CS 125 - Lecture 43

Objectives: Quicksort; BigO practice
Up next: all MPs regraded tonight;
turings craft tonight 11:59PM; break!

Final Exam: Tuesday Dec 13;
1:30PM - 4:30PM

Location: Watch your email!

Sample exams on CS 125 website

Conflict? Sign up on Piazza

Conflict Final Exam: Wed Dec 14; 7:00PM - 10:00PM

2. **QuickSort code**

12	**14**	**11**	**16**	**18**	**17**	**13**	**15**

```
static void quickSort(int[] data, int lo, int hi) {
      if (hi > lo) {

         int pivot = ?

         int newPivotIndex = ?

         quickSort(data, lo, newPivotIndex - 1);
         quickSort(data, newPivotIndex + 1, hi);
      }
   }
```

4. What is the Worst case Running Time for the following algorithms - "Big O" notation?

```
public static void foo2(int[] data) {
  int index = 0;
  while(index<data.length) {
    index+=Math.max(1,data[index]);
   }
}

public static void foo5(int[] array) {
   if(array.length ==0) return;
   int[] space = new int[array.length - 1];
   foo5(space);
   space[0]= 5;
}

public static double f1(double[][] array) {
   int N = array[0].length;
   double b = Math.random();
   for(int i = 0; i < 3; i++)
      for(int j = N - 1; j > 0; j--)
      b *= N * N * array[i][j];
   return b ;
}
```

1. **Partitioning an array into those elements less than a magic number and those elements greater than a magic number**

```
static int partition(int[] data,int lo,int hi,int magicIndex)
{
  // Move the magic number out of the way; for now we'll put
  // it at the start of the list and ignore it until the end.



 // Start working in, from both L and R ends of the list



  // The magic number will need to go to the left of the final
  // boundary if the last value is larger than the
  // magic number.



}
```

3. What is the Big O of the partition method for Quicksort?

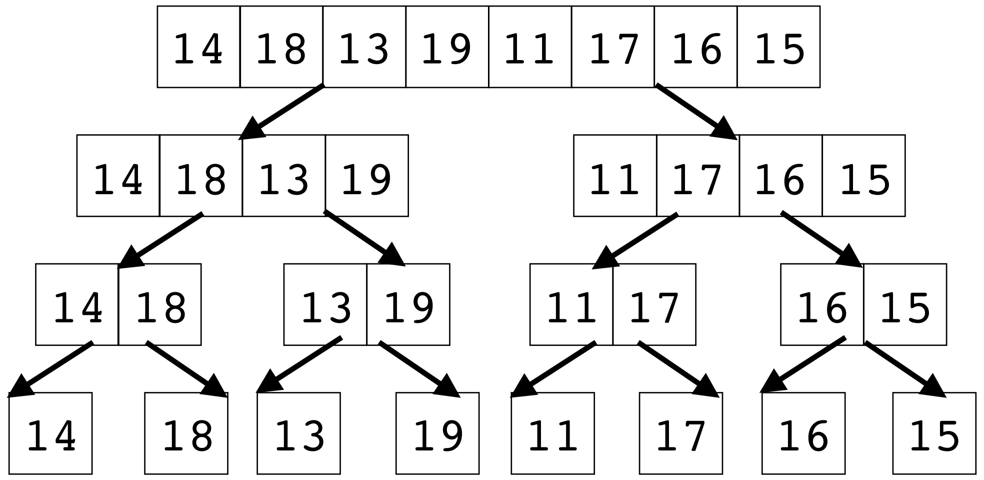
At best, how many levels does the Quicksort recursive tree have? Why?

At worst, how many levels does the Quicksort recursive tree have?

What is the best case Big O of the Quicksort algorithm?

How can we ensure that the worst case Quicksort never happens?

5. How many levels does the merge sort activation diagram have (roughly)? Why?



6a. At each level of the tree (j =0, 1, 2, ...), how many subproblems are there (as a function of N)?

6b. At each level of the tree (j =0, 1, 2, ...), how many values are in the array passed into the recursive activation (as a function of N)?

