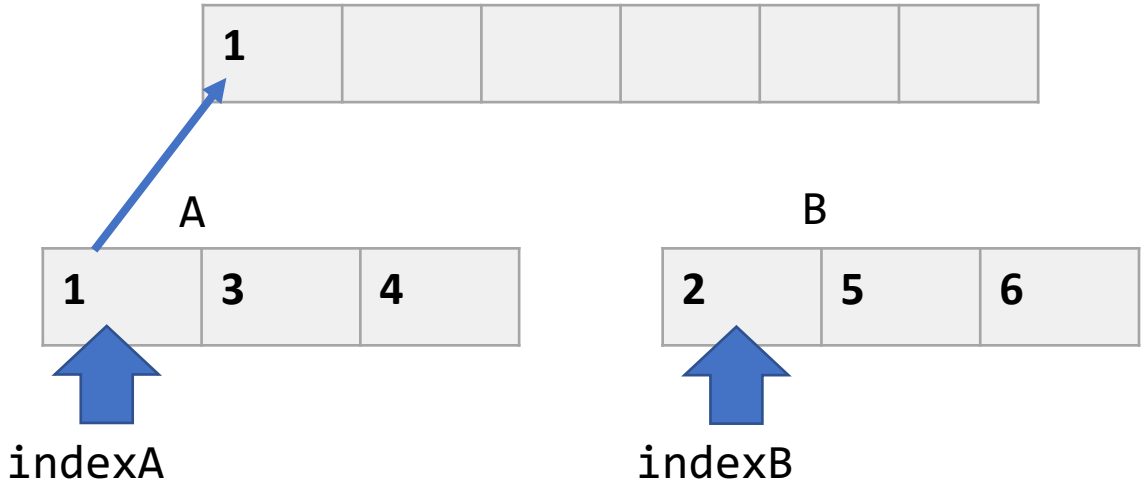
Recursively sort 1st half

Recursively sort 2nd half

Merge the 2 sorted parts

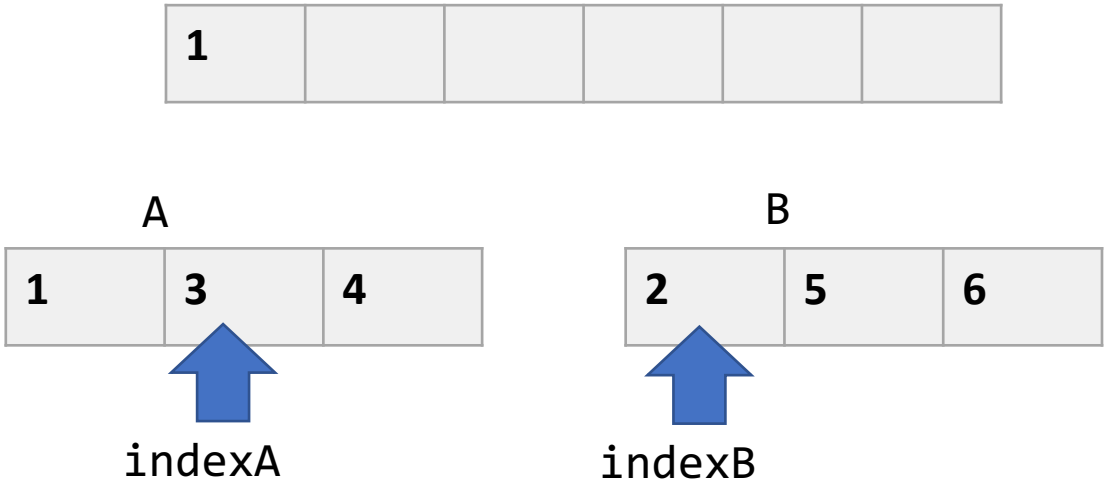
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



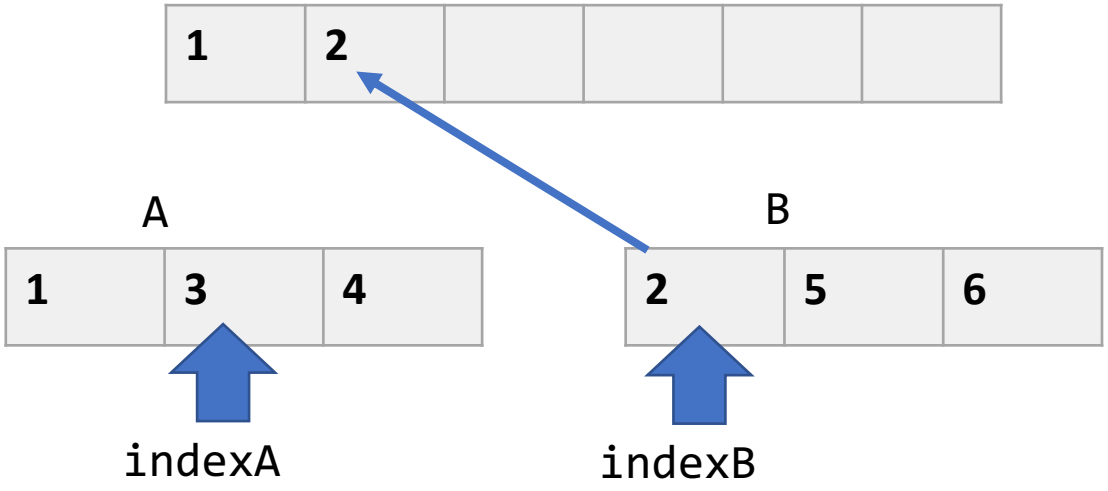
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



