CS-3410 **Algorithms** **Spring 2026**

Homework 2 **Brandon Aikman**

1. Exercise 6.1-1 (p163)
   The minimum number of elements in a heap of height h is 2^h, and the maximum number of elements in a heap of height h is $2^{h+1}-1$. A heap of height h has at least one element at depth h, so all levels 0 through $h-1$ must be full, giving a minimum of 2^h element. The maximum occurs when all levels 0 through h are full, yielding $2^{h+1}-1$ elements.

2. Evaluate [33, 19, 20, 15, 13, 10, 2, 13, 16, 12].
   The array with values [33, 19, 20, 15, 13, 10, 2, 13, 16, 12] is not a max-heap. The right child of 15 (index 4) is at index $4 \cdot 2 + 1 = 9$, corresponding with key 16, which is greater than 15, violating the max-heap property.

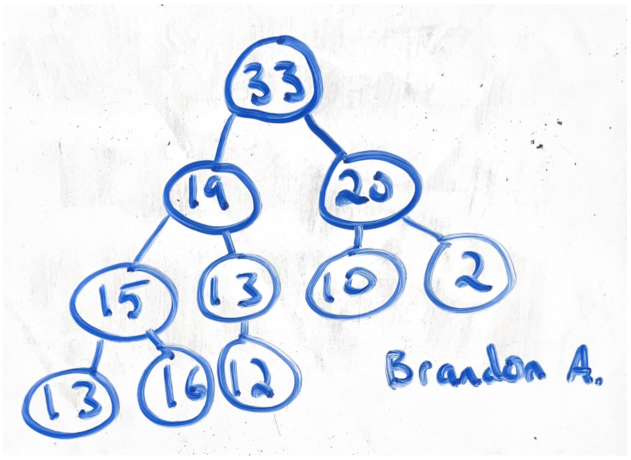

Figure 1: Max-heap

3. Exercise 7.1-2 (p187)
   When all elements in the subarray $A[p:r]$ have the same value, `Partition` returns r, because every element satisfies $A[j] \leq x$, so the pivot is swapped into r. To ensure that $q = \lfloor (p+r)/2 \rfloor$ when all elements in the subarray $A[p:r]$ have the same value, we can modify `Partition`. When all elements are equal to the pivot, we can limit only the first $\lfloor (r-p)/2 \rfloor$ elements to be placed to the left of the pivot, forcing the pivot to index $\lfloor (p+r)/2 \rfloor$. This modification applies only in the case where all elements are equal to the pivot, otherwise `Partition` behaves as normal without affecting correctness on general inputs.

4. Exercise 7.2-4 (p191)
   On nearly sorted input, the number of inversions is small, so insertion sort runs in near-linear time, whereas quicksort may still perform poorly due to unbalanced partitions. Insertion sort would tend to beat quicksort on problems of sorting 'almost-sorted' input because insertion sort's linear

runtime for sorted data is $O(n)$, while quicksort's runtime on sorted data achieves its worst-case of $O(n^2)$.

5. Exercise 7.4-3 (p198)
   We analyze the function $q^2 + (n-q-1)^2$ to see where quicksort's runtime is maximized. Because $\frac{d^2}{dq^2}[q^2+(n-q-1)^2] = 4$, we know that any critical points on the interval $[0, n-1]$ will be minima. Because we see that $\frac{d}{dq} = 4q - 2n + 2$, the minimum occurs at $q = \frac{n-1}{2}$. Thus, we see that the maxima occur at the endpoints $q = 0, n-1$.

6. Exercise 8.4-2 (p218)
   The worst-case runtime for bucket sort is $\Theta(n^2)$ because there could be a bucket which has all n values from an array. Because bucket sort uses insertion sort, which has a worst-case runtime of $O(n^2)$, sorting a bucket with all n values in it would take $O(n^2)$. Simply changing the sorting algorithm per bucket to one that has $O(n \log n)$ runtime, like merge sort, would preserve its linear average-case runtime while limiting its worst-case runtime at $O(n \log n)$.

7. Exercise 9.2-2 (p236)

```
Randomized-Select-Iterative(A,p,r,i)
  if p == r
    return A[p]
  while p < r
    q = Randomized-Partition(A,p,r)
    k = q - p + 1
    if i == k
      return A[q]
    if i < k
      r = q - 1
    else
      p = q + 1
      i = i - k
  return A[p]
```