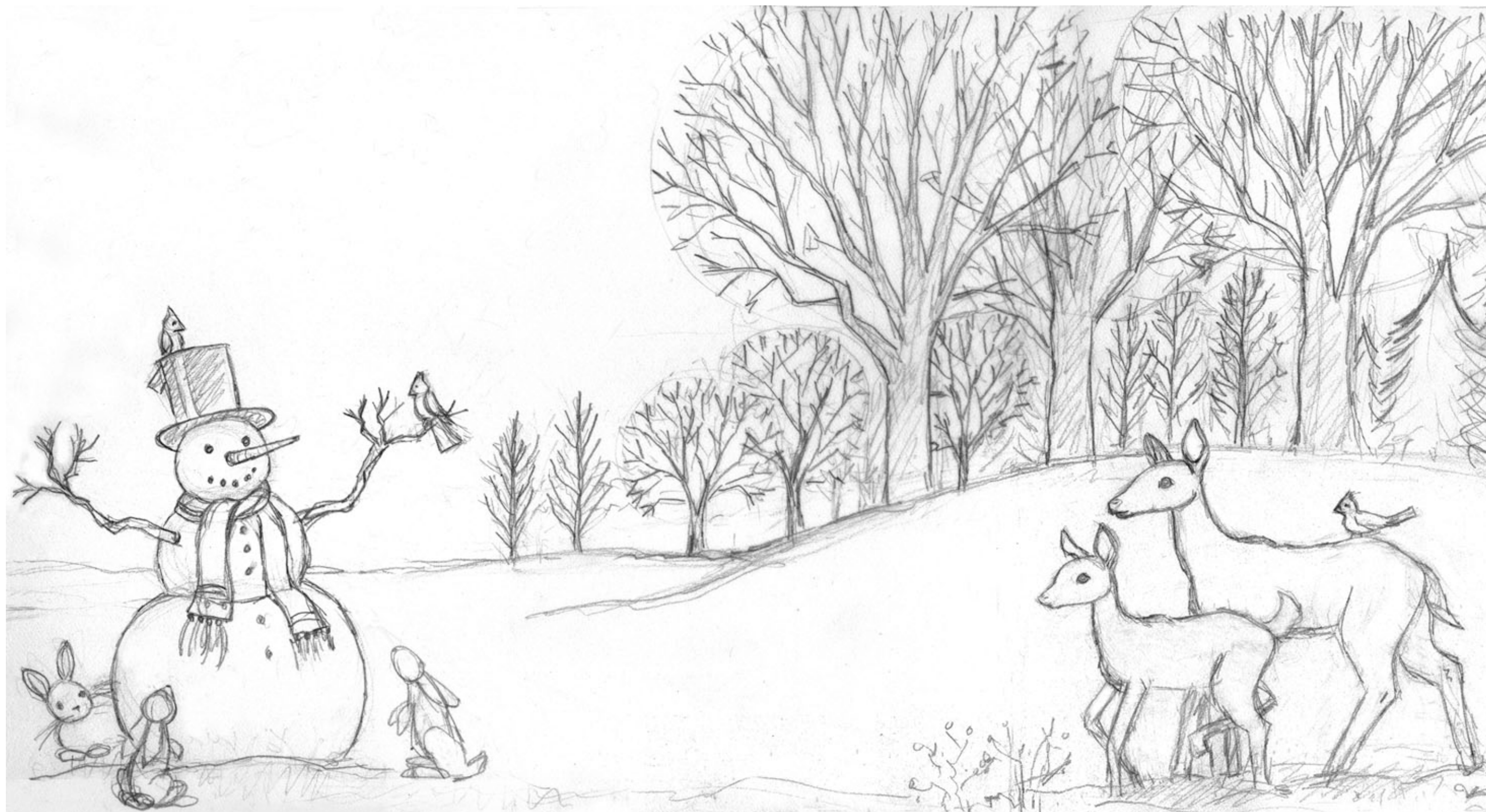
6c. What is the BigO of Merge Sort?

7. What is the worst-case running time of the following methods? Use Order-of-Growth "Big O" notation.

```
public static int f2(int[] arr) {
  // N is arr.length, N>>50
  int count = arr.length -1;
  while(count >0) {
    if(arr[count] < 0)
       count -= 7;
    count = count - arr.length / 50;
  }
  return count;
}
```

8. What is the worst-case running time of the following methods? Use Order-of-Growth "Big O" notation.

```
public static boolean f3(double[][] array) {
 //N is array.length
    for(int i=0;i<array.length;i++)
      for(int j=0;j<i;j++)
        if(array[i][j] == 0) return true;
   return false;
}
```



5. What is the worst-case running time of the following methods? Use Order-of-Growth "Big O" notation.

```
public static double f1(double[][] array) {
   int N = array[0].length;
   double b = Math.random();
   for(int i = 0; i < 3; i++)
      for(int j = N - 1; j > 0; j--)
      b *= N * N * array[i][j];
   return b ;
}
```

6. What is the worst-case running time of the following methods? Use Order-of-Growth "Big O" notation.

```
public static int f2(int[] arr) {
  // N is arr.length, N>>50
  int count = arr.length -1;
  while(count >0) {
    if(arr[count] < 0)
       count -= 7;
    count = count - arr.length / 50;
  }
  return count;
}
```

7. What is the worst-case running time of the following methods? U
Order-of-Growth "Big O" notation.

```
public static boolean f3(double[][] array) {
 //N is array.length
    for(int i=0;i<array.length;i++)
      for(int j=0;j<i;j++)
       if(array[i][j] == 0) return true;
   return false;
}
```

8. What is the worst-case running time of the following methods? Use Order-of-Growth "Big O" notation.

```
public boolean f4(int[] arr, int lo, int hi) {
// N = initial value of (hi-lo+1) of
// 1st activation of f4
 if(lo == hi) return false;
 if(arr[hi] * arr[lo]>10)
    return true;
  return f4(arr,lo+1 , hi);
}
```

9. What is the worst-case running time of the following methods? Use Order-of-Growth "Big O" notation.

```
public int f5(int[] array, int x) {
   // N = array.length
   int m = array.length-1;
   while(array[m] > x && m>0)
      m=m/2;
   return m;
```