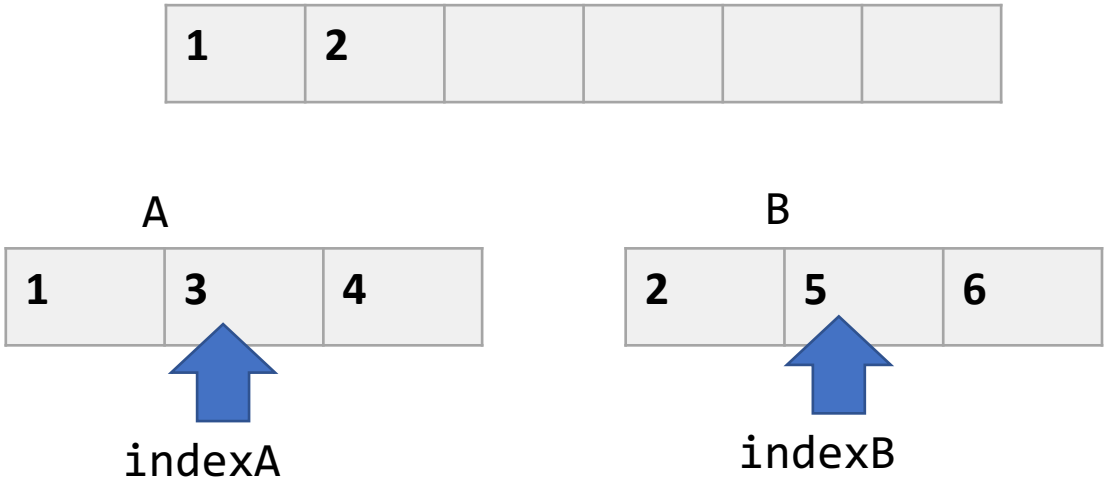
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



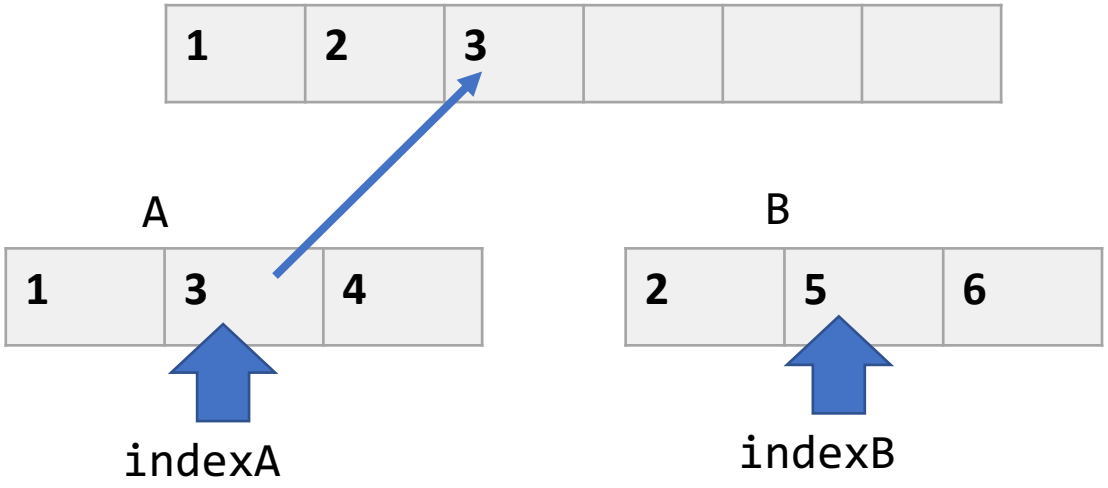
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



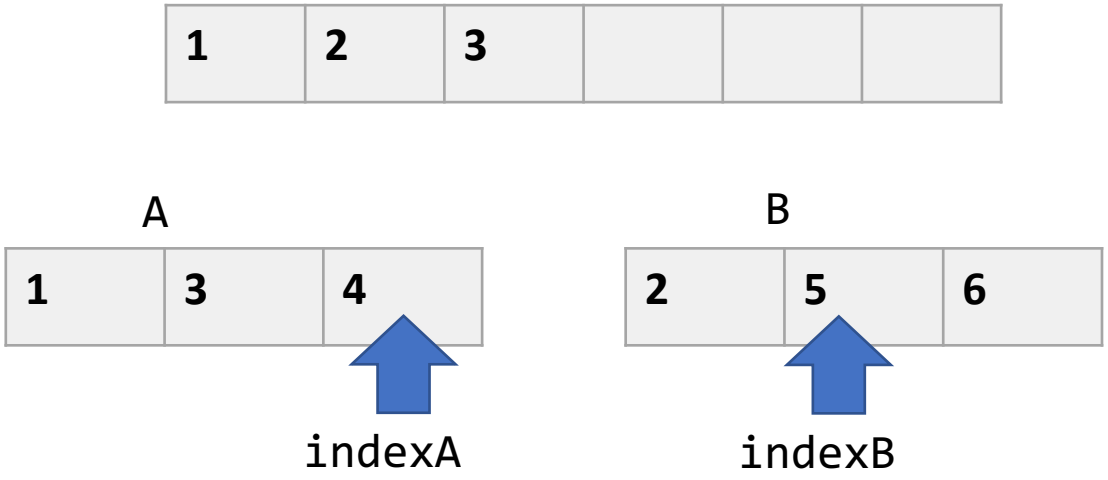
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



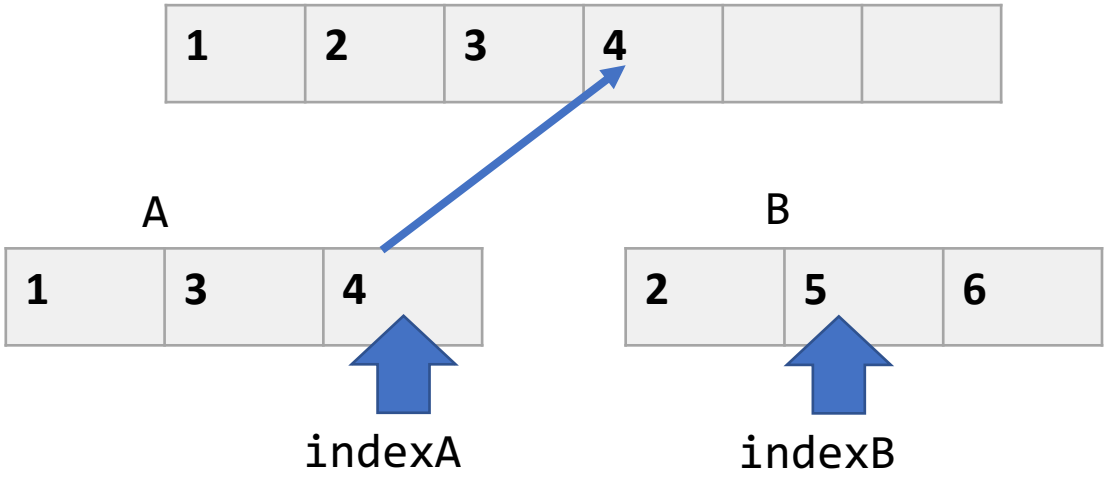
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



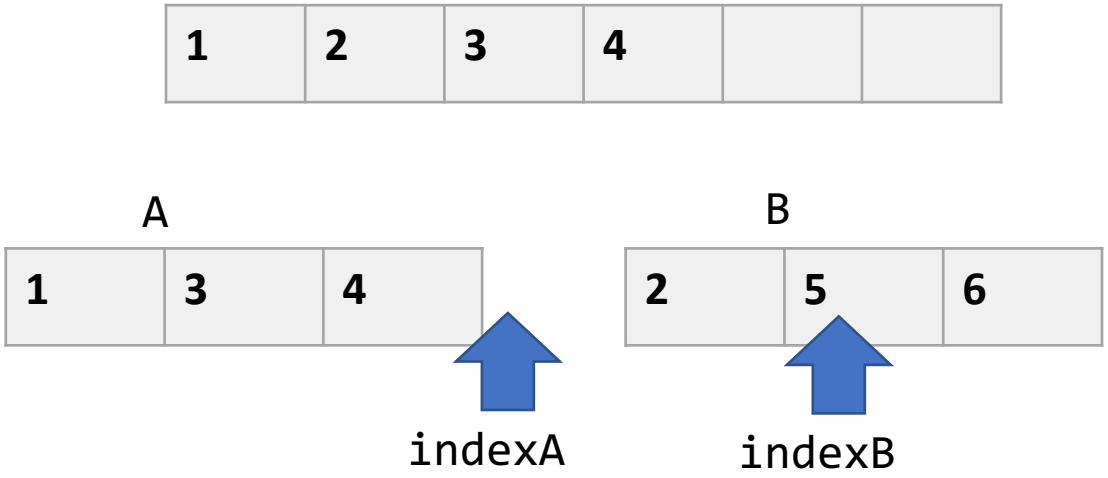
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



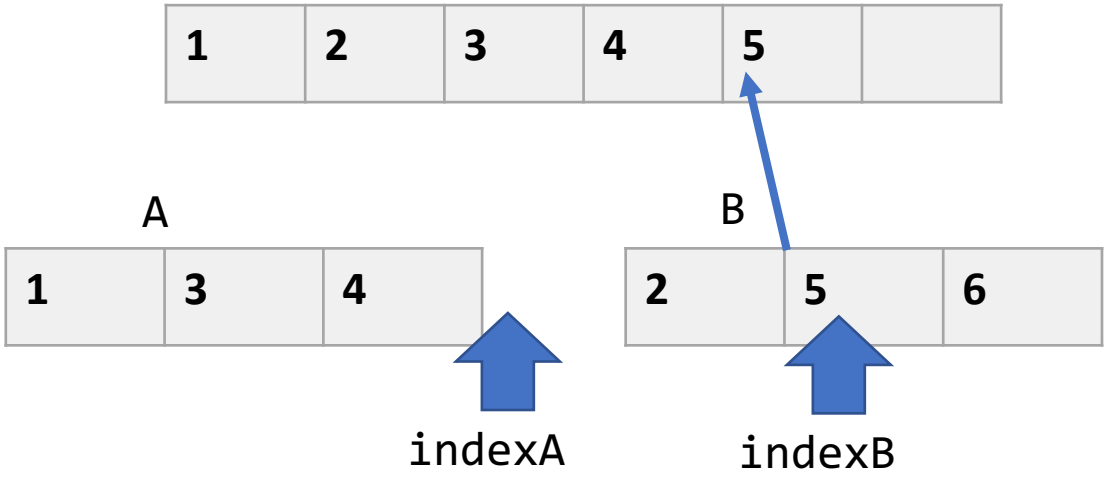
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



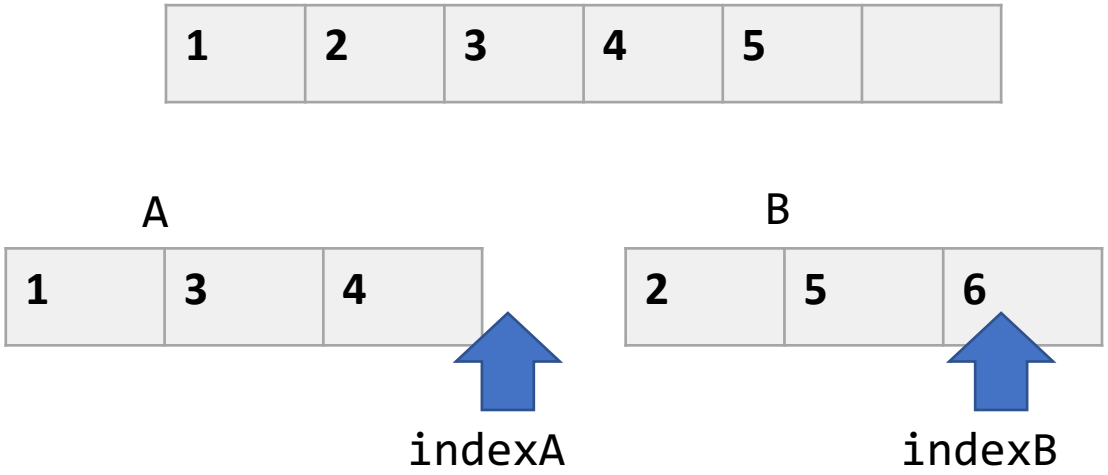
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



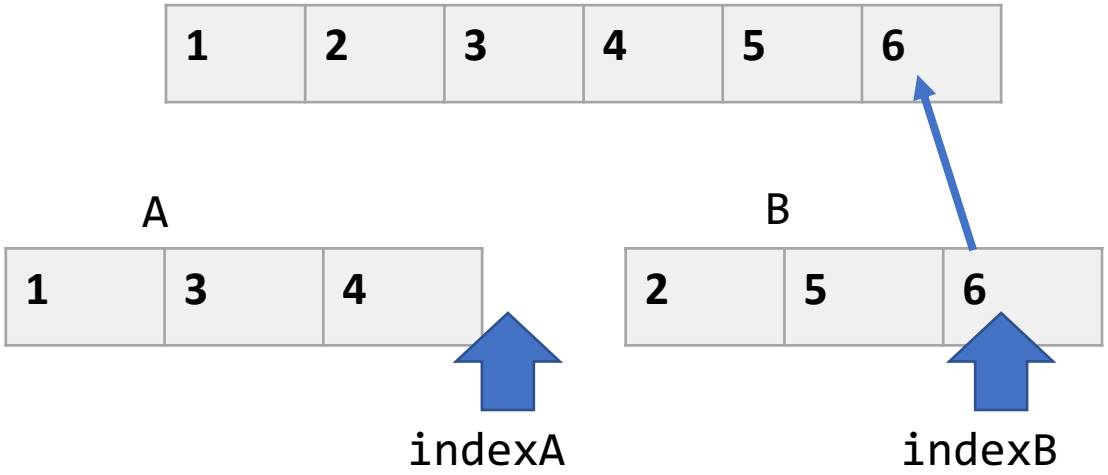
Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



Merge method

- Given two sorted arrays, A and B, want to merge them into one with all values from both.
- Need to keep track of **two** indices, `indexA` and `indexB`.



Merge method initialization

- Should merge `ar[l...mid]` and `ar[mid…r]`

```
43   public static void merge(int[] ar, int l, int mid, int r) {
44       int[] sorted = new int[r-l];
45       int sDex=0; int lDex=l; int rDex=mid;
```

- Need a new array `sorted` to put the merged results in, will copy back over `ar` later.
- Keeping track of 3 indices:
  - `sDex` = where we are in the `sorted` array
  - `lDex` = where we are in `ar[l…mid]`
  - `rDex` = where we are in `ar[mid…r]`

Merge method loop

```
46    while (lDex < mid && rDex < r) {
47        if (ar[lDex] <= ar[rDex]) {
48            sorted[sDex] = ar[lDex];
49            lDex++;
50        }
51        else {
52            sorted[sDex] = ar[rDex];
53            rDex++;
54        }
55        sDex++;
56    }
```

While something left in `ar[l…mid]` and `ar[mid…r]`

Add the smaller element and increment its index.

Increment `sDex` in either case

Finishing the merge method

- Will finish with `ar[l…mid]` or `ar[mid…r]` first, need to copy the rest of the other.
- Then need to copy sorted back onto `ar[l…r]`

```
57    if (lDex == mid) { System.arraycopy(ar, rDex, sorted, sDex, r-rDex); }
58    else { System.arraycopy(ar, lDex, sorted, sDex, mid-lDex); }
59    System.arraycopy(sorted, srcPos: 0, ar, l, r-l);
```

- Code uses the System.arraycopy method:

```
public static void arraycopy(Object src,
                             int srcPos,
                             Object dest,
                             int destPos,
                             int length)
```

Copies an array from the specified source array, beginning at the specified position, to the specified position of the destination array. A subsequence of

WOTO

Go to duke.is/43syw

Not graded for correctness, just participation.

Try to answer *without* looking back at slides and notes.

But do talk to your neighbors!

2

Our implementation of merge sort used two methods shown below. Which method(s) are recursive? *

```
53    public static void mergeSort(int[] ar) {
54        mergeHelper(ar, l: 0, ar.length);
55    }
56
57    public static void mergeHelper(int[] ar, int l, int r) {
58        int diff = r-l;
59        if (diff < 2) { return; }
60        int mid = l + diff/2;
61        mergeHelper(ar, l, mid);
62        mergeHelper(ar, mid, r);
63        merge(ar, l, mid, r);
64    }
```

- ◯ mergeSort and mergeHelper are both recursive
- ◯ mergeSort is recursive
- ◯ mergeHelper is recursive
- ◯ Neither are recursive

3

What best explains the purpose of the mergeSort *wrapper* method? *

```
53    public static void mergeSort(int[] ar) {
54        mergeHelper(ar, l: 0, ar.length);
55    }
56
57    public static void mergeHelper(int[] ar, int l, int r) {
58        int diff = r-l;
59        if (diff < 2) { return; }
60        int mid = l + diff/2;
61        mergeHelper(ar, l, mid);
62        mergeHelper(ar, mid, r);
63        merge(ar, l, mid, r);
64    }
```

- ◯ It helps us make the algorithm more efficient
- ◯ It helps us make the algorithm more correct
- ◯ It helps us avoid having to use recursion
- ◯ It helps us to initialize the parameters to the recursive calls

4

Based on what you see, how many *levels* of recursion will there be to the merge sort algorithm? To be precise: For a given index k in the original array, in how many recursive calls will l <= k < r?

Answer in asymptotic notation as a function of N where N is the length of ar.
*

```
53    public static void mergeSort(int[] ar) {
54        mergeHelper(ar, l: 0, ar.length);
55    }
56
57    public static void mergeHelper(int[] ar, int l, int r) {
58        int diff = r-l;
59        if (diff < 2) { return; }
60        int mid = l + diff/2;
61        mergeHelper(ar, l, mid);
62        mergeHelper(ar, mid, r);
63        merge(ar, l, mid, r);
64    }
```

- ◯ O(1)
- ◯ O(log(N))
- ◯ O(N)
- ◯ O(N log(N))

5

Let N = r-l. What is the asymptotic runtime complexity of the merge method? The runtime complexity of the arraycopy method is linear in the number of elements it copies, which is the last parameter of the method. *

```
66  public static void merge(int[] ar, int l, int mid, int r) {
67      int[] sorted = new int[r-l];
68      int sDex=0; int lDex=l; int rDex=mid;
69      while (lDex < mid && rDex < r) {
70          if (ar[lDex] <= ar[rDex]) {
71              sorted[sDex] = ar[lDex];
72              lDex++;
73          }
74          else {
75              sorted[sDex] = ar[rDex];
76              rDex++;
77          }
78          sDex++;
79      }
80      if (lDex == mid) { System.arraycopy(ar, rDex, sorted, sDex, r-rDex); }
81      else { System.arraycopy(ar, lDex, sorted, sDex, mid-lDex); }
82      System.arraycopy(sorted, srcPos: 0, ar, l, r-l);
83  }
```

- ◯ O(1)
- ◯ O(log(N))
- ◯ O(N)
- ◯ O(N log(N))
- ◯ O(N^2)

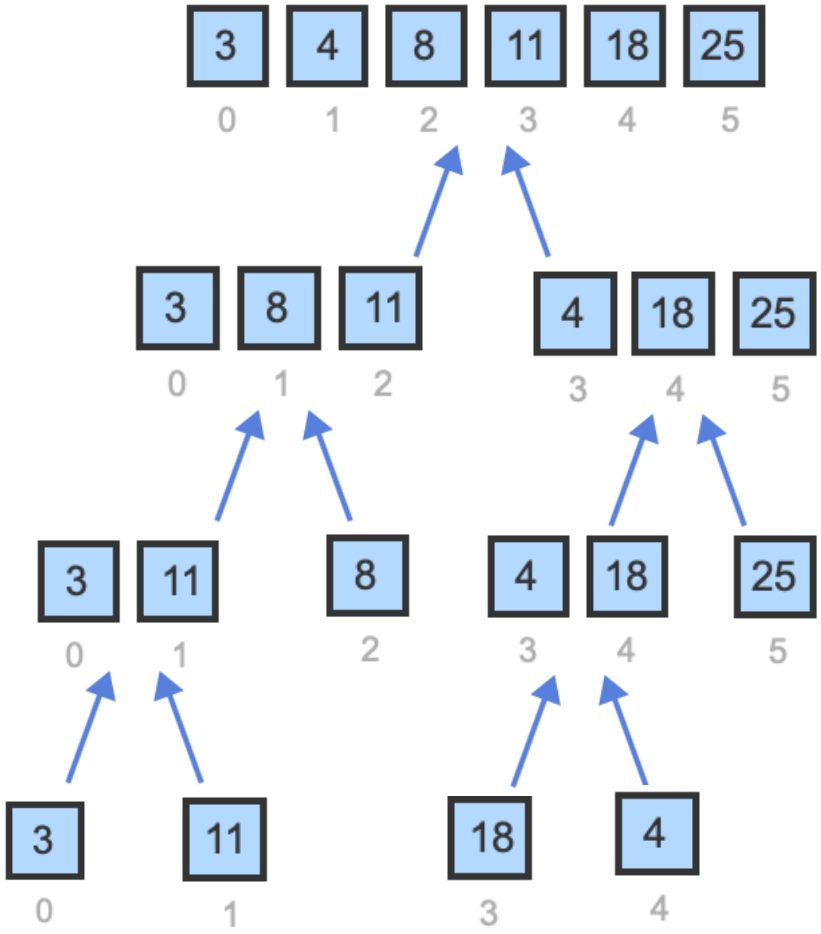
How fast is mergesort? Empirically…

N (thousands)	Selection sort (ms)	Insertion sort (ms)	Merge sort (ms)	Java.util Arrays.sort (ms)
10k	22	40	1	2
30k	168	334	2	2
90k	1481	967	7	6
270k	13175	8716	22	14

Looks linear but not quite:
O(N log(N)) is *nearly linear*.

Why mergesort is O(Nlog(N)), intuition

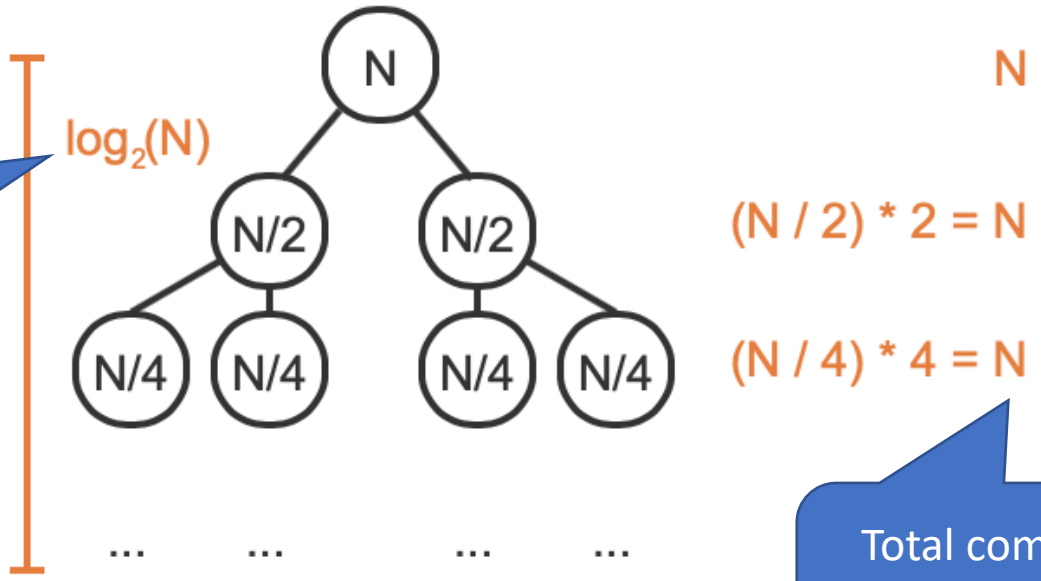
- Halves at each level, so just O(log(N)) levels.
- If we can do all of the merges at each level in O(N) time?
- Overall O(Nlog(N)).



Recursion tree

$$T(N) = N + T(N / 2) + T(N / 2)$$

Depth of the recursion tree: Number of recursive calls before base case.



Total complexity of each level across all of the recursive calls.

$$T(N) = O(N \log N)$$

Visualization from the Zybook

Analyzing Recursive Runtime

Develop a recurrence relation of the form

$$T(N) = a \cdot T\big(g(N)\big) + f(N)$$

Total runtime

Recursive call(s)

Non-recursive runtime

Where:

- $T(N)$ - runtime of method with input size *N*
- a is the number of recursive calls
- $g(N)$ - how input size decreases on each recursive call
- $f(N)$ - runtime of non-recursive code on input size N

Analyzing Runtime of Recursive Reverse

```
3   public static ListNode reverse(ListNode list) {
4       if (list == null || list.next == null) {
5           return list;
6       }
7       ListNode reversedLast = list.next;
8       ListNode reversedFirst = reverse(list.next);
9       reversedLast.next = list;
10      list.next = null;
11      return reversedFirst;
12  }
```

$g(N) = N - 1$

$f(N) = O(1)$

$$T(N) = T(N-1) + O(1)$$

Solving Recurrence Relations

$$
\begin{aligned}
T(N) &= T(N-1) + 1 \\
&= (T(N-2) + 1) + 1 \\
&= T(N-3) + 3 \\
&\vdots \\
&= T(1) + N \\
&= O(N)
\end{aligned}
$$

Apply recurrence again to T(N-1)

Total runtime

And again, to T(N-2)

T(1) is base case, just O(1)

recurrence relations and expectations in 201

- In general, will **not** be asked to **solve** recurrence relations on exams (for later classes in theory).
- You **may** be asked to determine the recurrence relation of a given algorithm/code.

Recurrence	Algorithm	Solution
`T(n) = T(n/2) + O(1)`	binary search	`O(log n)`
`T(n) = T(n-1) + O(1)`	sequential search	`O(n)`
`T(n) = 2T(n/2) + O(1)`	tree traversal	`O(n)`
`T(n) = T(n/2) + O(n)`	qsort partition ,find k^{th}	`O(n)`
`T(n) = 2T(n/2) + O(n)`	mergesort, quicksort	`O(n log n)`
`T(n) = T(n-1) + O(n)`	selection or bubble sort	$O(n^2)$

Runtime complexity of mergesort?

Let N = `r-l`, the number of elements to sort

```
34    public static void mergeHelper(int[] ar, int l, int r) {
35        int diff = r-l;
36        if (diff < 2) { return; }
37        int mid = l + diff/2;
38        mergeHelper(ar, l, mid);
39        mergeHelper(ar, mid, r);
40        merge(ar, l, mid, r);
41    }
```

T(N) = …

T(N/2)+…

T(N/2)+…

O(N)

$$T(N) = 2T\left(\frac{N}{2}\right) + O(N) \rightarrow T(N) \text{ is } O\big(N \log(N)\big)$$

Binary Search

Binary Search

- Given a **sorted** `list` of N elements and a `target`, in just O(log(N)) time, return:
  - Index `i` such that `list.get(i)` equals `target`, or
  - -1 if `target` not in `list`

- Example:
  - If we search for ‘h’, should return 4
  - If we search for ‘c’, should return -1

value	‘a’	‘b’	‘d’	‘g’	‘h’	‘j’	‘k’	‘m’	‘p’
index	0	1	2	3	4	5	6	7	8

Java API Binary Search

`Arrays.binarySearch` (for arrays) and `Collections.binarySearch` (for Lists).

```
String[] ar = {"ape", "bird", "cat", "dog", "elephant",
"ferret", "gecko", "hippo"};

int index = Arrays.binarySearch(ar, "cat");
```

Returns 2

Careful, does not check for you if list is sorted!

```
String[] ar = {"cat", "ape", "bird",…

int index = Arrays.binarySearch(ar, "cat");
```

Returns -4

Java API Binary Search with Comparator

Can pass a comparator `comp`, in which case:

1. Array/List should be sorted by that `comp`, and
2. Want an index `i` with `i`'th element e_i has `comp.compare(`e_i`, target)==0`.

Sorted by length

```
[ape, cat, dog, bird, gecko, hippo, ferret, elephant]
```

```
Comparator<String> comp =
    Comparator.comparing(String::length);

index = Arrays.binarySearch(ar, "dog", comp);
```

Returns 1. `comp.compare("cat", "dog")==0`

How is Binary Search O(log(N))?

- How to find something in a list of N elements without looping over the list?
- Let `low` (initially 0) and `high` (initially N-1) mark the limits of the active search space.
- Want to cut down the search space by half at each step:

N
N/2
N/4
N/8
…
1

log_2(N) steps!

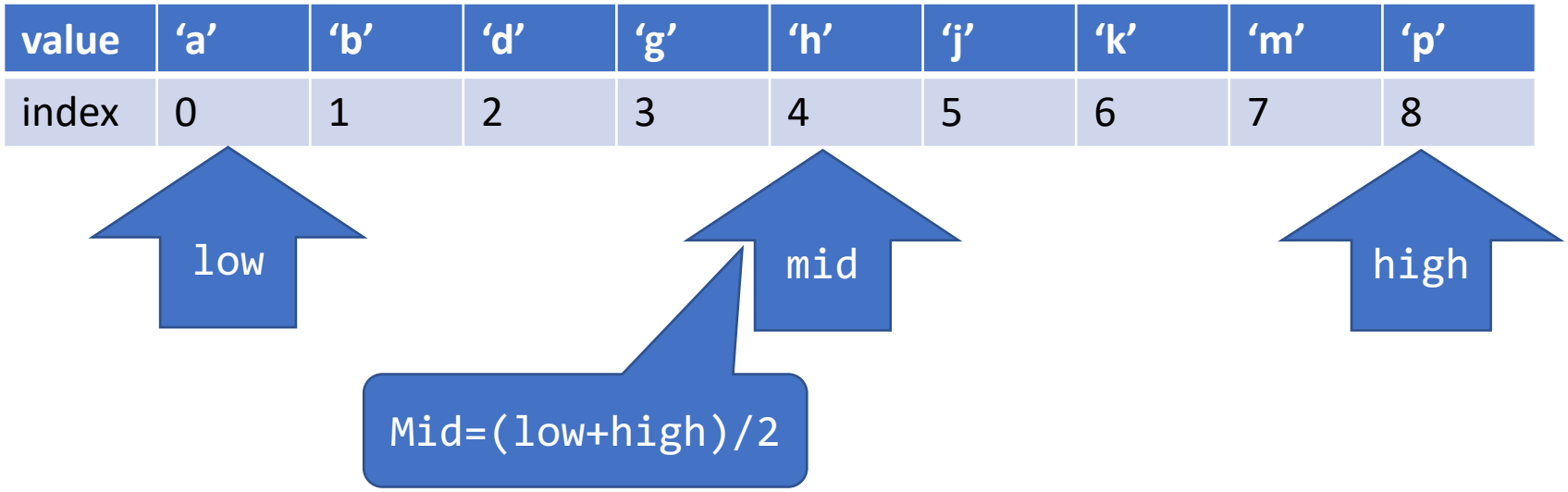
value	'a'	'b'	'd'	'g'	'h'	'j'	'k'	'm'	'p'
index	0	1	2	3	4	5	6	7	8

`low`

`high`

Binary Search in Pictures

- Searching for ‘d’ in

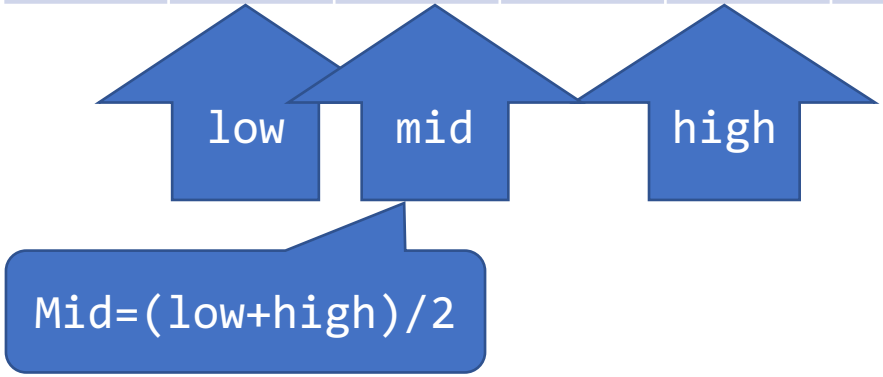


- ‘h’ > ‘d’, so need to keep searching in the *lower* half.
- Set `high = mid-1;`

Binary Search in Pictures

- Searching for ‘d’ in

value	‘a’	‘b’	‘d’	‘g’	‘h’	‘j’	‘k’	‘m’	‘p’
index	0	1	2	3	4	5	6	7	8



- ‘b’ < ‘d’, so need to keep searching in the *upper* half.
- Set `low = mid+1;`

Binary Search in Pictures

- Searching for ‘d’ in

value	‘a’	‘b’	‘d’	‘g’	‘h’	‘j’	‘k’	‘m’	‘p’
index	0	1	2	3	4	5	6	7	8

- ‘d’ equals ‘d’, return `mid (2)`

Reasoning about Coding Binary Search

- Going to loop `while (low <= high)`
  - Looping while there is anything left to search

- For correctness, want to maintain the following **loop invariant:**
  - If the target is in the array/list, it is in the range [low, high]

- At each step, either find the target and return, or…cut [low, high] in half without losing the target
  - Needs sortedness

Code for DIY Binary Search?

```
7   public static <T> int binarySearch(List<T> list, T target, Comparator<T> comp) {
8       int low = 0;
9       int high = list.size()-1;
10      while (low <= high) {
11          int mid = (low + high)/2;
12          T midval = list.get(mid);
13
14          int cmp = comp.compare(midval,target);
15          if (cmp < 0)
16              low = mid + 1;
17          else if (cmp > 0)
18              high = mid - 1;
19          else
20              return mid; // target found
21      }
22      return -1;  // target not found
23  }
```

<T> for generic type, can be a String list, Integer list, …, just need `target` and `Comparator` of the same type.

WOTO

Go to duke.is/gaucs

Not graded for correctness, just participation.

Try to answer *without* looking back at slides and notes.

But do talk to your neighbors!

2

What will index be after this call to binary search? *

```
29      String[] ar = {"ape", "bird", "cat", "dog", "elephant", "ferret
30      int index = Arrays.binarySearch(ar, "ape");
```

- ◯ -1
- ✓ 0
- ◯ 1
- ◯ 2

3

After running this code, index will be... *

```
31      String[] ar = {"cat", "dog", "dog", "bird", "hippo", "elephant"
32      int index = Arrays.binarySearch(ar, "ape", Comparator.comparing
```

- ◯ -1
- ◯ 0
- ✓ Can't tell because there are multiple possible correct values
- ◯ Can't tell because the elements are not in the correct sorted order
- ◯ Can't tell because there are duplicates in the array

4

How many calls to the compare method will result from the call to binary search in the main method on line 44? *

```
25    public static <T> int binarySearch(String[] array, String target, Compa
26        int low = 0;
27        int high = array.length-1;
28        while (low <= high) {
29            int mid = (low + high)/2;
30            String midval = array[mid];
31
32            int cmp = comp.compare(midval,target);
33            if (cmp < 0)
34                low = mid + 1;
35            else if (cmp > 0)
36                high = mid - 1;
37            else
38                return mid; // target found
39        }
40        return -1;  // target not found
41    }
      Run | Debug
42    public static void main(String[] args) {
43        String[] ar = {"cat", "dog", "dog", "bird", "hippo", "elephant"};
44        int index = StringSorting.binarySearch(ar, "snake", Comparator.comp
```

Select your answer

2

5

In the code shown above, is it important that we set low to mid+1 or high to mid-1 at each step instead of just setting low = mid or high = mid? *

- [x] Yes, it is important to prevent an infinite loop in edge cases
- [ ] Yes, it is important to have O(log(N)) complexity instead of O(N) complexity
- [ ] No, you could just use low=mid or high=mid in these cases

This content is neither created nor endorsed by Microsoft. The data you submit will be sent to the form owner.

Microsoft Forms

Finding the first or last?

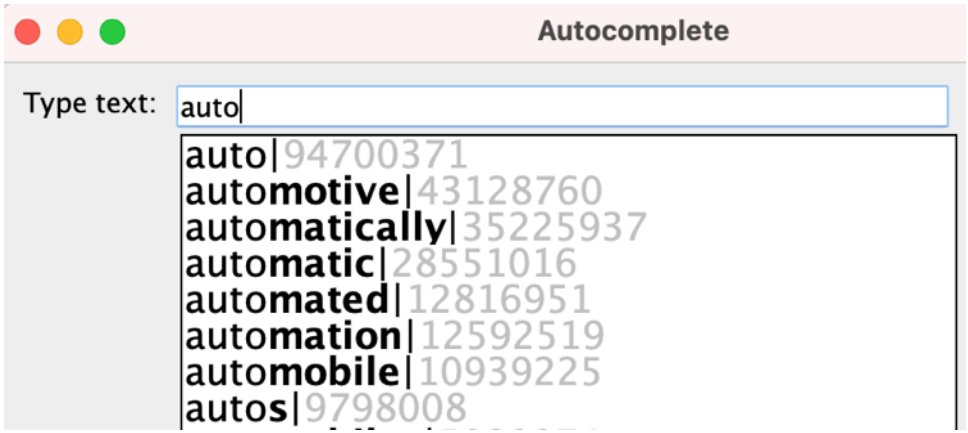
- Algorithm we have shown does *not* guarantee to find the first or last such index if there are multiple.

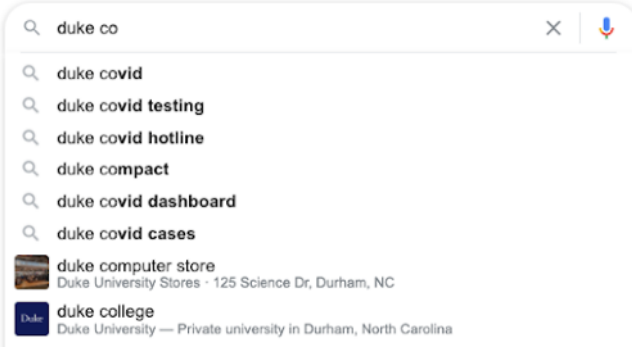
- You will reason about developing these modifications of binary search in Project 4 Autocomplete.

- Speaking of…

Project 4 Autocomplete

- How to create something like:





- All about two things:
  - Searching for all words that match on a prefix, and…
  - Sorting them by how common they are,
  - Return these words to show in the GUI above